Methods in Molecular Biology 2243

Springer Protocols

Noam Shomron *Editor*

Deep Sequencing Data Analysis

Second Edition

Humana Press

METHODS IN MOLECULAR BIOLOGY

Series Editor
John M. Walker
School of Life and Medical Sciences
University of Hertfordshire
Hatfield, Hertfordshire, UK

For further volumes:
http://www.springer.com/series/7651

For over 35 years, biological scientists have come to rely on the research protocols and methodologies in the critically acclaimed *Methods in Molecular Biology* series. The series was the first to introduce the step-by-step protocols approach that has become the standard in all biomedical protocol publishing. Each protocol is provided in readily-reproducible step-by-step fashion, opening with an introductory overview, a list of the materials and reagents needed to complete the experiment, and followed by a detailed procedure that is supported with a helpful notes section offering tips and tricks of the trade as well as troubleshooting advice. These hallmark features were introduced by series editor Dr. John Walker and constitute the key ingredient in each and every volume of the *Methods in Molecular Biology* series. Tested and trusted, comprehensive and reliable, all protocols from the series are indexed in PubMed.

Deep Sequencing Data Analysis

Second Edition

Edited by

Noam Shomron

Sackler Faculty of Medicine, Tel Aviv University, Tel Aviv, Israel

Humana Press

Editor
Noam Shomron
Sackler Faculty of Medicine
Tel Aviv University
Tel Aviv, Israel

ISSN 1064-3745 ISSN 1940-6029 (electronic)
Methods in Molecular Biology
ISBN 978-1-0716-1102-9 ISBN 978-1-0716-1103-6 (eBook)
https://doi.org/10.1007/978-1-0716-1103-6

This Humana imprint is published by the registered company Springer Science+Business Media, LLC, part of Springer Nature.
The registered company address is: 1 New York Plaza, New York, NY 10004, U.S.A.

Preface

I recall reading my first "NGS" paper (in 2005), during my postdoc at MIT, when a company with a strange name—454 [1] (454 Life Sciences, later purchased by Roche)—described its method to sequence millions of nucleotides in every experimental run. While reading the methodology, I felt like I was entering a great science fiction novel or watching a Mission Impossible movie. "How could all this work…" I wondered. There are so many potential pitfalls along the process, and yet the publication was solid and the results convincing. I admired the engineers and the biologists who devised it all. Before long we started getting used to these DNA churning machines. Our appetites grew and we sought more and more data from every spin of the experimental wheel. A few years later we were flabbergasted again with the report of a new machine with even larger capacity. This time it was the Solexa technology [2] (purchased by Illumina). Not long after, the Pacific Biosciences novel single molecule reads [3] apparatus matured, followed by the Ion Torrent [4] from Life Technologies, which skipped fluorescence and luminescence, and then the SOLiD [5] system from Applied Biosystems. This was followed by exciting advances in Nanopore readers [6] (from Oxford Nanopore Technologies). Fifteen years have passed since the first NGS publication. During this period, sample preparation protocols have been established (at first machines came without instructions or preparation kits); secrets of how to obtain the most reads out of the DNA sequencers have passed from one technician to another (setting the air conditioner in the lab to the minimum temperature, for example, increases the nucleotide output); and bioinformaticians have learned to deal with error-prone (very) short DNA reads (at first Illumina reads, for example, were limited to 35 nucleotides). In parallel, engineers, chemists, and biologists have developed advanced machines and supportive protocols to improve outputs. All along, the bioinformaticians, computational biologists, and computer scientists have supported these technologies. Their goals have been to ensure that the read output matches the genuine nucleotide sequence, and that the data analysis is accurate. In our second edition of the book, entitled *Deep Sequencing Data Analysis*, under the "Methods in Molecular Biology" series, leading authors contributed to the multidimensional task of deep sequencing data analysis. We start by describing methods of detecting detrimental variants using whole genome sequencing, statistical considerations for inferring copy number variations, whole-metagenome shotgun sequencing studies and 16S amplicon sequencing, mapping the accessible chromatin landscape, and (small/single cell) RNA sequencing (RNA seq). In our coverage of computational oriented methods, we discuss the use of deep learning for data analysis and genome-wide noninvasive prenatal diagnosis (NIPD) of SNPs, indels, and *de novo* mutations. We end with specialized topics that deal with accurate imputation of untyped variants, multi-region sequence analysis to predict intratumor heterogeneity, and cancer classification. We present several topics as primers for the bioinformatics student, while we take an in-depth dive for the professional readers. The topics should be of great use for both beginner and savvy bioinformaticians when tackling deep sequencing data analysis.

Tel Aviv, Israel ***Noam Shomron***

References

1. Margulies M et al (2005) Genome sequencing in microfabricated high-density picolitre reactors. Nature 437(7057):376–380
2. Bentley DR et al (2008) Accurate whole human genome sequencing using reversible terminator chemistry. Nature 456(7218):53–59
3. Lundquist PM et al (2008) Parallel confocal detection of single molecules in real time. Opt Lett 33 (9):1026–1028
4. Rothberg JM et al (2011) An integrated semiconductor device enabling non-optical genome sequencing. Nature 475(7356):348–352
5. McKernan KJ et al (2009) Sequence and structural variation in a human genome uncovered by short-read, massively parallel ligation sequencing using two-base encoding. Genome Res 19(9):1527–1541
6. Kasianowicz JJ et al (1996) Characterization of individual polynucleotide molecules using a membrane channel. Proc Natl Acad Sci U S A 93(24):13770–13773

Contents

Contributors

Soyeon Ahn • *Division of Statistics, Medical Research Collaborating Center, Seoul National University Bundang Hospital, Seongnam, Republic of Korea*

Amnon Amir • *Microbiome Center, The Chaim Sheba Medical Center, Tel-Hashomer, Ramat-Gan, Israel*

T. Daniel Andrews • *John Curtin School of Medical Research, Australian National University, Canberra, ACT, Australia*

François Belzile • *Département de Phytologie, Université Laval, Québec City, QC, Canada; Institut de Biologie Intégrative et des Systèmes (IBIS), Université Laval, Québec City, QC, Canada*

Vered Chalifa-Caspi • *Bioinformatics Core Facility, Ben-Gurion University of the Negev, Beer-Sheva, Israel*

Jie Chen • *Division of Biostatistics and Data Science, Department of Population Health Sciences, Medical College of Georgia, Augusta University, Augusta, GA, USA*

Artem Danilevsky • *Faculty of Medicine, Tel Aviv University, Tel Aviv, Israel*

Matt A. Field • *John Curtin School of Medical Research, Australian National University, Canberra, ACT, Australia; Centre for Tropical Bioinformatics and Molecular Biology, Australian Institute of Tropical Health and Medicine, James Cook University, Cairns, QLD, Australia*

Sydnie Grabell • *Faculty of Medicine, Tel Aviv University, Tel Aviv, Israel*

Abdul Rezzak Hamzeh • *John Curtin School of Medical Research, Australian National University, Canberra, ACT, Australia*

Haiyan Huang • *Department of Statistics, University of California, Berkeley, CA, USA; Center for Computational Biology, University of California, Berkeley, CA, USA*

Tyler A. Joseph • *Department of Computer Science, Fu Foundation School of Engineering & Applied Science, Columbia University, New York, NY, USA*

Tomer Kalisky • *BIU, Department of Bioengineering, Bar-Ilan University, Ramat Gan, Israel*

Itamar Kanter • *BIU, Department of Bioengineering, Bar-Ilan University, Ramat Gan, Israel*

Wendao Liu • *Faculty of Medicine, Tel Aviv University, Tel Aviv, Israel*

Georgi K. Marinov • *Department of Genetics, Stanford University, Stanford, CA, USA*

Yaron Orenstein • *Ben-Gurion University of the Negev, Beersheba, Israel*

Metsada Pasmanik-Chor • *Bioinformatics Unit, G.S.W. Faculty of Life Science, Tel Aviv University, Tel Aviv, Israel*

Itsik Pe'er • *Department of Computer Science, Fu Foundation School of Engineering & Applied Science, Columbia University, New York, NY, USA*

Ravit Peretz-Machluf • *Faculty of Medicine, Tel Aviv University, Tel Aviv, Israel*

Tom Rabinowitz • *Faculty of Medicine, Tel Aviv University, Tel Aviv, Israel*

Guy Shapira • *Faculty of Medicine, Tel Aviv University, Tel Aviv, Israel*

Zohar Shipony • *Department of Genetics, Stanford University, Stanford, CA, USA*

Noam Shomron • *Faculty of Medicine, Tel Aviv University, Tel Aviv, Israel*
Yue Yang (Alan) Teo • *Faculty of Medicine, Tel Aviv University, Tel Aviv, Israel*
Davoud Torkamaneh • *Département de Phytologie, Université Laval, Québec City, QC, Canada; Institut de Biologie Intégrative et des Systèmes (IBIS), Université Laval, Québec City, QC, Canada; Department of Plant Agriculture, University of Guelph, Guelph, ON, Canada*
Gur Yaari • *BIU, Department of Bioengineering, Bar-Ilan University, Ramat Gan, Israel*
Yazeed Zoabi • *Faculty of Medicine, Tel Aviv University, Tel Aviv, Israel*

Chapter 1

Detecting Causal Variants in Mendelian Disorders Using Whole-Genome Sequencing

Abdul Rezzak Hamzeh, T. Daniel Andrews, and Matt A. Field

Abstract

Increasingly affordable sequencing technologies are revolutionizing the field of genomic medicine. It is now feasible to interrogate all major classes of variation in an individual across the entire genome for less than $1000 USD. While the generation of patient sequence information using these technologies has become routine, the analysis and interpretation of this data remains the greatest obstacle to widespread clinical implementation. This chapter summarizes the steps to identify, annotate, and prioritize variant information required for clinical report generation. We discuss methods to detect each variant class and describe strategies to increase the likelihood of detecting causal variant(s) in Mendelian disease. Lastly, we describe a sample workflow for synthesizing large amount of genetic information into concise clinical reports.

Key words Variant detection, Variant annotation, Clinical reports, SNV, Copy number variation, Missense mutation, Mendelian disease

1 Introduction

Rapid improvements in sequencing technologies have made it feasible to interrogate an individual's genetic information in order to identify disease-causing and risk-factor variants [1–4]. While computational algorithms aimed at deriving accurate data from these technologies are increasingly mature [5], the routine use of genomic data to improve clinical diagnosis and treatment requires formalizing existing research methods into clinical best practices. Several clinical sequencing options are now widely available including targeted gene panel sequencing, whole-exome sequencing (WES), or whole-genome sequencing (WGS). These options differ with regard to the portion of the genome examined as well as the class of variants that can be detected.

Many clinical sequencing projects are increasingly looking to whole genome sequencing (WGS) with initiatives like Genome England's 100,000 Genomes Project solely employing WGS.

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_1,

While such large-scale initiatives have been successful and improved the outcome for many patients, they have also made it increasingly clear that the clinical bottleneck has shifted from data generation to data interpretation. While steps such as sequence data generation and read alignment have become increasingly standardized, the functional interpretation of the increasingly large volume of patient variation information remains largely unresolved. While great strides have been made in the form of functional assays able to more accurately predict and characterize the functional impact of variation, we still lack a deep understanding of the complexity of genome function. This results in a gap in our ability to relate genotype to phenotype and represents the greatest hurdle to large-scale implementation of patient clinical sequencing. In this chapter, we will describe how to detect, annotate, prioritize, and report patient variation consistently in order to begin to shrink this gap.

2 Variant Detection Workflow

A typical clinical variant detection workflow is shown in Fig. 1.

Generally, clinical variant detection workflows consist of sequence QC, read alignment, BAM alignment file preprocessing, variant detection, variant annotation, gene annotation, variant prioritization, and clinical report generation. The steps of sequence QC, read alignment, and BAM preprocessing have become relatively standardized with multiple mature software packages available. For example, sequence data QC options include Trimmomatic [6] or NGS QC Toolkit [7], while read alignment can be done with BWA [8] or Bowtie2 [9] for DNA and STAR [10] or TopHat [11] for RNA. After initial read alignment, BAM files are preprocessed for variant detection following GATK best practices [12], and variant detection algorithm(s) run which aim to differentiate true genetic variation from experimental error. Variant detection algorithms employ sophisticated statistical computational methodologies which are generally specific to a single type of variation. While most early clinical sequence workflows focused on identifying SNVs and small insertions/deletions (indels), the shift to WGS has enabled the detection of all types of variation including noncoding SNVs and small indels, structural and copy number variants, and repeat variants. With all major variant classes implicated in causing human disease [13], it is increasingly common to run algorithms for each variant class.

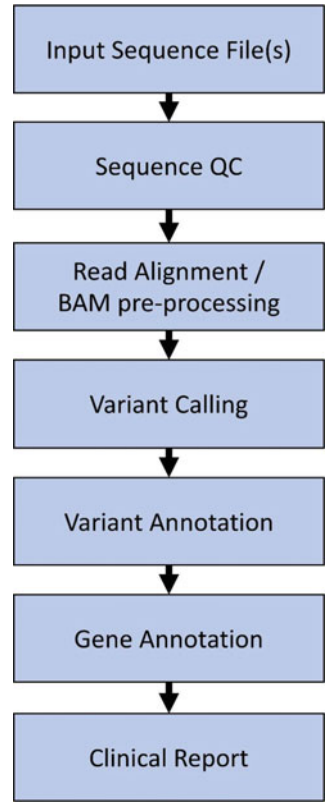


Fig. 1 Typical steps in clinical variant detection workflow

3 Variation Types

3.1 SNVs and Small Indels

The vast majority of variation in the human genome consists of single nucleotide polymorphisms and small indels (<50 bp). Any human genome aligned to the standard reference genome (currently GRCh38) results in roughly four million SNVs and 400,000 small indel calls. Many algorithms exist for detecting SNVs and small indels, with most tools detecting both types of variation in a single pass. Some of the most commonly used algorithms used for SNV/small indel detection are listed in Table 1.

Most algorithms require a BAM alignment file as input and output a variant call format (vcf) file. Some programs run more than one variant detection algorithm and employ a consensus approach (BAYSIC [14] and appreci8 [15]) while others support the use of molecular tagging techniques in order to detect variation within sequence reads derived from individual input DNA molecules (DeepSNVMiner [16] and smCounter2 [17]). Overall, SNV/small indel algorithms achieve the highest accuracy relative to other variants types with a recent review showing all algorithms achieving F-scores >0.975 for SNVs and >0.85 for small indels [18].

Table 1
Commonly used germline variant callers for SNVs/indels

Tool	Programming language	Variant type	Multiple tools	UMI barcode handling
GATK Haplotype [12]	Java	SNV, indel	No	No
SAMtools [85]	C	SNV, indel	No	No
VarScan/VarScan2 [86]	Java	SNV, indel	No	No
Platypus [87]	Python	SNV, indel, SV	No	No
Strelka2 [88]	Python	SNV, indel	No	No
BAYSIC [14]	Perl/R	SNV	Yes	No
VarDict [89]	Java/Perl	SNV, indel, SV	No	No
DeepSNVMiner [16]	Perl/R	SNV, indel	No	Yes
LoFreq [90]	C/python	SNV, indel	No	No
Appreci8R [15]	R	SNV, indel	Yes	No
smCounter2 [16]	Python	SNV, indel	No	Yes

3.2 Structural/Copy Number Variation

Structural variation (SV) is defined as variants >50 bp that can be classified as deletions, insertions, duplications, inversions, and translocations. SVs are further classified as balanced or unbalanced based on whether they alter the size of the resultant genome. Inversions and translocations events are classified as balanced, while deletions and duplications (collectively referred to as copy number variants or CNVs) and insertions are referred to as unbalanced. While somewhat arbitrary, SVs are identified separately from SNVs and small indels due to the distinct mechanism in how they are formed [19]. In any human genome, there are far fewer SVs compared to SNVs and small indels (<10,000 for all SV types); however, their larger size means they are more likely to have an impact on function. SVs are more challenging to detect and resolve with short-read data than SNV/small indels partly because SVs are often longer than the sequence read length. Additional challenges arise as each SV type requires a distinct algorithm due to its unique read alignment pattern with the problem further confounded by SV breakpoints being enriched in repetitive regions where short read aligners struggle [20]. SV detection algorithms vary in both the types of SVs they detect and the types of alignment evidence they use in detection. Some of the most commonly used algorithms used for SV/CNV detection are listed in Table 2.

Table 2
Commonly used germline variant callers for structural/copy number variation

Tool	Programming language	Variant type	Split reads	Paired reads	Assembly
Delly [91]	C++	DEL, INV, DUP, TRANS	Yes	Yes	No
Lumpy [92]	C++	DEL, INV, DUP, TRANS, INS	Yes	Yes	No
Manta [93]	Python	DEL, INV, DUP, TRANS, INS	Yes	Yes	Yes
CLEVER [94]	Python	DEL, INS	No	Yes	No
GridSS [22]	Java/R	DEL, INV, DUP, TRANS, INS	Yes	Yes	Yes
Breakdancer [95]	C++/Perl	DEL, INV, TRANS, INS	No	Yes	No
PRISM [96]	C	DEL, INV, DUP	Yes	Yes	No
CNVnator [21]	C++	DEL, DUP	No	No	No
HYDRA [23]	C++	DEL, INV, DUP, INS	Yes	No	Yes
Pindel [97]	C++	DEL, INS	Yes	No	No
CREST [24]	Perl	DEL, INV, DUP, TRANS, INS	Yes	Yes	Yes

In general, SV detection algorithms rely on three main types of evidence: read depth, discordant read pairs, and split reads. Methods like CNVnator [21] identify regions of increased/decreased read depth across the genome (an approach similar to previous array-based methods such as microarray comparative genome hybridization) but offers greater resolution in determining exact breakpoints and detecting smaller SVs. To identify SV hotspots, most algorithms rely on both discordant read pairs (exhibiting either aberrant insert size, incorrect read pair orientation, or mate pairs aligned to different chromosomes) and split reads (reads with truncated alignments preserved through an alignment process called soft-clipping). When a hotspot is identified, some algorithms use additional assembly-based methods to generate contigs in order to more succinctly resolve potential SV breakpoints (GridSS [22], Hydra [23], and CREST ([24]).

SV detection algorithms are less accurate than SNV/small detection algorithms with performance varying significantly with regard to both SV type and algorithm. A recent review of 69 algorithms showed the highest accuracy for deletions (precision 0.74–0.91 and recall 0.20–0.28) compared to the lowest accuracy for duplications (precision 0.40–0.57 and recall 0.07–0.14)

[25]. Such results highlight performance differences across both algorithm and SV type and has resulting in most large-scale projects running >1 SV detection algorithm and employing a downstream consensus-based approach [26].

3.3 Repeat Variation

High copy repeat variation consists broadly of mobile elements and tandem repeats with tandem repeats further classified by size into microsatellites (1–6 bp; also called short tandem repeats), minisatellites (7–49 bp), and larger satellite repeats typically found in centromeres and heterochromatin regions. Due to the challenges of detecting this type of variation the estimated number per human genome is less certain; however, current estimates are ~10,000 tandem repeats and ~ 2000 mobile elements per genome [27]. Reliable detection of repeat variation is challenging with short reads due to issues with accurate read alignment and the accuracy of the reference genome in these regions. Some of the most commonly used algorithms for repeat variant detection are listed in Table 3.

In general, the repeat variation detection algorithms are extremely specialized employing a wide variety of approaches. For example, RepeatExplorer2 [28] uses graph-based clustering of reads and identifies different families of repetitive elements based on the clustering patterns, patterns which can be further analyzed with regard to the sequence composition and abundance of these repeat variants. STRetch [29], on the other hand, constructs a set of additional STR sequences comprising all possible 1–6 bp repeats which are added to the reference genome as decoy sequences.

Table 3
Commonly used germline variant callers for repeat variation

Tool	Programming language	Variant type
STRetch [29]	Python/R	STR
ExpansionHunter [98]	C++	STR
HipSTR [99]	C++	STR
exSTRa [100]	Perl/R	STR
TREDPARSE [101]	Python/C	STR
RepeatExplorer2 [28]	Python/C++	STR, Minisatellites, Satellites, Mobile elements
Tandem Repeats Finder [102]	Desktop	STR, Minisatellites, Satellites
Phobos [103]	Desktop	STR, Minisatellites, Satellites
RetroSeq [104]	Perl	Mobile elements
TanGram [105]	C++	Mobile elements

Alignment to the modified genome identifies reads aligning to the decoy as short tandem repeat candidates. Overall, the accuracy of repeat variation algorithms is likely the lowest among all variant classes; however, it is difficult to benchmark due to the lack of gold standard reference databases.

4 Variant Annotation Algorithms

Following variant detection several computational algorithms are run which provide critical variant annotation information used downstream for prioritization. These steps include determining the impact of the variants on proteins and predicting how damaging missense mutations are likely to be to protein function.

4.1 Variant Impact on Proteins

Typically, variants are compared to a gene model such as ENSEMBL or RefSeq [30] to determine their potential functional impact on proteins. A variety of software annotates variants relative to a gene model, two of the most popular being the Variant Effect Predictor (VEP) [31] and AnnoVar [32]. Overall, the algorithms are similar; however, they differ in terms of whether additional gene models are supported and also in how they determine the variant effect when multiple overlapping transcripts are available for a gene. For example, VEP offers an option to only consider the impact on the "canonical" transcript (generally the longest CCDS transcripts with no stop codon); however, the canonical transcript does not necessarily represent the most biologically active transcript meaning the annotations may be inaccurate when noncanonical transcripts are biologically active.

Following annotation, variants are broadly grouped into coding and noncoding with coding variants further divided based on their effect on the protein amino acid sequence. Coding SNVs are classified as missense (alter amino acid sequence), nonsense (generate stop codon), or synonymous (no effect on amino acid sequence), while coding indels are divided into frameshift or nonframeshift based on whether their length is a factor of 3. Noncoding SNVs and indels are overlapped to a variety of features known to influence gene function including splice sites, 3′ and 5′ UTRs, miRNAs, or known regulatory elements. Splice site mutations are enriched for causal variants often resulting in intron retention, exon skipping, or exon creation, all of which may lead to the production of aberrant proteins [33]. Larger and more complex variants (SVs and repeats) are examined in terms of their proximity to genes and exons, the potential rearrangement of known regulatory elements, and any possible gene fusion transcripts generated. For SVs, the genomic region to examine is affected by the SV type with balanced SVs limited to the immediate breakpoint region compared to CNVs where the entire duplicated/deleted region is examined.

4.2 Missense Mutation Prediction Tools

Missense mutations often cause disease; however, any human genome contains hundreds of such mutations most of which produce no discernible phenotype. Currently, it is not feasible to functionally test all missense mutations so computational tools have been developed to predict how damaging any given mutation is likely to be to protein function. These tools are usually trained on both known disease mutations and common polymorphisms; however, they have yet to be tested against an unbiased spectrum of random de novo mutations. A variety of algorithms exist that rely on four types of evidence: sequence conservation, protein structure, annotations, and training data (Fig. 2).

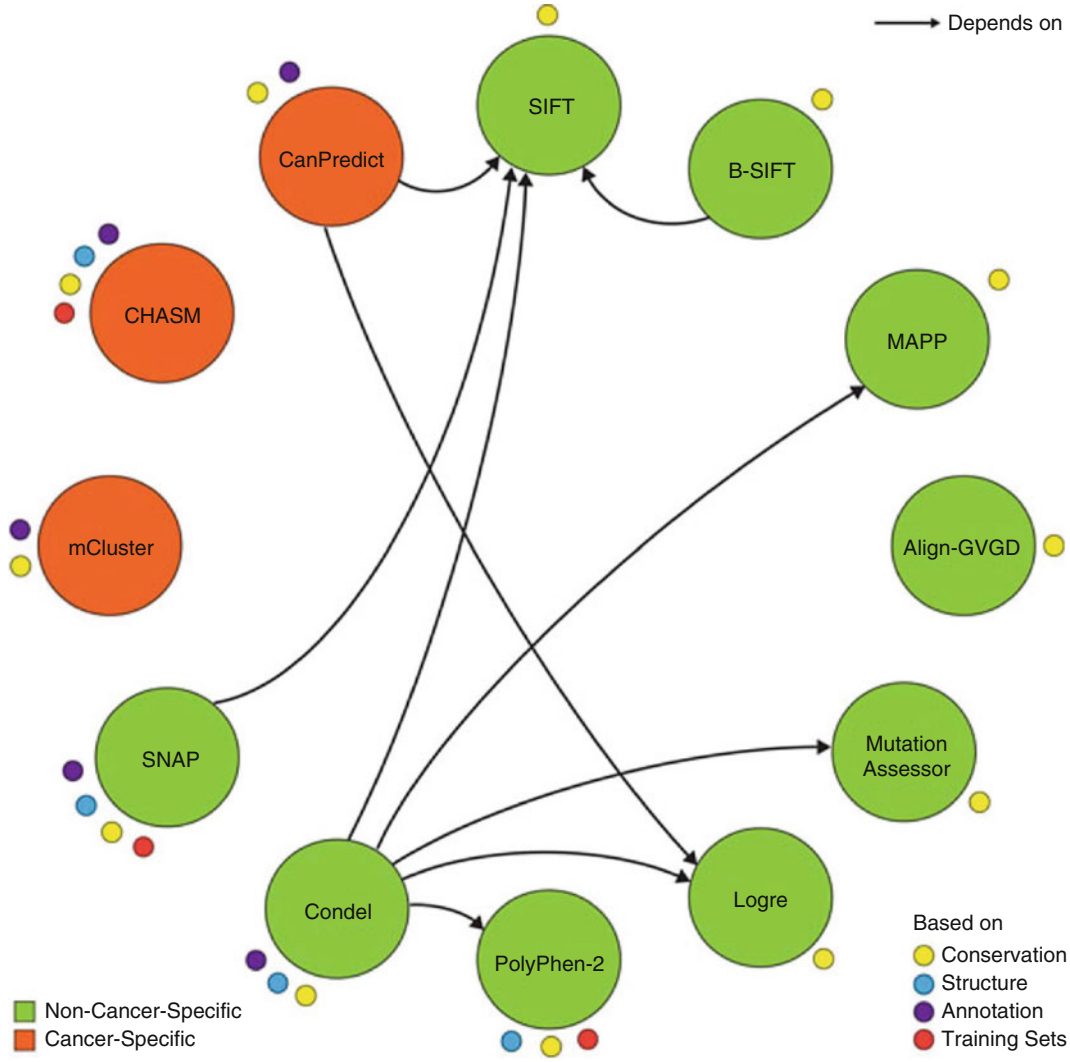


Fig. 2 Missense mutation prediction tools and evidence types utilized

It has been shown that these algorithms suffer from high false positive rates and often the predictions do not track with clinical phenotype [34]. This is reflected by the American College of Medical Genetics and Genomics (AGMG) recommendations where they state, "These are only predictions, however, and their use in sequence variant interpretation should be implemented carefully. It is not recommended that these predictions be used as the sole source of evidence to make a clinical assertion." Likely the high false positive rate can be explained by the heavy reliance of all the tools on sequence conservation. Heavily relying on sequence conservation effectively measures purifying selection; however, not all variants under purifying selection result in a clinical phenotype. Some variants only generate a phenotype under specific environmental conditions, the so-called "nearly neutral" mutations first described by Tomoko Ohta in 1992. Currently these tools are unable to differentiate immediately clinically relevant mutations from nearly neutral mutations [34].

5 Specialized Strategies

Strategies to increase diagnosis rates can broadly be broken down into three categories: sample selection, sequencing technology selection, and software optimization.

5.1 Sample Selection Strategies

While not always possible, choosing which sample(s) to sequence can increase the likelihood of identifying disease-causing variants. If only sequencing a single individual, a strategy of focusing on early-onset cases with extreme phenotypes that are clinically well-defined has been shown to increase diagnosis rates [35]. In such cases rare or de novo mutations are most likely causal, particularly when no other family members are known to be affected. Another example where a single individual is often sufficient is within consanguineous families where homozygous mutations are first examined thus greatly reducing the total genetic search space [36].

When multiple samples are available for sequencing, if the individuals are unrelated, it is critical to focus on patients with a well-defined shared phenotype. In such instances a common mutation can be found [37, 38]; however, it is often necessary to employ gene network analysis software (e.g., Ingenuity [39] or String [40]) to identify mutations affecting the same disease pathway via different genes.

The greatest success rates in detecting causal variants in Mendelian diseases currently comes with sequencing multiple individuals within a family or pedigree. While clinical sequencing has been successfully in discovering causal variation using small numbers of unrelated individuals [41] it is clear this approach is insufficient to reliably identify the underlying genetic causes in many cases

[42]. With pedigree sequencing the search space for causal variants is reduced, by both the prioritization of variants common to affected individuals and the exclusion of benign variants shared between affected and unaffected individuals. The effective analysis of sequenced pedigrees requires tools capable of combining variant specific and pedigree wide annotations to dramatically reduce the causal variation search space. Generally tools focus on either progressively removing variants based on criteria deemed unlikely to be causal [43] or by focusing on variants matching specific inheritance models (compound heterozygotes [44]; autosomal dominant [45]). Other tools like Gemini [46] or VASP [47] do not make any assumptions regarding disease inheritance.

5.2 Sequencing Strategies

Alternatives to standard Illumina-based whole genome sequencing such as UMI barcoding, transcriptome sequencing, and long read sequencing are increasingly being used to improve diagnosis rates.

Unique molecular identifier (UMI) molecular tagging is useful when the input DNA is derived from heterogeneous cell mixtures containing both wild-type and disease-related states resulting in low frequency variants. Molecular tagging works by affixing a unique identifier to all sequence reads derived from an individual DNA molecule resulting in read groups sharing a common sequence tag that derive from a single input DNA molecule. Read disambiguation generates huge numbers of reads groups, each of which requires custom variant detection analyses. While lab-based techniques for producing these data are approaching maturity [48], the relative immaturity of computational tools for analyzing such heterogeneous sequence data remains an obstacle to the widespread adoption of this technology.

Another increasingly common strategy is to utilize both DNA and RNA sequencing for a single patient in order to identify disease causing noncoding variants [49]. This approach uses functional genomic information obtained by RNA-Seq to identify transcriptional perturbations likely caused by genetic changes [50]. Great improvements in diagnosis rates have been reported with a recent study using RNA-Seq to improve diagnosis rates by 35% by identifying deep splicing variants causing exon skipping, exon extension, and intronic splice gains [49]. It should be noted for this approach to work it is critical to sequence the disease specific tissue and compare RNA-Seq to a reference panel of the same tissue type.

Finally, long read sequencing technologies such as Oxford Nanopore and PacBio are increasingly being combined with short reads in order to resolve repeat and structural variants, regions which are challenging to resolve solely with short reads [51]. While unlikely to replace short read technologies in the near future due to their higher error rates and increased cost per base, they are increasingly being utilized in instances where short reads alone are unable to detect or resolve causal variant(s).

5.3 Software Strategies

Finally, software strategies such as running multiple variant callers and employing a consensus-based approach are being employed to increase diagnosis rates. This strategy has arisen because it has become increasingly clear that no single variant caller for any variant type performs optimally under all conditions [18, 25]. Several studies have shown that combining variant calls of multiple tools results in the best quality resultant variant set, for either specificity or sensitivity, depending on whether the intersection or union, of all variant calls is used respectively [15, 52, 53]. While this view is increasingly accepted, a tendency still exists to rely on the results from a single tool alone given the current complexity of incorporating external software into a genome analysis infrastructure. While implementing such features represents an increase in complexity and computation the results offer indisputable improvements in data quality. Such an approach is especially important in a clinical setting where there is low tolerance for false negative variants.

6 Clinical Reporting

Communicating the findings of clinical sequencing to the referring clinician/s is achieved through clinical reports, which focus primarily on these findings and their interpretation. Despite the fact that different diagnostic laboratories use distinctive reporting formats, all reports share certain features as listed below:

1. Summary of the patient's clinical information and family history.
2. The variant results, commonly presented as a table containing the identified variant/s that are linked to the given phenotype. This includes detailed description of the reported variant/s at the DNA, RNA, and protein levels.
3. The reported variants' characteristics, based on information given by population databases (e.g., gnomAD [54]), disease association databases (ClinVar [55], Human Gene Mutation Database [56], and UniProt [57]) and the functional impact of the variant as predicted by various computational tools.
4. Previously reported in vivo and/or in vitro studies involving the variant, if available.
5. Evidence linking pathogenic variants in the gene of interest with the patient's phenotype, accompanied by a discussion about the degree of overlap between the reported phenotype/s and observed one.
6. Variant classification as per the guidance of ACMG-AMP, including any special considerations/modifications due the nature of the condition.
7. References to publications cited in the report

6.1 External Data Sources

Variant-specific data embedded in clinical reports typically come from the same data sources used throughout the filtration process. These sources can be classified either into databases containing disease, sequence, and population information, or into software (Table 4).

Gene–disease databases are obligatory sources for any clinical report as they provide verifiable evidence of links between human genes and human disorders. The most important of these databases is OMIM [58], which focuses on the relationship between variation in human genes and genetic disorders at the molecular level. Similar databases include Orphanet, The Monarch Initiative [59], The Phenomizer [60], and Genomics England PanelApp. These databases are similar in that they do not attempt to exhaustively discuss all the variants in every single gene, as is the case with clinical databases such as ClinVar [55], Human Gene Mutation Database [61] and DECIPHER [62]. Additionally, there are locus, gene, and disease-specific databases such as IARC TP53 [63], Infevers (registry of Hereditary Auto-inflammatory Disorders Mutations) [64], and locus-specific databases built in the LOVD system [65]. While such resources are invaluable, there are limitations as most interpretations are formulated by expert opinions based on evaluation of often incomplete functional evidence. A review of annotations for recessive disease-causing genes found 27% percent of mutations cited in the literature were incorrect, and were identified as common polymorphisms or misannotated in public databases [66]. A follow up study of literature revealed even worse numbers with only 7.5% of 239 unique variants annotated as disease-causing in HGMD found to fit the definition [67].

Other database such as dbSNP [68], dbVar [69], and Database of Genomic Variants [70] aim to catalog all population level variation regardless of their clinical importance. Population level variant databases are indispensable for assessing the pathogenicity of variants, examples include the genome aggregation database [54], 1000 Genomes Project Phase 3 [71] and Exome Variant Server. Additionally, aggregation-style databases exist with the aim of providing a comprehensive review of the variant, as is the case of VarSome [72] or MARRVEL [73]. These comprehensive databases are beneficial is that they follow HGVS-approved nomenclature for the variant in DNA, RNA, and protein sequences, which can be confirmed using the rules published by the Human Genome Variation Society. This is important as only HGNC-approved gene names should be used throughout the entire process of sequence alignment, variant annotation, and prioritization. To ensure standardization of names, use the "Multi-symbol checker" tool from HGNC (HUGO Gene Nomenclature Committee) [56].

When consulting any of these databases, it is important to be aware of the time lag between the appearance of reportable findings in peer-reviewed publications and their subsequent inclusion in the

Table 4
List of databases, resources and software that are commonly used during variant prioritization subsequent to clinical sequencing

Database	Link
OMIM (Online Mendelian Inheritance in Man) [58]	https://www.omim.org
Orphanet	https://www.orpha.net
The Monarch Initiative [59]	https://monarchinitiative.org
The Phenomizer [60]	http://compbio.charite.de/phenomizer
Genomics England PanelApp	https://panelapp.genomicsengland.co.uk
ClinVar [55]	https://www.ncbi.nlm.nih.gov/clinvar
Human Gene Mutation Database (HGMD) [61]	http://www.hgmd.cf.ac.uk
DECIPHER [62]	http://decipher.sanger.ac.uk
Infevers (The registry of Hereditary Auto-inflammatory Disorders Mutations) [64]	https://infevers.umai-montpellier.fr
LOVD [65]	http://www.lovd.nl
dbSNP [68]	http://www.ncbi.nlm.nih.gov/snp
dbVar [69]	http://www.ncbi.nlm.nih.gov/dbvar
Database of Genomic Variants [70]	http://dgv.tcag.ca/dgv/app/home
gnomAD (Genome aggregation database) [54]	https://gnomad.broadinstitute.org
1000 Genomes Project Phase 3 [71]	http://phase3browser.1000genomes.org
Exome Variant Server	http://evs.gs.washington.edu/EVS
VarSome [72]	https://varsome.com
MARRVEL [73]	http://marrvel.org
Human Genome Variation Society	https://varnomen.hgvs.org
Multi-symbol checker [56]	https://www.genenames.org/tools/multi-symbol-checker
HGNC (HUGO Gene Nomenclature Committee) [56]	https://www.genenames.org
PubMed	https://ncbi.nlm.nih.gov/pubmed
KEGG PATHWAY Database [106]	https://www.genome.jp/kegg/pathway.html
GTEx (The Genotype-Tissue Expression project) [81]	https://gtexportal.org
Variant Effect Predictor [31]	https://ensembl.org/info/docs/tools/vep/index.html
AnnoVar [32]	http://annovar.openbioinformatics.org/en/latest/
VarAFT [107]	https://varaft.eu/

(continued)

Table 4
(continued)

Database	Link
dbNFSP [108]	https://sites.google.com/site/jpopgen/dbNSFP
snpEff [109]	http://snpeff.sourceforge.net/
Polyphen2 [79]	http://genetics.bwh.harvard.edu/pph2/
SIFT [80]	https://sift.bii.a-star.edu.sg/
CADD [45]	https://cadd.gs.washington.edu/

various databases. The lag is often relatively long, even for critically important clinical data, making it essential to comb the relevant literature on PubMed in order to guarantee that the clinical report incorporates the most up-to-date information.

6.2 Variant Classification

Molecular diagnosis depends on assessing available evidence in order to reach a verdict regarding the pathogenicity of the variant (s) remaining following prioritization. In practice, the framework for assessing pathogenicity relies on the recommendations for interpreting sequence variants published by the American College of Medical Genetics and Genomics and the Association for Molecular Pathology (ACMG/AMP) in 2015 [74]. This initial report led to significant improvements in variant classification consistency with the subsequent release offering further guidelines on avoiding overstating the significance of certain evidence types and introducing additional classification criteria [75]. In any clinical report, different types of evidence-based criteria are included such as population frequencies, segregation patterns, predictive algorithms, and functional studies, among others. Combining these criteria is performed according to a set of guidelines detailed in the same ACMG/AMP publication, a process which results in one of five variant classifications being assigned (pathogenic, likely pathogenic, variant of unknown significance (VUS), likely benign, or benign). When the evidence is insufficient or conflicting, the variant is classified as VUS; however, the increasing number of such cases is becoming a real challenge for both clinicians and patients. Importantly, the ambiguity of these variants of unknown significance is not static, as they can be revisited and reclassified using newly emerging evidence. Unfortunately, the rate of resolving such variants is much slower than the rate of their accumulation with the advent of massively parallel sequencing meaning the number of variants classified as VUS continues to grow. An additional consideration when including a variant in a clinical report is that prior to discussing evidence for and/or against pathogenicity of any

probable causal variant uncovered by sequencing, it is widely accepted that candidate causal variants must be validated by Sanger dideoxy terminator sequencing. This confirmation is critical as we are increasingly observing recurrent false positive variants appearing in population level variant databases [76].

Implementation of the aforementioned ACMG/AMP guidelines is most suited to inherited monogenic disorders with high penetrance. As we move along the continuum between strictly "Mendelian" conditions and multifactorial ones it becomes very challenging to follow these guidelines given the lack of solid borders between the two categories. Even more challenging is when a variant is identified in a gene belonging to a cellular pathway involved in the pathophysiology of the condition; however, there is no clinically established link. Overall, it is generally not recommended that the 2015 ACMG/AMP guidelines are applied to such cases, nor to variants found in asymptomatic or healthy individuals.

6.3 Secondary Findings

Standardization of clinical sequencing by ACMG was not limited to variants related to the primary purpose of testing, it was also extended to "secondary findings." In fact, ACMG published a list of 59 genes that are linked to a number of disorders which fulfil certain criteria such as having a reliable clinical genetic test for early diagnosis and the possibility of effective intervention [77]. Known pathogenic (and sometimes expected pathogenic) variants in these genes are reported—upon consent—as secondary findings depending on the internal policy of the diagnostic laboratory. Reportable secondary findings contain similar information to cases with primary findings; however, reportable secondary findings depend primarily on previously published reports, and to a lesser degree on predictive algorithms.

6.4 Strategy of Variant Prioritization

Following clinical sequencing, several different lines of variant filtration are applied to each of the four types of variants: SNVs, indels, repeats, and structural variants. Additional filters are also applied when analyzing variants uncovered as part of a cohort compared to variants from a singleton sample. Generally, the first step in SNVs prioritization is to exclude noncoding, synonymous, and "common" population variants from the total pool of SNVs from the affected/s. Minor allele frequencies (MAF) are obtained from databases such as gnomAD [54] or the Exome Sequencing Project (ESP), and variants with MAF above the threshold of 1% are considered common. In some instances, collective exclusion of variants based on absolute MAF cut-offs may remove pathogenic variants that are more prevalent than expected (as in cases of the founder effect). Similarly, mass exclusion of all synonymous variants ignores the effects that these variants may have on RNA level (e.g., altered splicing). It is therefore important to go through the literature about the condition under study in search for such exceptions

prior to such mass exclusions. In any case, applying these filters remove around 95% of the initial list of SNVs with the remaining variants directed through two further lines of interrogation; Custom Gene Lists (CGL) and multiple in silico prediction tools.

CGLs are essentially built using all available gene–disease associations at the time of analysis; bringing together all genes with known connections relevant to a certain condition. Data from OMIM [58], Orphanet, The Monarch Initiative [59], and PanelApp are first combined and then supplemented with up-to-date information from the latest peer-reviewed articles and case reports. Increasingly, gene lists for many monogenic conditions are maintained by genetic testing companies such as GeneDx, Invitae, Fulgent, CENTOGENE, and many others. Such lists offer another potential source of information to ensure CGLs are comprehensive. Specialized tools for obtaining genes linked to a certain phenotype, such as The Phenomizer [60], can also be of help. Finally, using HGNC-approved gene names is essential for any CGL list, which can be achieved using the *Multi-Symbol Checker* from HGNC [56]. This process generates phenotype-specific CGL lists so when a matching phenotype is encountered, all nonsynonymous and rare variants in CGL list genes are prioritized. The number of genes containing such variants is usually 20–30 times less than the number of genes in the CGL resulting in a much smaller genetic search space. This generates a smaller list of variants suitable for detailed manual interrogation where the characteristics of the individual variant and gene are examined. For the majority of these variants, exclusion from being causal for the phenotype can be easily done based on simple observations. For instance, a variant in a gene linked to a congenital condition with dominant mode of inheritance (and full penetrance) can be quickly excluded upon noting the existence of hundreds of healthy heterozygotes with this variant in population databases. Similarly, incompatibility in terms of an affected patients zygosity and/or detailed phenotype with expected mode of inheritance and phenotype of the mutations in the gene under analysis causes exclusions of an additional set of variants. Analyzing cohorts (as opposed to singletons) uncovers segregation patterns which offer a huge advantage to effectively filter out variants that do not segregate with the phenotype, for example variants for which healthy family members are homozygous. Conversely, segregation data can be used to prioritize variants that appear de novo in the affected, especially for the numerous conditions that are known to result from these types of variants. Variants that survive these rounds of detailed assessments are discussed in meetings with molecular pathologists who evaluate existing evidence for each of the candidate variants until a clear verdict is reached.

Indels found in genes are analyzed similarly to SNVs and focus on identifying out of frame variants. Structural variants are filtered differently, however, on the basis of the variant's length and the number of paired reads supporting it. Structural variants that have less than 4 paired reads supporting them or exceed 200 bp in length are excluded, the remaining variants are intersected with genes in CGL. Resulting genes/variants are examined using bam files uploaded into Integrated Genomic Viewer (IGV) software [78]. Comparisons can be made between affected and unaffected individuals as well as reference individuals.

To complement the CGL-based approach, a second filtration strategy is applied to SNVs and indels. This strategy uses computational prediction tools of pathogenicity as a starting point, which allows for variants with pathogenic in silico scores to be highlighted and analyzed regardless of their previous inclusion in routinely constructed gene lists for the condition. A consensus approach is employed using three tools CADD [45], PolyPhen2 [79], and SIFT [80], with variants classified as "benign" excluded. Nonbenign variants are then examined for gene–phenotype links relevant to patient condition. This information can be obtained from a variety of databases which reports links such as phenotypic descriptions in model organisms (e.g., OMIM [58]) or expression profiles in various tissues (e.g., GTeX [81]). Typically, the majority of variants identified using this strategy are located in genes with "novel" links to human condition/s, making it is challenging to apply ACMG/AMP criteria for classification. Nevertheless, reporting such variants (including VUS) is critical as these reports can be shared with other diagnostic labs, and consequently upgraded (or downgraded) in terms of classification depending on emerging data from other patients. Variants resulting from prioritization based on in silico prediction tools are also scrutinized in terms of other evidence of pathogenicity such as the case with variants uncovered using the CGL-based strategy.

The known difficulty of applying ACMG/AMP guidelines uniformly to all types of conditions described as "Mendelian" has instigated the amendment of certain criteria and the extension of other criteria. Despite this, numerous discrepancies continue to surface due to variables such as underlying cause of disorder (loss of function or gain of function mutations), level of penetrance/expressivity for the disorder, disease prevalence, and prominence of certain types of mutations underlying the condition's pathophysiology. This has led to the formation of ClinGen Sequence Variant Interpretation Working Groups, where each group focuses on a specific disorder and issues modified ACMG/AMP guidelines for that particular set of conditions [82] or even for particular genes such as *MYH7* and *MEN1* [83, 84]. The emergence of disorder-specific guidelines will greatly improve consistency in usage and transparency in classification rationale.

References

1. Taupin D, Lam W, Rangiah D, McCallum L, Whittle B, Zhang Y, Andrews D, Field M, Goodnow CC, Cook MC (2015) A deleterious RNF43 germline mutation in a severely affected serrated polyposis kindred. Hum Genome Var 2:15013
2. Dunkerton S, Field M, Cho V, Bertram E, Whittle B, Groves A, Goel H (2015) A de novo mutation in KMT2A (MLL) in monozygotic twins with Wiedemann-Steiner syndrome. Am J Med Genet A 167A(9):2182-2187. https://doi.org/10.1002/ajmg.a.37130
3. van Dijk EL, Auger H, Jaszczyszyn Y, Thermes C (2014) Ten years of next-generation sequencing technology. Trends Genet 30(9):418–426. https://doi.org/10.1016/j.tig.2014.07.001
4. Johar AS, Mastronardi C, Rojas-Villarraga A, Patel HR, Chuah A, Peng K, Higgins A, Milburn P, Palmer S, Silva-Lara MF, Velez JI, Andrews D, Field M, Huttley G, Goodnow C, Anaya JM, Arcos-Burgos M (2015) Novel and rare functional genomic variants in multiple autoimmune syndrome and Sjogren's syndrome. J Transl Med 13:173. https://doi.org/10.1186/s12967-015-0525-x
5. Pabinger S, Dander A, Fischer M, Snajder R, Sperk M, Efremova M, Krabichler B, Speicher MR, Zschocke J, Trajanoski Z (2014) A survey of tools for variant analysis of next-generation genome sequencing data. Brief Bioinform 15(2):256–278. https://doi.org/10.1093/bib/bbs086
6. Bolger AM, Lohse M, Usadel B (2014) Trimmomatic: a flexible trimmer for Illumina sequence data. Bioinformatics 30(15):2114–2120. https://doi.org/10.1093/bioinformatics/btu170
7. Patel RK, Jain M (2012) NGS QC Toolkit: a toolkit for quality control of next generation sequencing data. PLoS One 7(2):e30619. https://doi.org/10.1371/journal.pone.0030619
8. Li H, Durbin R (2009) Fast and accurate short read alignment with Burrows-Wheeler transform. Bioinformatics 25(14):1754–1760. https://doi.org/10.1093/bioinformatics/btp324
9. Langmead B, Salzberg SL (2012) Fast gapped-read alignment with Bowtie 2. Nat Methods 9(4):357–359. https://doi.org/10.1038/nmeth.1923
10. Dobin A, Davis CA, Schlesinger F, Drenkow J, Zaleski C, Jha S, Batut P, Chaisson M, Gingeras TR (2013) STAR: ultrafast universal RNA-seq aligner. Bioinformatics 29(1):15–21. https://doi.org/10.1093/bioinformatics/bts635
11. Kim D, Pertea G, Trapnell C, Pimentel H, Kelley R, Salzberg SL (2013) TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions. Genome Biol 14(4):R36. https://doi.org/10.1186/gb-2013-14-4-r36
12. McKenna A, Hanna M, Banks E, Sivachenko A, Cibulskis K, Kernytsky A, Garimella K, Altshuler D, Gabriel S, Daly M, DePristo MA (2010) The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. Genome Res 20(9):1297–1303. https://doi.org/10.1101/gr.107524.110
13. Lappalainen T, Scott AJ, Brandt M, Hall IM (2019) Genomic analysis in the age of human genome sequencing. Cell 177(1):70–84. https://doi.org/10.1016/j.cell.2019.02.032
14. Cantarel BL, Weaver D, McNeill N, Zhang J, Mackey AJ, Reese J (2014) BAYSIC: a Bayesian method for combining sets of genome variants with improved specificity and sensitivity. BMC Bioinformatics 15:104. https://doi.org/10.1186/1471-2105-15-104
15. Sandmann S, Karimi M, de Graaf AO, Rohde C, Gollner S, Varghese J, Ernsting J, Walldin G, van der Reijden BA, Muller-Tidow C, Malcovati L, Hellstrom-Lindberg E, Jansen JH, Dugas M (2018) appreci8: a pipeline for precise variant calling integrating 8 tools. Bioinformatics 34(24):4205–4212. https://doi.org/10.1093/bioinformatics/bty518
16. Andrews TD, Jeelall Y, Talaulikar D, Goodnow CC, Field MA (2016) DeepSNVMiner: a sequence analysis tool to detect emergent, rare mutations in subsets of cell populations. PeerJ 4:e2074. https://doi.org/10.7717/peerj.2074
17. Xu C, Gu X, Padmanabhan R, Wu Z, Peng Q, DiCarlo J, Wang Y (2019) smCounter2: an accurate low-frequency variant caller for targeted sequencing data with unique molecular identifiers. Bioinformatics 35(8):1299–1309. https://doi.org/10.1093/bioinformatics/bty790
18. Chen J, Li X, Zhong H, Meng Y, Du H (2019) Systematic comparison of germline variant calling pipelines cross multiple next-generation sequencers. Sci Rep 9(1):9345. https://doi.org/10.1038/s41598-019-45835-3

19. Abyzov A, Li S, Kim DR, Mohiyuddin M, Stutz AM, Parrish NF, Mu XJ, Clark W, Chen K, Hurles M, Korbel JO, Lam HY, Lee C, Gerstein MB (2015) Analysis of deletion breakpoints from 1,092 humans reveals details of mutation mechanisms. Nat Commun 6:7256. https://doi.org/10.1038/ncomms8256
20. Monlong J, Cossette P, Meloche C, Rouleau G, Girard SL, Bourque G (2018) Human copy number variants are enriched in regions of low mappability. Nucleic Acids Res 46(14):7236–7249. https://doi.org/10.1093/nar/gky538
21. Abyzov A, Urban AE, Snyder M, Gerstein M (2011) CNVnator: an approach to discover, genotype, and characterize typical and atypical CNVs from family and population genome sequencing. Genome Res 21(6):974–984. https://doi.org/10.1101/gr.114876.110
22. Cameron DL, Schroder J, Penington JS, Do H, Molania R, Dobrovic A, Speed TP, Papenfuss AT (2017) GRIDSS: sensitive and specific genomic rearrangement detection using positional de Bruijn graph assembly. Genome Res 27(12):2050–2060. https://doi.org/10.1101/gr.222109.117
23. Quinlan AR, Clark RA, Sokolova S, Leibowitz ML, Zhang Y, Hurles ME, Mell JC, Hall IM (2010) Genome-wide mapping and assembly of structural variant breakpoints in the mouse genome. Genome Res 20(5):623–635. https://doi.org/10.1101/gr.102970.109
24. Wang J, Mullighan CG, Easton J, Roberts S, Heatley SL, Ma J, Rusch MC, Chen K, Harris CC, Ding L, Holmfeldt L, Payne-Turner D, Fan X, Wei L, Zhao D, Obenauer JC, Naeve C, Mardis ER, Wilson RK, Downing JR, Zhang J (2011) CREST maps somatic structural variation in cancer genomes with base-pair resolution. Nat Methods 8(8):652–654. https://doi.org/10.1038/nmeth.1628
25. Kosugi S, Momozawa Y, Liu X, Terao C, Kubo M, Kamatani Y (2019) Comprehensive evaluation of structural variation detection algorithms for whole genome sequencing. Genome Biol 20(1):117. https://doi.org/10.1186/s13059-019-1720-5
26. Sudmant PH, Rausch T, Gardner EJ, Handsaker RE, Abyzov A, Huddleston J, Zhang Y, Ye K, Jun G, Fritz MH, Konkel MK, Malhotra A, Stutz AM, Shi X, Casale FP, Chen J, Hormozdiari F, Dayama G, Chen K, Malig M, Chaisson MJP, Walter K, Meiers S, Kashin S, Garrison E, Auton A, Lam HYK, Mu XJ, Alkan C, Antaki D, Bae T, Cerveira E, Chines P, Chong Z, Clarke L, Dal E, Ding L, Emery S, Fan X, Gujral M, Kahveci F, Kidd JM, Kong Y, Lameijer EW, McCarthy S, Flicek P, Gibbs RA, Marth G, Mason CE, Menelaou A, Muzny DM, Nelson BJ, Noor A, Parrish NF, Pendleton M, Quitadamo A, Raeder B, Schadt EE, Romanovitch M, Schlattl A, Sebra R, Shabalin AA, Untergasser A, Walker JA, Wang M, Yu F, Zhang C, Zhang J, Zheng-Bradley X, Zhou W, Zichner T, Sebat J, Batzer MA, McCarroll SA, Genomes Project C, Mills RE, Gerstein MB, Bashir A, Stegle O, Devine SE, Lee C, Eichler EE, Korbel JO (2015) An integrated map of structural variation in 2,504 human genomes. Nature 526(7571):75–81. https://doi.org/10.1038/nature15394
27. Sudmant PH, Mallick S, Nelson BJ, Hormozdiari F, Krumm N, Huddleston J, Coe BP, Baker C, Nordenfelt S, Bamshad M, Jorde LB, Posukh OL, Sahakyan H, Watkins WS, Yepiskoposyan L, Abdullah MS, Bravi CM, Capelli C, Hervig T, Wee JT, Tyler-Smith C, van Driem G, Romero IG, Jha AR, Karachanak-Yankova S, Toncheva D, Comas D, Henn B, Kivisild T, Ruiz-Linares A, Sajantila A, Metspalu E, Parik J, Villems R, Starikovskaya EB, Ayodo G, Beall CM, Di Rienzo A, Hammer MF, Khusainova R, Khusnutdinova E, Klitz W, Winkler C, Labuda D, Metspalu M, Tishkoff SA, Dryomov S, Sukernik R, Patterson N, Reich D, Eichler EE (2015) Global diversity, population stratification, and selection of human copy-number variation. Science 349(6253):aab3761. https://doi.org/10.1126/science.aab3761
28. Novak P, Neumann P, Pech J, Steinhaisl J, Macas J (2013) RepeatExplorer: a Galaxy-based web server for genome-wide characterization of eukaryotic repetitive elements from next-generation sequence reads. Bioinformatics 29(6):792–793. https://doi.org/10.1093/bioinformatics/btt054
29. Dashnow H, Lek M, Phipson B, Halman A, Sadedin S, Lonsdale A, Davis M, Lamont P, Clayton JS, Laing NG, MacArthur DG, Oshlack A (2018) STRetch: detecting and discovering pathogenic short tandem repeat expansions. Genome Biol 19(1):121. https://doi.org/10.1186/s13059-018-1505-2
30. Pruitt KD, Maglott DR (2001) RefSeq and LocusLink: NCBI gene-centered resources. Nucleic Acids Res 29(1):137–140. https://doi.org/10.1093/nar/29.1.137
31. McLaren W, Pritchard B, Rios D, Chen Y, Flicek P, Cunningham F (2010) Deriving the

consequences of genomic variants with the Ensembl API and SNP Effect Predictor. Bioinformatics 26(16):2069–2070. https://doi.org/10.1093/bioinformatics/btq330

32. Wang K, Li M, Hakonarson H (2010) ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data. Nucleic Acids Res 38(16):e164. https://doi.org/10.1093/nar/gkq603
33. Anna A, Monika G (2018) Splicing mutations in human genetic disorders: examples, detection, and confirmation. J Appl Genet 59 (3):253–268. https://doi.org/10.1007/s13353-018-0444-7
34. Miosge LA, Field MA, Sontani Y, Cho V, Johnson S, Palkova A, Balakishnan B, Liang R, Zhang Y, Lyon S, Beutler B, Whittle B, Bertram EM, Enders A, Goodnow CC, Andrews TD (2015) Comparison of predicted and actual consequences of missense mutations. Proc Natl Acad Sci USA 112(37): E5189-E5198. https://doi.org/10.1073/pnas.1511585112
35. Johar AS, Anaya JM, Andrews D, Patel HR, Field M, Goodnow C, Arcos-Burgos M (2014) Candidate gene discovery in autoimmunity by using extreme phenotypes, next generation sequencing and whole exome capture. Autoimmun Rev 14(3):204-209. https://doi.org/10.1016/j.autrev.2014.10.021
36. Al Sukaiti N, AbdelRahman K, AlShekaili J, Al Oraimi S, Al Sinani A, Al Rahbi N, Cho V, Field M, Cook MC (2017) Agammaglobulinaemia despite terminal B-cell differentiation in a patient with a novel LRBA mutation. Clin Transl Immunol 6(5):e144
37. Cardinez C, Miraghazadeh B, Tanita K, da Silva E, Hoshino A, Okada S, Chand R, Asano T, Tsumura M, Yoshida K, Ohnishi H, Kato Z, Yamazaki M, Okuno Y, Miyano S, Kojima S, Ogawa S, Andrews TD, Field MA, Burgio G, Morio T, Vinuesa CG, Kanegane H, Cook MC (2018) Gain-of-function IKBKB mutation causes human combined immune deficiency. J Exp Med. https://doi.org/10.1084/jem.20180639
38. Jiang SH, Athanasopoulos V, Ellyard JI, Chuah A, Cappello J, Cook A, Prabhu SB, Cardenas J, Gu J, Stanley M, Roco JA, Papa I, Yabas M, Walters GD, Burgio G, McKeon K, Byers JM, Burrin C, Enders A, Miosge LA, Canete PF, Jelusic M, Tasic V, Lungu AC, Alexander SI, Kitching AR, Fulcher DA, Shen N, Arsov T, Gatenby PA, Babon JJ, Mallon DF, de Lucas CC, Stone EA, Wu P, Field MA, Andrews TD, Cho E, Pascual V, Cook MC, Vinuesa CG (2019) Functional rare and low frequency variants in BLK and BANK1 contribute to human lupus. Nat Commun 10(1):2201. https://doi.org/10.1038/s41467-019-10242-9
39. Kramer A, Green J, Pollard J Jr, Tugendreich S (2014) Causal analysis approaches in Ingenuity Pathway Analysis. Bioinformatics 30 (4):523–530. https://doi.org/10.1093/bioinformatics/btt703
40. Szklarczyk D, Morris JH, Cook H, Kuhn M, Wyder S, Simonovic M, Santos A, Doncheva NT, Roth A, Bork P, Jensen LJ, von Mering C (2017) The STRING database in 2017: quality-controlled protein-protein association networks, made broadly accessible. Nucleic Acids Res 45(D1):D362–D368. https://doi.org/10.1093/nar/gkw937
41. Ng SB, Buckingham KJ, Lee C, Bigham AW, Tabor HK, Dent KM, Huff CD, Shannon PT, Jabs EW, Nickerson DA, Shendure J, Bamshad MJ (2010) Exome sequencing identifies the cause of a mendelian disorder. Nat Genet 42(1):30–35. https://doi.org/10.1038/ng.499
42. Yang Y, Muzny DM, Reid JG, Bainbridge MN, Willis A, Ward PA, Braxton A, Beuten J, Xia F, Niu Z, Hardison M, Person R, Bekheirnia MR, Leduc MS, Kirby A, Pham P, Scull J, Wang M, Ding Y, Plon SE, Lupski JR, Beaudet AL, Gibbs RA, Eng CM (2013) Clinical whole-exome sequencing for the diagnosis of mendelian disorders. N Engl J Med 369 (16):1502–1511. https://doi.org/10.1056/NEJMoa1306555
43. Li MX, Gui HS, Kwan JS, Bao SY, Sham PC (2012) A comprehensive framework for prioritizing variants in exome sequencing studies of Mendelian diseases. Nucleic Acids Res 40(7): e53. https://doi.org/10.1093/nar/gkr1257
44. Kamphans T, Sabri P, Zhu N, Heinrich V, Mundlos S, Robinson PN, Parkhomchuk D, Krawitz PM (2013) Filtering for compound heterozygous sequence variants in non-consanguineous pedigrees. PLoS One 8 (8):e70151. https://doi.org/10.1371/journal.pone.0070151
45. Kircher M, Witten DM, Jain P, O'Roak BJ, Cooper GM, Shendure J (2014) A general framework for estimating the relative pathogenicity of human genetic variants. Nat Genet 46(3):310–315. https://doi.org/10.1038/ng.2892
46. Paila U, Chapman BA, Kirchner R, Quinlan AR (2013) GEMINI: integrative exploration of genetic variation and genome annotations.

PLoS Comput Biol 9(7):e1003153. https://doi.org/10.1371/journal.pcbi.1003153

47. Field MA, Cho V, Cook MC, Enders A, Vinuesa C, Whittle B, Andrews TD, Goodnow CC (2015) Reducing the search space for causal genetic variants with VASP: Variant Analysis of Sequenced Pedigrees. Bioinformatics 31(14):2377-2379. https://doi.org/10.1093/bioinformatics/btv135
48. Schmitt MW, Kennedy SR, Salk JJ, Fox EJ, Hiatt JB, Loeb LA (2012) Detection of ultra-rare mutations by next-generation sequencing. Proc Natl Acad Sci U S A 109 (36):14508–14513. https://doi.org/10.1073/pnas.1208715109
49. Cummings BB, Marshall JL, Tukiainen T, Lek M, Donkervoort S, Foley AR, Bolduc V, Waddell LB, Sandaradura SA, O'Grady GL, Estrella E, Reddy HM, Zhao F, Weisburd B, Karczewski KJ, O'Donnell-Luria AH, Birnbaum D, Sarkozy A, Hu Y, Gonorazky H, Claeys K, Joshi H, Bournazos A, Oates EC, Ghaoui R, Davis MR, Laing NG, Topf A, Genotype-Tissue Expression C, Kang PB, Beggs AH, North KN, Straub V, Dowling JJ, Muntoni F, Clarke NF, Cooper ST, Bonnemann CG, MacArthur DG (2017) Improving genetic diagnosis in Mendelian disease with transcriptome sequencing. Sci Transl Med 9(386): eaal5209. https://doi.org/10.1126/scitranslmed.aal5209
50. Byron SA, Van Keuren-Jensen KR, Engelthaler DM, Carpten JD, Craig DW (2016) Translating RNA sequencing into clinical diagnostics: opportunities and challenges. Nat Rev Genet 17(5):257–271. https://doi.org/10.1038/nrg.2016.10
51. Merker JD, Wenger AM, Sneddon T, Grove M, Zappala Z, Fresard L, Waggott D, Utiramerur S, Hou Y, Smith KS, Montgomery SB, Wheeler M, Buchan JG, Lambert CC, Eng KS, Hickey L, Korlach J, Ford J, Ashley EA (2018) Long-read genome sequencing identifies causal structural variation in a Mendelian disease. Genet Med 20(1):159–163. https://doi.org/10.1038/gim.2017.86
52. Field MA, Cho V, Andrews TD, Goodnow CC (2015) Reliably detecting clinically important variants requires both combined variant calls and optimized filtering strategies. PLoS One 10(11):e0143199. https://doi.org/10.1371/journal.pone.0143199
53. Waardenberg AJ, Field MA (2019). consensusDE: an R package for assessing consensus of multiple RNA-seq algorithms with RUV correction. PeerJ 7:e8206. https://doi.org/10.7717/peerj.8206
54. Karczewski KJ, Francioli LC, Tiao G, Cummings BB, Alföldi J, Wang Q, Collins RL, Laricchia KM, Ganna A, Birnbaum DP, Gauthier LD, Brand H, Solomonson M, Watts NA, Rhodes D, Singer-Berk M, Seaby EG, Kosmicki JA, Walters RK, Tashman K, Farjoun Y, Banks E, Poterba T, Wang A, Seed C, Whiffin N, Chong JX, Samocha KE, Pierce-Hoffman E, Zappala Z, O'Donnell-Luria AH, Vallabh Minikel E, Weisburd B, Lek M, Ware JS, Vittal C, Armean IM, Bergelson L, Cibulskis K, Connolly KM, Covarrubias M, Donnelly S, Ferriera S, Gabriel S, Gentry J, Gupta N, Jeandet T, Kaplan D, Llanwarne C, Munshi R, Novod S, Petrillo N, Roazen D, Ruano-Rubio V, Saltzman A, Schleicher M, Soto J, Tibbetts K, Tolonen C, Wade G, Talkowski ME, Neale BM, Daly MJ, MacArthur DG (2019) Variation across 141,456 human exomes and genomes reveals the spectrum of loss-of-function intolerance across human protein-coding genes. bioRxiv:531210. https://doi.org/10.1101/531210
55. Landrum MJ, Lee JM, Riley GR, Jang W, Rubinstein WS, Church DM, Maglott DR (2014) ClinVar: public archive of relationships among sequence variation and human phenotype. Nucleic Acids Res 42(Database issue):D980–D985. https://doi.org/10.1093/nar/gkt1113
56. Braschi B, Denny P, Gray K, Jones T, Seal R, Tweedie S, Yates B, Bruford E (2019) Genenames.org: the HGNC and VGNC resources in 2019. Nucleic Acids Res 47(D1): D786–D792. https://doi.org/10.1093/nar/gky930
57. UniProt C (2008) The universal protein resource (UniProt). Nucleic Acids Res 36 (Database issue):D190–D195. https://doi.org/10.1093/nar/gkm895
58. Hamosh A, Scott AF, Amberger JS, Bocchini CA, McKusick VA (2005) Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders. Nucleic Acids Res 33(Database issue): D514–D517. https://doi.org/10.1093/nar/gki033
59. Mungall CJ, McMurry JA, Kohler S, Balhoff JP, Borromeo C, Brush M, Carbon S, Conlin T, Dunn N, Engelstad M, Foster E, Gourdine JP, Jacobsen JO, Keith D, Laraway B, Lewis SE, NguyenXuan J, Shefchek K, Vasilevsky N, Yuan Z, Washington N, Hochheiser H, Groza T, Smedley D, Robinson PN, Haendel MA (2017) The Monarch initiative: an integrative data and analytic platform connecting

phenotypes to genotypes across species. Nucleic Acids Res 45(D1):D712–D722. https://doi.org/10.1093/nar/gkw1128

60. Kohler S, Schulz MH, Krawitz P, Bauer S, Dolken S, Ott CE, Mundlos C, Horn D, Mundlos S, Robinson PN (2009) Clinical diagnostics in human genetics with semantic similarity searches in ontologies. Am J Hum Genet 85(4):457–464. https://doi.org/10.1016/j.ajhg.2009.09.003
61. Stenson PD, Mort M, Ball EV, Evans K, Hayden M, Heywood S, Hussain M, Phillips AD, Cooper DN (2017) The Human Gene Mutation Database: towards a comprehensive repository of inherited mutation data for medical research, genetic diagnosis and next-generation sequencing studies. Hum Genet 136(6):665–677. https://doi.org/10.1007/s00439-017-1779-6
62. Bragin E, Chatzimichali EA, Wright CF, Hurles ME, Firth HV, Bevan AP, Swaminathan GJ (2014) DECIPHER: database for the interpretation of phenotype-linked plausibly pathogenic sequence and copy-number variation. Nucleic Acids Res 42(Database issue):D993–D1000. https://doi.org/10.1093/nar/gkt937
63. Olivier M, Eeles R, Hollstein M, Khan MA, Harris CC, Hainaut P (2002) The IARC TP53 database: new online mutation analysis and recommendations to users. Hum Mutat 19(6):607–614. https://doi.org/10.1002/humu.10081
64. Sarrauste de Menthiere C, Terriere S, Pugnere D, Ruiz M, Demaille J, Touitou I (2003) INFEVERS: the Registry for FMF and hereditary inflammatory disorders mutations. Nucleic Acids Res 31(1):282–285. https://doi.org/10.1093/nar/gkg031
65. Fokkema IF, Taschner PE, Schaafsma GC, Celli J, Laros JF, den Dunnen JT (2011) LOVD v.2.0: the next generation in gene variant databases. Hum Mutat 32(5):557–563. https://doi.org/10.1002/humu.21438
66. Bell CJ, Dinwiddie DL, Miller NA, Hateley SL, Ganusova EE, Mudge J, Langley RJ, Zhang L, Lee CC, Schilkey FD, Sheth V, Woodward JE, Peckham HE, Schroth GP, Kim RW, Kingsmore SF (2011) Carrier testing for severe childhood recessive diseases by next-generation sequencing. Sci Transl Med 3(65):65ra64. https://doi.org/10.1126/scitranslmed.3001756
67. Dorschner MO, Amendola LM, Turner EH, Robertson PD, Shirts BH, Gallego CJ, Bennett RL, Jones KL, Tokita MJ, Bennett JT, Kim JH, Rosenthal EA, Kim DS, National Heart L, Blood Institute Grand Opportunity Exome Sequencing P, Tabor HK, Bamshad MJ, Motulsky AG, Scott CR, Pritchard CC, Walsh T, Burke W, Raskind WH, Byers P, Hisama FM, Nickerson DA, Jarvik GP (2013) Actionable, pathogenic incidental findings in 1,000 participants' exomes. Am J Hum Genet 93(4):631–640. https://doi.org/10.1016/j.ajhg.2013.08.006
68. Sherry ST, Ward MH, Kholodov M, Baker J, Phan L, Smigielski EM, Sirotkin K (2001) dbSNP: the NCBI database of genetic variation. Nucleic Acids Res 29(1):308–311
69. Lappalainen I, Lopez J, Skipper L, Hefferon T, Spalding JD, Garner J, Chen C, Maguire M, Corbett M, Zhou G, Paschall J, Ananiev V, Flicek P, Church DM (2013) DbVar and DGVa: public archives for genomic structural variation. Nucleic Acids Res 41(Database issue):D936–D941. https://doi.org/10.1093/nar/gks1213
70. MacDonald JR, Ziman R, Yuen RK, Feuk L, Scherer SW (2014) The Database of Genomic Variants: a curated collection of structural variation in the human genome. Nucleic Acids Res 42(Database issue):D986–D992. https://doi.org/10.1093/nar/gkt958
71. Genomes Project C, Auton A, Brooks LD, Durbin RM, Garrison EP, Kang HM, Korbel JO, Marchini JL, McCarthy S, McVean GA, Abecasis GR (2015) A global reference for human genetic variation. Nature 526(7571):68–74. https://doi.org/10.1038/nature15393
72. Kopanos C, Tsiolkas V, Kouris A, Chapple CE, Albarca Aguilera M, Meyer R, Massouras A (2019) VarSome: the human genomic variant search engine. Bioinformatics 35(11):1978–1980. https://doi.org/10.1093/bioinformatics/bty897
73. Wang J, Al-Ouran R, Hu Y, Kim SY, Wan YW, Wangler MF, Yamamoto S, Chao HT, Comjean A, Mohr SE, Udn PN, Liu Z, Bellen HJ (2017) MARRVEL: integration of human and model organism genetic resources to facilitate functional annotation of the human genome. Am J Hum Genet 100(6):843–853. https://doi.org/10.1016/j.ajhg.2017.04.010
74. Richards S, Aziz N, Bale S, Bick D, Das S, Gastier-Foster J, Grody WW, Hegde M, Lyon E, Spector E, Voelkerding K, Rehm HL, Committee ALQA (2015) Standards and guidelines for the interpretation of sequence variants: a joint consensus recommendation of the American College of Medical Genetics and Genomics and the

Association for Molecular Pathology. Genet Med 17(5):405–424. https://doi.org/10.1038/gim.2015.30

75. Nykamp K, Anderson M, Powers M, Garcia J, Herrera B, Ho YY, Kobayashi Y, Patil N, Thusberg J, Westbrook M, Invitae Clinical Genomics G, Topper S (2017) Sherloc: a comprehensive refinement of the ACMG-AMP variant classification criteria. Genet Med 19(10):1105–1117. https://doi.org/10.1038/gim.2017.37
76. Field MA, Burgio G, Chuah A, Al Shekaili J, Hassan B, Al Sukaiti N, Foote SJ, Cook MC, Andrews TD (2019) Recurrent miscalling of missense variation from short-read genome sequence data. BMC Genomics 20(Suppl 8):546. https://doi.org/10.1186/s12864-019-5863-2
77. Kalia SS, Adelman K, Bale SJ, Chung WK, Eng C, Evans JP, Herman GE, Hufnagel SB, Klein TE, Korf BR, McKelvey KD, Ormond KE, Richards CS, Vlangos CN, Watson M, Martin CL, Miller DT (2017) Recommendations for reporting of secondary findings in clinical exome and genome sequencing, 2016 update (ACMG SF v2.0): a policy statement of the American College of Medical Genetics and Genomics. Genet Med 19(2):249–255. https://doi.org/10.1038/gim.2016.190
78. Robinson JT, Thorvaldsdottir H, Winckler W, Guttman M, Lander ES, Getz G, Mesirov JP (2011) Integrative genomics viewer. Nat Biotechnol 29(1):24–26. https://doi.org/10.1038/nbt.1754
79. Adzhubei IA, Schmidt S, Peshkin L, Ramensky VE, Gerasimova A, Bork P, Kondrashov AS, Sunyaev SR (2010) A method and server for predicting damaging missense mutations. Nat Methods 7(4):248–249. https://doi.org/10.1038/nmeth0410-248
80. Sim NL, Kumar P, Hu J, Henikoff S, Schneider G, Ng PC (2012) SIFT web server: predicting effects of amino acid substitutions on proteins. Nucleic Acids Res 40(Web Server issue):W452–W457. https://doi.org/10.1093/nar/gks539
81. Consortium GT (2013) The Genotype-Tissue Expression (GTEx) project. Nat Genet 45(6):580–585. https://doi.org/10.1038/ng.2653
82. Gelb BD, Cave H, Dillon MW, Gripp KW, Lee JA, Mason-Suares H, Rauen KA, Williams B, Zenker M, Vincent LM, ClinGen RWG (2018) ClinGen's RASopathy Expert Panel consensus methods for variant interpretation. Genet Med 20(11):1334–1345. https://doi.org/10.1038/gim.2018.3
83. Romanet P, Odou MF, North MO, Saveanu A, Coppin L, Pasmant E, Mohamed A, Goudet P, Borson-Chazot F, Calender A, Beroud C, Levy N, Giraud S, Barlier A (2019) Proposition of adjustments to the ACMG-AMP framework for the interpretation of MEN1 missense variants. Hum Mutat 40(6):661–674. https://doi.org/10.1002/humu.23746
84. Kelly MA, Caleshu C, Morales A, Buchan J, Wolf Z, Harrison SM, Cook S, Dillon MW, Garcia J, Haverfield E, Jongbloed JDH, Macaya D, Manrai A, Orland K, Richard G, Spoonamore K, Thomas M, Thomson K, Vincent LM, Walsh R, Watkins H, Whiffin N, Ingles J, van Tintelen JP, Semsarian C, Ware JS, Hershberger R, Funke B (2018) Adaptation and validation of the ACMG/AMP variant classification framework for MYH7-associated inherited cardiomyopathies: recommendations by ClinGen's Inherited Cardiomyopathy Expert Panel. Genet Med 20(3):351–359. https://doi.org/10.1038/gim.2017.218
85. Li H (2011) A statistical framework for SNP calling, mutation discovery, association mapping and population genetical parameter estimation from sequencing data. Bioinformatics 27(21):2987–2993. https://doi.org/10.1093/bioinformatics/btr509
86. Koboldt DC, Chen K, Wylie T, Larson DE, McLellan MD, Mardis ER, Weinstock GM, Wilson RK, Ding L (2009) VarScan: variant detection in massively parallel sequencing of individual and pooled samples. Bioinformatics 25(17):2283–2285. https://doi.org/10.1093/bioinformatics/btp373
87. Rimmer A, Phan H, Mathieson I, Iqbal Z, Twigg SRF, Consortium WGS, Wilkie AOM, McVean G, Lunter G (2014) Integrating mapping-, assembly- and haplotype-based approaches for calling variants in clinical sequencing applications. Nat Genet 46 (8):912–918. https://doi.org/10.1038/ng.3036
88. Kim S, Scheffler K, Halpern AL, Bekritsky MA, Noh E, Kallberg M, Chen X, Kim Y, Beyter D, Krusche P, Saunders CT (2018) Strelka2: fast and accurate calling of germline and somatic variants. Nat Methods 15 (8):591–594. https://doi.org/10.1038/s41592-018-0051-x
89. Lai Z, Markovets A, Ahdesmaki M, Chapman B, Hofmann O, McEwen R, Johnson J, Dougherty B, Barrett JC, Dry JR (2016) VarDict: a novel and versatile variant caller for next-generation sequencing in cancer research. Nucleic Acids Res 44(11):e108. https://doi.org/10.1093/nar/gkw227

90. Wilm A, Aw PP, Bertrand D, Yeo GH, Ong SH, Wong CH, Khor CC, Petric R, Hibberd ML, Nagarajan N (2012) LoFreq: a sequence-quality aware, ultra-sensitive variant caller for uncovering cell-population heterogeneity from high-throughput sequencing datasets. Nucleic Acids Res 40(22):11189–11201. https://doi.org/10.1093/nar/gks918
91. Rausch T, Zichner T, Schlattl A, Stutz AM, Benes V, Korbel JO (2012) DELLY: structural variant discovery by integrated paired-end and split-read analysis. Bioinformatics 28 (18):i333–i339. https://doi.org/10.1093/bioinformatics/bts378
92. Layer RM, Chiang C, Quinlan AR, Hall IM (2014) LUMPY: a probabilistic framework for structural variant discovery. Genome Biol 15(6):R84. https://doi.org/10.1186/gb-2014-15-6-r84
93. Chen X, Schulz-Trieglaff O, Shaw R, Barnes B, Schlesinger F, Kallberg M, Cox AJ, Kruglyak S, Saunders CT (2016) Manta: rapid detection of structural variants and indels for germline and cancer sequencing applications. Bioinformatics 32 (8):1220–1222. https://doi.org/10.1093/bioinformatics/btv710
94. Marschall T, Costa IG, Canzar S, Bauer M, Klau GW, Schliep A, Schonhuth A (2012) CLEVER: clique-enumerating variant finder. Bioinformatics 28(22):2875–2882. https://doi.org/10.1093/bioinformatics/bts566
95. Chen K, Wallis JW, McLellan MD, Larson DE, Kalicki JM, Pohl CS, McGrath SD, Wendl MC, Zhang Q, Locke DP, Shi X, Fulton RS, Ley TJ, Wilson RK, Ding L, Mardis ER (2009) BreakDancer: an algorithm for high-resolution mapping of genomic structural variation. Nat Methods 6(9):677–681. https://doi.org/10.1038/nmeth.1363
96. Jiang Y, Wang Y, Brudno M (2012) PRISM: pair-read informed split-read mapping for base-pair level detection of insertion, deletion and structural variants. Bioinformatics 28 (20):2576–2583. https://doi.org/10.1093/bioinformatics/bts484
97. Ye K, Schulz MH, Long Q, Apweiler R, Ning Z (2009) Pindel: a pattern growth approach to detect break points of large deletions and medium sized insertions from paired-end short reads. Bioinformatics 25 (21):2865–2871. https://doi.org/10.1093/bioinformatics/btp394
98. Dolzhenko E, van Vugt J, Shaw RJ, Bekritsky MA, van Blitterswijk M, Narzisi G, Ajay SS, Rajan V, Lajoie BR, Johnson NH, Kingsbury Z, Humphray SJ, Schellevis RD, Brands WJ, Baker M, Rademakers R, Kooyman M, Tazelaar GHP, van Es MA, McLaughlin R, Sproviero W, Shatunov A, Jones A, Al Khleifat A, Pittman A, Morgan S, Hardiman O, Al-Chalabi A, Shaw C, Smith B, Neo EJ, Morrison K, Shaw PJ, Reeves C, Winterkorn L, Wexler NS, Group US-VCR, Housman DE, Ng CW, Li AL, Taft RJ, van den Berg LH, Bentley DR, Veldink JH, Eberle MA (2017) Detection of long repeat expansions from PCR-free whole-genome sequence data. Genome Res 27(11):1895–1903. https://doi.org/10.1101/gr.225672.117
99. Willems T, Zielinski D, Yuan J, Gordon A, Gymrek M, Erlich Y (2017) Genome-wide profiling of heritable and de novo STR variations. Nat Methods 14(6):590–592. https://doi.org/10.1038/nmeth.4267
100. Tankard RM, Bennett MF, Degorski P, Delatycki MB, Lockhart PJ, Bahlo M (2018) Detecting expansions of tandem repeats in cohorts sequenced with short-read sequencing data. Am J Hum Genet 103(6):858–873. https://doi.org/10.1016/j.ajhg.2018.10.015
101. Tang H, Kirkness EF, Lippert C, Biggs WH, Fabani M, Guzman E, Ramakrishnan S, Lavrenko V, Kakaradov B, Hou C, Hicks B, Heckerman D, Och FJ, Caskey CT, Venter JC, Telenti A (2017) Profiling of short-tandem-repeat disease alleles in 12,632 human whole genomes. Am J Hum Genet 101(5):700–715. https://doi.org/10.1016/j.ajhg.2017.09.013
102. Benson G (1999) Tandem repeats finder: a program to analyze DNA sequences. Nucleic Acids Res 27(2):573–580. https://doi.org/10.1093/nar/27.2.573
103. Mayer C, Leese F, Tollrian R (2010) Genome-wide analysis of tandem repeats in Daphnia pulex—a comparative approach. BMC Genomics 11:277. https://doi.org/10.1186/1471-2164-11-277
104. Keane TM, Wong K, Adams DJ (2013) RetroSeq: transposable element discovery from next-generation sequencing data. Bioinformatics 29(3):389–390. https://doi.org/10.1093/bioinformatics/bts697
105. Wu J, Lee WP, Ward A, Walker JA, Konkel MK, Batzer MA, Marth GT (2014) Tangram: a comprehensive toolbox for mobile element insertion detection. BMC Genomics 15:795. https://doi.org/10.1186/1471-2164-15-795
106. Kanehisa M, Goto S (2000) KEGG: kyoto encyclopedia of genes and genomes. Nucleic Acids Res 28(1):27–30. https://doi.org/10.1093/nar/28.1.27

107. Desvignes JP, Bartoli M, Delague V, Krahn M, Miltgen M, Beroud C, Salgado D (2018) VarAFT: a variant annotation and filtration system for human next generation sequencing data. Nucleic Acids Res 46(W1): W545–W553. https://doi.org/10.1093/nar/gky471
108. Liu X, Jian X, Boerwinkle E (2011) dbNSFP: a lightweight database of human nonsynonymous SNPs and their functional predictions. Hum Mutat 32(8):894–899. https://doi.org/10.1002/humu.21517
109. Cingolani P, Platts A, le Wang L, Coon M, Nguyen T, Wang L, Land SJ, Lu X, Ruden DM (2012) A program for annotating and predicting the effects of single nucleotide polymorphisms, SnpEff: SNPs in the genome of Drosophila melanogaster strain w1118; iso-2; iso-3. Fly (Austin) 6(2):80–92. https://doi.org/10.4161/fly.19695

Chapter 2

Statistical Considerations on NGS Data for Inferring Copy Number Variations

Jie Chen

Abstract

The next-generation sequencing (NGS) technology has revolutionized research in genetics and genomics, resulting in massive NGS data and opening more fronts to answer unresolved issues in genetics. NGS data are usually stored at three levels: image files, sequence tags, and alignment reads. The sizes of these types of data usually range from several hundreds of gigabytes to several terabytes. Biostatisticians and bioinformaticians are typically working with the aligned NGS read count data (hence the last level of NGS data) for data modeling and interpretation.

To horn in on the use of NGS technology, researchers utilize it to profile the whole genome to study DNA copy number variations (CNVs) for an individual subject (or patient) as well as groups of subjects (or patients). The resulting aligned NGS read count data are then modeled by proper mathematical and statistical approaches so that the loci of CNVs can be accurately detected. In this book chapter, a summary of most popularly used statistical methods for detecting CNVs using NGS data is given. The goal is to provide readers with a comprehensive resource of available statistical approaches for inferring DNA copy number variations using NGS data.

Key words Bayesian analysis, CNVs, Information criterion, Likelihood ratio test, NGS reads, Read counts, Read depth, Statistical change point analysis

1 Introduction and Background

Following the breakthrough of the microarray technology in 1990s, the innovation of the next-generation sequencing (NGS) technology early in the twenty-first century has made it possible to study more biological problems at the genomic level. NGS technology is now widely used in variety of biological experiments. For instance, RNA-sequencing is popularly used to quantify mRNA abundance and the differential levels of transcripts under different conditions (normal versus cancerous, wild type versus knock-out type, among others); DNA-sequencing is used in whole genome sequencing for DNA variants detection; ChIP-sequencing is used to study protein-DNA interactions, and bisulfate sequencing is applied to study DNA methylation, etc. In this chapter, we will

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_2,

briefly discuss how to infer DNA copy number variants or variations (CNVS) using DNA-sequencing data.

CNVs are genomic regions where DNA copy number deviates from the normal copy number. For example, the normal copy number for human being is 2. Genetic variation exists in all humans and takes different forms, thus making one individual different from another individual, and the abundance of CNVs can range from kilobases (kb) to megabases (Mb) in size [1]. CNVs account for about twenty percent of variation in gene expression [2]. It is also known that variations in DNA copy numbers are common in cancer and other diseases. Therefore, identification of CNVs and determination of their boundaries (that outline a CNV region) will facilitate the development of medical diagnostic tools and treatment regimes for cancer, genetic and other diseases. In the last decade or so, array Comparative Genomic Hybridization (aCGH) technology and the single nucleotide polymorphism (SNP) arrays were dominantly used in modern copy number study experiments. With the most recently available and more affordable NGS technology, there are more opportunities to detect CNVs using the NGS reads (also called shotgun reads or read counts) data. The NGS reads data are quite different from aCGH and SNP array data, therefore, identifying CNV regions using NGS reads requires appropriate modeling of the data.

The bridge from genome-wide sequencing data to medical diagnostics of diseases, such as cancer, osteoporosis, autism, and other genetic disorders, should be built on a solid statistical analysis of the information rich NGS data. Therefore, the development of statistical methods becomes important in the study of CNVs. In this chapter, we try to provide a timely review of recently developed statistical methods for CNV studies using NGS data. We aim to provide the readers whose research emphasis is in statistical methodology development with a new aspect of statistical application in genomics data; and to present the readers whose research foci are in the study of CNVs some available statistical modeling that are suitable for interpreting their research outcomes.

This chapter expands our previous review paper [3] to include additional recent works in statistical approaches for CNV detection using NGS data. The rest of the chapter is organized as follows. A brief review of NGS technology for CNV detection is given in Subheading 2. Computational methods developed for detecting CNVs using NGS data are briefly introduced in Subheading 3, and statistical methods are emphasized and reviewed in Subheading 4. A summary and conclusion section is given at the end.

2 NGS Technology for CNV Detection: A Brief Summary

When the NGS technology is used in DNA copy number study experiments, the entire genome is first broken into small pieces, and these small pieces are then ligated to adapters for massive parallel sequencing. The pool of millions of replicates from sequencing is called a library of reads. Reads are then mapped and assembled back to the genome. The reads and count of reads for each genomic position are known as the NGS raw data. Along the pipeline, raw reads (typically in the form of fastq files) from human DNA-sequencing experiments are aligned to the human genome (hg19) using Bowtie [4], aligned reads are further processed to correct bias occurred during sequencing and are then normalized by the GC content, since it was shown in [5, 6] that the NGS read counts are highly dependent on GC content. The normalized read count data is used for detecting CNVs in the follow-up modeling or computational steps.

The above sequencing and alignment procedures are repeated for both the test (target, tumor, cancer, etc.) genome and the reference (control) genome. However, scientific questions are often interested in the test genome rather than the reference genome. With respect to detecting CNVs in the test genome, algorithms are developed for different data type including (1) data from both the test and the reference genome; (2) data of the test genome by standardizing the reference genome copy number to be one; (3) multiple sample data from the test genome by standardizing the reference copy number to be one, etc.

It is a consensus in the biological community that the number of reads aligned in a given region of the genome is proportional to the DNA copy numbers in that region. Therefore the excessive or moderate number of reads in a region is hypothesized to have DNA copy number addition or deletion, in comparison with neighboring regions.

In the literature, many computational and statistical methods have been developed for detecting CNVs in the experimental data resulting from using the microarray based technology such as the aCGH technology and the SNP arrays. There are, however, only a few methods that are derived for the CNV detection directly using NGS data. The computational and statistical analysis of NGS reads data for CNV detection faces challenges in several ways: the reads data are of high throughput, high noises, and having low mappability. Various computational and statistical methods have been proposed to analyze aligned NGS reads data for CNV detection recently. Among these many approaches, we summarize them into two broad classes of methods: computational approaches and statistical model-based approaches.

As the chapter is focusing on statistical aspects of CNV detection using NGS data, our emphasis will be on the statistical model-based approaches, hence we will only briefly review two computational approaches in Subheading 3, and we will provide some detailed reviews on several statistical model-based approaches in Subheading 4.

3 Computational Approaches

Although some computational approaches assume mathematical and statistical models in the first place, compared to the elaborated statistical models to be introduced in Subheading 4 and their extensive computational elements in the respective algorithms, we classify them as computational approaches and we summarize two of the computational algorithms in this section.

3.1 The SeqSeq Software Tool

A lung cancer NGS data was analyzed in [7] using a local change-point method followed by a merging procedure that joins adjacent segments. Specifically, for each tumor read mapping position, a window is created by extending to the left and to the right, respectively, to include w reads, where w is a pre-defined parameter. A log-ratio statistic D for the tumor position of interest is calculated based on read counts from the tumor and normal samples at the two sides of the tumor position within the window. A p-value was computed for change point detection based on the D statistics. If the p-value is smaller than an initially pre-defined parameter p_{init}, then the tumor position of interest will be in the list of potential breakpoints for the next step. The next step is to examine these potential breakpoints to reduce false discovery rate (FDR). Specifically, the whole chromosome or genome are segmented by the list of potential breakpoints detected in the above step. Starting from the least significant breakpoint, using the read count data in the tumor and the normal samples in the entire segment, which is often longer than a window in the first step, the algorithm iteratively joins adjacent segments if the p-value is greater than the pre-defined threshold p_{merge}. The algorithm is illustrated in Fig. 1, i.e., Figure 2 (a), (b), (c) in [7]. A software tool, SegSeq [7], was developed to implement the above described algorithm.

3.2 The rSW-seq Algorithm

Another computational approach is rSW-seq, an algorithm developed by Kim et al. [8] using dynamic programming and is purely based on sequence patterns.

In step one, all reads from both tumor and normal samples are combined and sorted based on genomic positions from the smallest to the largest. Then, weights are assigned to reads. If the tumor sample and the normal sample have the same number of reads, then the weight for a tumor read, $W_T = 1$, and the weight for a normal

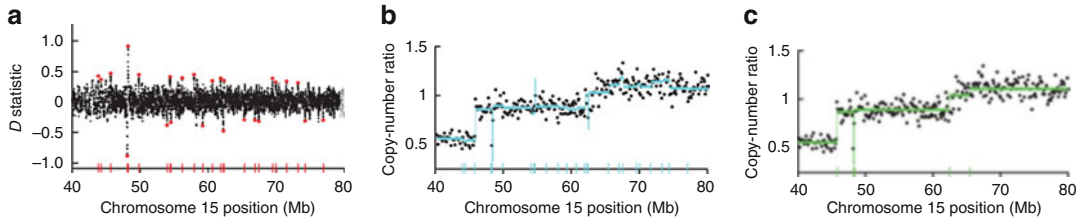


Fig. 1 Segmentation algorithm for aligned sequenced reads. (**a**) Candidate breakpoints (red dots) correspond to tumor read positions (black dots) whose local log-ratio statistic, *D*, passes a lenient significance threshold. (**b**) These candidate breakpoints define the boundaries of the initial copy number segments (blue lines). Each point represents the estimated copy number ratio for a 100-kb window. (**c**) A merging procedure yields the final list of copy number segments (green lines) obtained for 10 genome-wide false positives. Reproduced from [7] with permission from Springer Nature

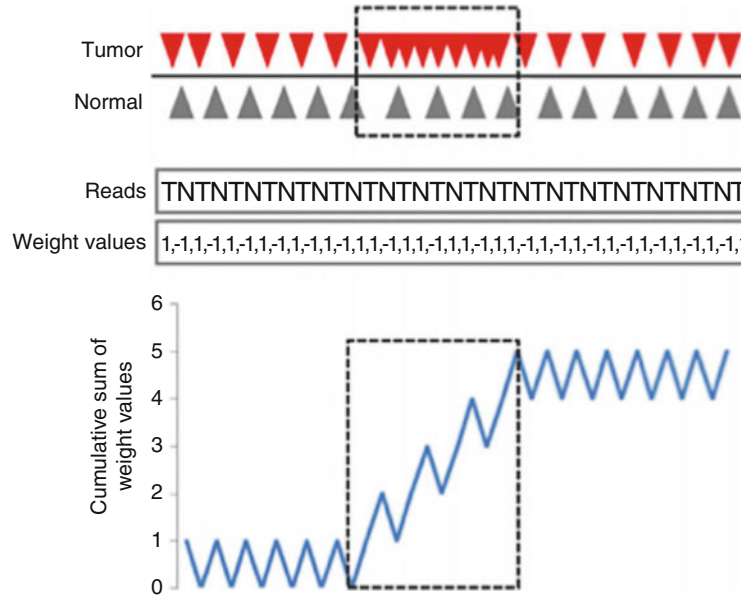


Fig. 2 The sequencing reads for tumor (red triangles) and matched normal genomes (black triangles) are shown. For simplicity, we assume the weights are $W_T = 1$ and $W_N = -1$. The cumulative sum of weight values shows an upward slope (indicated by the box) for a region of local copy number gain (prevalence of tumor reads over normal reads). Figure adopted from Open Access article [8]

read, $W_N = -1$, are equal in absolute values with opposite sign. Otherwise, in order to adjust library size bias, the weights are computed as $W_T = 2N_N/(N_T + N_N)$ and $W_N = -2N_T/(N_T + N_N)$, where N_N is the total number of normal reads and N_T is the total number of tumor reads. Then a cumulative sum of weight values can be computed for any given region. An example is given in Fig. 2, i.e., Figure 1 A in [8]. Let $r_j = (W_j, s_j)$, $j = 1, \ldots, N$, where

r_j is the jth read combining the target and normal samples; W_j is the weight for read j; and s_j is the location of read j. Then using a slightly modified Smith-Waterman algorithm [9], one can find the genomic region (s_l, s_m) such that the accumulative weight sum, $S(l, m) = \sum_{j=l}^{m} W_j$, is maximized, and (s_l, s_m) is a potential copy number gain region. For copy number loss region, one can search for the region where the cumulative sum is negatively maximized. When there is no copy number change, the cumulative sum in a region should be close to 0. Due to the noise in sequencing and alignment, a pre-defined threshold is used to reduce false positives. rSW-seq stands for recursive Smith-Waterman-seq. In fact, the Smith-Waterman algorithm was previously used in detecting CNVs for array-CGH data proposed in [10].

4 Statistical Model-Based Approaches

As the NGS data are relatively new in comparison with the microarray data, only a few statistical model-based approaches have been presented in modeling NGS data for CNV detections. We selected eight methods to review by categorizing them into four groups based on their main ideas. We entail these selected methods in the following subsections.

4.1 Hidden Markov Models

Reading through the literature one can notice that the Hidden Markov Model (HMM) [11–14] has been applied to detect CNVs in several papers for the aCGH data, such as the Bayesian method in [15] and BioHMM by Marioni et al. [16]. Inspired by the HMM application in aCGH data, a few examples of using HMM for detecting CNVs using the NGS data are seen in CNAseg [17] and m-HMM [18]. In general, these methods are formulated in three steps: segmentation, state estimation using HMM, and merging (or change point adjustment).

The HMM-type methods start with segmentation of the chromosome or genome of interest into windows and assume that, within each window, the genomic positions share the same copy number state. The algorithm CNAseg, developed by Ivakhno et al. [17], requires that all windows have a fixed genomic distance. The m-HMM method [18] uses K-means method to form windows by joining adjacent genomic sites. Specifically, each chromosome is segmented into 20 parts, where the breakpoints correspond to the longest physical distances between adjacent sites. Then a K-means clustering method is applied to further group adjacent genomic positions in each of the 20 parts. The value of K is selected such that after clustering, each group has, on average, around 40 genomic positions, as recommended by Wang et al. [18].

After forming windows, we use w to index windows, where $w = 1, \ldots, W$, and let $u_w^{[t]}$ denote the total number of reads in window w for the test sample, and $u_w^{[r]}$ for the reference sample accordingly. A Hidden Markov chain is established with the following elements: the hidden states, transition probabilities among the hidden states, observations, and the emission probabilities for observations given hidden states. In [18], there are four designated states: "copy number gain," "no change (or normal)," "copy number loss," and "(gene) absent," respectively, when comparing the test sample against the reference sample. An HMM assumes that the sequence of unobserved hidden states following a random process with a sequence of observations emitted from these hidden states.

Using the notations in [18], an HMM is a bivariate random process $\{S_w, U_w\}$, $w = 1, \ldots, W$, where $\{S_w\}$ is the sequence of unobserved hidden states and $\{U_w\}$ is the sequence of observations. The transition probabilities have Markov property, i.e., $P(S_{w+1}|S_1, \ldots, S_w) = P(S_{w+1}|S_w)$, which is modeled as $P(S_{w+1} = l|S_w = k; p_{kl}, \rho, d_w) = p_{kl}(1 - e^{-\rho d_w})$ when $l \neq k$, and $P(S_{w+1} = l|S_w = k; p_{kl}, \rho, d_w) = 1 - \left(\sum_{j \neq k} p_{kl}\right)(1 - e^{-\rho d_w})$ when $l = k$ for k, $l = 1, \ldots, 4$ and $w = 1, \ldots, W - 1$. p_{kl}'s and ρ are the model parameters to be estimated. If we use the median genomic position of a window to denote the window location, then d_w is the genomic distance between windows w and $w + 1$. The rationale behind the model is that the greater the distance between two windows, the higher chance of copy number change from one state to another. Furthermore, suppose the reference read counts follow a Poisson process that $U_w^{[r]}|\lambda_w^{[r]} \sim \text{Poisson}(\lambda_w^{[r]})$ where $\lambda_w^{[r]}$ is the model parameter. Then the emission probability $P(U_w|S_w)$ in the target genome can be modeled as a mixture of Poisson distributions as in (1).

$$U_w^{[t]}|(\lambda_w^{[r]}, S_w = k, q_{kj}, v_{kj}) \sim \sum_{j=1}^{4} q_{kj}\text{Poisson}(v_{kj} c_0 \lambda_w^{[r]}), \qquad (1)$$

where c_0 is a normalization scale factor on the read counts,

$$\mathbf{Q} = [q_{kj}]_{4\times 4} = \begin{pmatrix} q_{11} & 1 - q_{11} & 0 & 0 \\ \frac{1 - q_{22}}{2} & q_{22} & \frac{1 - q_{22}}{2} & 0 \\ 0 & \frac{1 - q_{33}}{2} & q_{33} & \frac{1 - q_{33}}{2} \\ 0 & 0 & 1 - q_{44} & q_{44} \end{pmatrix}$$

and

Table 1
mHMM output for chromosome 10 of maize dataset in [18]. Only segments with more than 1 Mb are listed. The "Left" and "Right" columns indicate the starting and ending genomic positions. The "Ratio" column represents the normalized ratio of read counts between Mo17 and B73 for the genomic region between the start position and the end position. The normalization factor $c_0 = 0.493$. That is, ratio is Mo17/($c_0 \times$B73). Reproduced from [3] with permission from John Wiley and Sons

Left	Right	State	Length	Ratio
23,423,717	24,579,530	Deletion	1,155,813	0.488
25,823,914	27,456,101	Deletion	1,632,187	0.525
34,987,963	36,387,068	Deletion	1,399,105	0.683
39,062,280	41,276,135	Deletion	2,213,855	0.501
73,084,028	74,554,051	Deletion	1,470,023	0.453

$$\mathbf{V} = [v_{kj}]_{4\times 4} = \begin{pmatrix} 2 & v_{12} & 0 & 0 \\ v_{21} & 1 & v_{23} & 0 \\ 0 & v_{32} & 0.5 & v_{34} \\ 0 & 0 & v_{43} & 0 \end{pmatrix}$$

with constraints $q_{kk} \in (0.5, 1)$ for $k = 1, \ldots, 4$, $v_{12}, v_{21} \in (1, 2)$, $v_{23}, v_{32} \in (0.5, 1)$, $v_{34}, v_{43} \in (0, 0.5)$. The rationale of using a mixture Poisson model is to incorporate extra noise in read counts due to sequencing and alignment. Parameters in the model are estimated through the Expectation-Maximization (EM) algorithm, which is a classic approach for estimating parameters for HMMs.

We used one dataset from [18] as an example—the chromosome 10 of maize varieties B73 (as the reference genome) and Mo17 (as the target or test genome). The output of using m-HMM for detecting CNVs for chromosome 10 of the maize dataset is shown in Table 1, which gives the estimated start and end genomic sites for copy number changes, the state of the test genome compared to the reference, the length of the CNV region, and the ratio defined as Mo17/($c_0 \times$B73) with $c_0 = 0.493$ for this example dataset.

It should be noted that using HMM, each window has an estimated "state," depending on the formation method of windows, henceforth, there could be various problems associated with it. For example, when the CNAseg method is used on NGS data, false positive rate (FPR) could be high since windows are formed by using predetermined size of genomic distance in the reference genome when within-sample variation is not assessed. In addition, the true change points may not coincide with the window boundaries when either the CNAseg method or the m-HMM methods is used for NGS data. The CNAseg method utilizes the

within-sample and inter-sample variation and Pearson χ^2 test to merge adjacent windows to further reduce FPR. The m-HMM uses Pearson χ^2 test to fine-tune the change point position by searching through all positions in the vicinity of the window boundaries where change points are reported. By doing so, both CNAseg method and m-HMM method have reduced FPR to some extent. Therefore, reducing FPR becomes one of the goals for improving HMM-based methods in the future.

4.2 Shifting Level Model

A joint shifting level model (JointSLM) was proposed [19] for detecting CNVs for the NGS data, which is based on a similar idea in [20] for array data. The model and method can be briefly summarized as follows.

JointSLM takes data from multiple samples. Let t index samples, $t = 1, \ldots, M$, where M denotes the total number of samples. For each target genome sample, read count data are measured by counting the number of mapped reads in every 100 base pair (bp) windows, GC content corrected and median normalized to copy number 2. Then the data are further transformed to log-scale with base 2. Let N denote the total number of windows and x_{it} denote the read count of window i ($i = 1, \ldots, N$) from sample t ($t = 1, \ldots, M$). In addition, let x_i denote the vector $(x_{i1}, \ldots, x_{iM})^T$, which is the vector of observations for window i from all M sequencing samples.

The JointSLM models CNV as the change of means in a stochastic process, i.e.,

$$x_i = m_i + e_i \text{ and } m_i = (1 - z_{i-1}) \cdot m_{i-1} + (z_{i-1}) \cdot \mu_i, \qquad (2)$$

where $m_i = (m_{i1}, \ldots, m_{iM})^T$ is the vector of unobserved means; e_i is the vector of error terms following a multivariate normal distribution with mean vector 0 and a variance-covariance matrix Σ_e; z_i's are binomial variables with $P(z_i = 1) = \eta$, and μ_i is a vector of normal variables with mean μ and variance-covariance matrix Σ_μ. Thus, we have $e_i \sim N(0, \Sigma_e)$ and $m_i \sim N(\mu, \Sigma_\mu)$. To simplify, independence is assumed across samples, i.e., Σe is a diagonal matrix with $\sigma^2_{e,t}$ at the tth position on the diagonal, and Σ_μ is a diagonal matrix with $\sigma^2_{\mu,t}$ at the tth position on the diagonal.

Model (2) contains two sequences: a hidden sequence of unobserved underlying states (m_i, z_i)'s and an observed sequence of emission states x_i's. Thus JointSLM model in fact is an HMM model though taking a different perspective in initial modeling. Utilizing the HMM inference method and pooling data from multiple samples, model parameters, the number of states, and the best sequence of (m_i, z_i)'s are derived. JointSLM is one of a few methods that uses multiple samples information for CNV detection using NGS data.

4.3 Change Point Model

The change point model is very feasible for CNV detection naturally due to their model characteristics. Namely, the CNVs reflected via the data are visually and conceptually conceived as parameter changes in an underlying model of the data. The change point analysis methods can be roughly classified into two categories: the frequentist's approach, such as likelihood ratio based methods, and the Bayesian approach. These two types of approaches are, respectively, introduced in the following two subsections. Extensive reviews of change point analysis methods can be found in [21, 22], and [23], to just name a few.

4.3.1 Frequentist Approaches

We will present two approaches in this section. The first one is based on a mean change point model. The Circular Binary Segmentation (CBS) method proposed by Olshen et al. [24], built on a mean change point Gaussian model, is a popular method for detecting CNVs using array-CGH data. The computational implementation of CBS was later improved and implemented in the R package DNAcopy [25]. Notably, several methods/algorithms proposed lately for the detection of CNV regions using NGS data are actually making use of the CBS algorithm, therefore, we want to attribute our summary to the CBS method first. The CBS algorithm can be adopted for detecting CNVs using NGS data for either the log ratio of read counts from the test and reference samples or normalized read counts from the test sample alone if normality in these two types of NGS data is justified. Moreover, since the R package by Venkatraman and Olshen [25] improves the work in [24], we will focus on introducing the specific algorithm in [25] as follows.

Let $X_1, \ldots, X_m$ be the observations from a chromosome or a whole genome of interest. Moreover, $X_1, \ldots, X_m$ are assumed to follow Gaussian distributions with possible changes in the means and a constant variance. The test statistic is the maximal t-statistic given by searching for locations i and j such that $T = \max_{1 \le i < j \le m} |T_{ij}|$, where

$$T_{ij} = \frac{\bar{Y}_{ij} - \bar{Z}_{ij}}{S_{ij}\{(j-i)^{-1} + (m-j+1)^{-1}\}^{1/2}}, \tag{3}$$

$\bar{Y}_{ij} = (X_{i+1} + \cdots + X_j)/(j-i)$, $\bar{Z}_{ij} = (X_1 + \cdots + X_i + X_{j+1} + \cdots + X_m)/(m-j+i)$, and S_{ij} is the pooled mean squared error. A change is statistically significant if the p-value of the test statistics is smaller than a pre-defined type I error threshold α, and the estimated change points are locations i and j that maximize the test statistic.

One way to compute p-value for a given test statistic is to use the permutation method, which is computationally extensive. In order to improve the computational efficiency, Venkatraman and Olshen [25] proposed a method to calculate the so-called hybrid p-value. Specifically, suppose the test statistic T for observations X_1,

$\ldots, X_m$ is of value b. If m is smaller than m_0, then a permutation test is adopted to compute the permutation p-value. If m is larger than m_0, then an approximated p-value is computed as follows. For a given proper k, one can construct two complement sets $A_1 = \{i, j: j - i \le k \text{ or } > m - k\}$ and $A_2 = \{i, j: k+1 \le j - i \le m - k\}$. Let $T_l = \max_{A_l} |T_{ij}|, l = 1, 2$, and $T = \max\{T_1, T_2\}$. Notice that for any observed test statistic with value b, we have $P(T_1 > b) + P(T_2 > b) \ge P(T > b) \ge P(T_2 > b)$. According to [26, 27] and [28], let $\delta = k/m$, ϕ be the standard normal density function and Φ be the standard normal cumulative function, we have

$$P(T_2 > b) \approx \frac{1}{2} b^3 \phi(b) \int_{1/2}^{1-\delta} \frac{v^2(b/(mt(1-t))^{1/2})}{t^2(1-t)^2} dt, \tag{4}$$

where function $v(x)$ is defined as

$$v(x) = 2x^{-2} \exp\left\{ -2 \sum_{l=1}^{\infty} l^{-1} \Phi\left(-\frac{1}{2} x l^{1/2} \right) \right\}. \tag{5}$$

If $P(T_2 > b) > \alpha$ computed by (4), declare that there is no change. Otherwise, compute $P(T_1 > b) + P(T_2 > b)$ where $P(T_2 > b)$ is computed as in (4) and $P(T_1 > b)$ is computed using the permutation method. Computing $P(T_1 > b)$ by the permutation method only requires mk statistics, thus it is computationally efficient for properly chosen k. If $P(T_1 > b) + P(T_2 > b) < \alpha$, a change is declared. Through simulation studies, Venkatraman and Olshen [25] suggests $k = 25$ for any $m \le 1000$, and increase k by 5 as m is doubled.

For searching multiple change points, the CBS algorithm takes data from the whole chromosome or the whole genome, and looks for the most significant change points, i.e., locations i and j. If no change is more significant than the pre-defined threshold α, then the algorithm stops and claims no change point. Otherwise, the whole chromosome or the whole genome will be broken into segments with breakpoints as the change points. Within each segment, repeat the above procedure to search the most significant breakpoint. The algorithm stops until no more significant change points can be claimed for the subsegments.

An example of using Chromosomes 8 and 9 of the lung cancer H2347 dataset from [7] is shown in Fig. 3. Read counts for each 10,000 bp in genomic positions are grouped in one bin for the target and reference genomes, respectively. The logarithm of ratio of read counts from the target genome and reference genome for each bin is used.

Although the CBS algorithm is quite straightforward, its Gaussian assumption may not be proper for modeling the intrinsically discrete NGS data. A similar idea [29] was adopted from CBS algorithm with improvement. The algorithm applies to GC-corrected test sample data, and assumes negative binomial

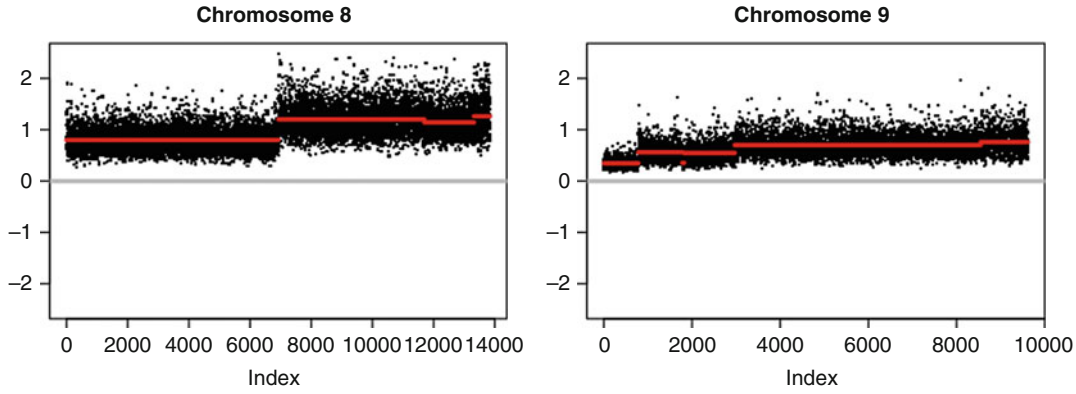


Fig. 3 H2347 Chromosome 8 and Chromosome 9 segmentation results using DNAcopy. The data are the log ratio of target genome read counts versus the reference genome read counts adjusted by GC content. The red line shows the estimated mean copy numbers. The vertical axis shows the log ratios, and the horizontal axis shows the genomic positions in 10 kbp. Reproduced from [3] with permission from John Wiley and Sons

models. Then a circular binary segmentation algorithm is iteratively applied to detect multiple change points. In order to speed up the computing, a parallel strategy was also adopted. The algorithm was built into an R package called ReadDepth.

Alternatively, the NGS read count data were modeled in [30] as inhomogeneous Poisson Processes. They [30] approached the problem of detecting CNVs using NGS data in a two-fold way. The first aspect is to propose a score and a generalized likelihood ratio statistics for detecting CNVs assuming the number of change points K is known; and the second fold is to propose a modified Bayes information criterion (mBIC) for change point model selection also known as deciding the number of change points. Their algorithm is written into an R package [30], SeqCBS, and can be briefly summarized as follows.

Let $\{U_1, \ldots, U_{m1}\}$ and $\{V_1, \ldots, V_{m2}\}$ be the mapped genomic locations for all reads from the target genome and the reference genome, respectively, and m_1 and m_2 are the total number of reads from the target and the reference samples, respectively. Let $W_1, \ldots, W_m$ be the combined events, and $m = m_1 + m_2$ be the total number of reads. Define Z_i, $i = 1, \ldots, m$,

$$Z_i = \begin{cases} 1, & \text{if } W_i \in \{U_1, \ldots, U_{m1}\} \\ 0, & \text{if } W_i \in \{V_1, \ldots, V_{m2}\}, \end{cases} \tag{6}$$

and $Z_i = 1$ is called "success."

For testing a single copy change interval spanning from position i to position j, it is equivalent to test the null hypothesis H_0: $p_{ij} = p$, where p_{ij} is the success rate inside the interval, and p is the overall success rate. Two test statistics were proposed according to

the size of the interval (i, j). The first one is based on the conditional score statistic that, as discussed in [31],

$$S_{ij} = \sum_{i \le k \le j} Z_k - \hat{p}(j - i + 1), \tag{7}$$

where $\hat{p} = \sum Z_k / m$. The variance of S_{ij} under the null model is $\hat{\sigma}^2_{ij} = (1 - (j - i + 1)/m)(j - i + 1)p(1 - p)$. With the above notations, the standard score statistic $T_{ij} = S_{ij}/\sigma_{ij}$ approximately follows the standard normal distribution if $(j - i)$ is large. Otherwise, a second test statistic, based on binomial generalized likelihood ratio test, is proposed to achieve the propose to test for a change point. Define

$$\begin{aligned} \Lambda_{ij} &= \sup_{p_0, p_{ij}} l_1(p_0, p_{ij}) - \sup_p l_0(p) \\ &= \sum_{k \in \{i,j\}} \left\{ Z_k \log\left(\frac{\hat{p}_{ij}}{\hat{p}}\right) + (1 - Z_k)\log\left(\frac{1 - \hat{p}_{ij}}{1 - \hat{p}}\right) \right\} \\ &\quad + \sum_{k \notin \{i,j\}} \left\{ Z_k \log\left(\frac{\hat{p}_0}{\hat{p}}\right) + (1 - Z_k)\log\left(\frac{1 - \hat{p}_0}{1 - \hat{p}}\right) \right\}, \end{aligned} \tag{8}$$

where $\hat{p} = \sum_{k=1}^{m} Z_k / m$, $\hat{p}_{ij} = \sum_{k \in \{i,j\}} Z_k/(j - i + 1)$, and $\hat{p}_0 = \sum_{k \notin \{i,j\}} Z_k/(m - j + i - 1)$. The log-likelihood test statistic Λ_{ij} in (8) can be interpreted as the log-likelihood comparison between the null model where one overall success rate p is assumed and the alternative model with one parameter p_{ij} within the interval (i, j) and another parameter p_0 outside the interval. For searching multiple change points, a CBS similar algorithm was adopted.

One issue that needs to be addressed is to choose the number of change points, K. Similar to the idea in [32], a modified Bayes information criterion (mBIC), inspired by Schwarz [33], is adopted by Shen and Zhang [30] as in (9), to aid the determination of K,

$$\begin{aligned} \text{mBIC}(K) &= \log\left(\frac{\sup_{p(t),\tau} L(p(t), \tau)}{\sup_p L(p)}\right) - \frac{1}{2}\sum_{k=0}^{K} \log(\hat{\tau}_{k+1} - \hat{\tau}_k) \\ &\quad + \frac{1}{2}\log(m) - K\log(m'), \end{aligned} \tag{9}$$

where m' is the number of unique values in $\{W_1, \ldots, W_m\}$; τ_k's are the set of change points, where $k = 0, \ldots, K+1$ and $\{1 = \tau_0 < \tau_1 < \cdots < \tau_{K+1} = m\}$; and $p(t) = p_k$ if $\tau_k \le t < \tau_{k+1}$. The mBIC in (9) can be interpreted as the generalized log-likelihood ratio for the model with K change points versus the null model with no change points, and then deducting the penalty terms. The number of change points, K, is estimated to be $\hat{K} = \text{argmax}_K \text{mBIC}(K)$.

With the help of mBIC, the situation of too many change points is penalized, and the number of change points $\hat{K}$ is more reasonable to be used in the test statistic (8) for assessing the significance of the CNV region in comparison with the whole chromosome. Moreover, Bayesian confidence intervals on the estimated copy number for any given regions are also provided in [30].

4.3.2 Bayesian Approaches

Another way for assessing significance of change points (or CNVs in the genomic data context) is through Bayesian inference. Specifically, with suitable choices of prior distributions on model parameters, a posterior probability distribution of the change point is analytically or computationally derived. Then the change point is the one that maximizes the posterior probability among all possible change point locations. One advantage of using Bayes method is to avoid lengthy recursion and large factorials that are often involved in the computation of the p-values of the test statistics used in frequentist approaches. We present three Bayesian approaches, one is based on the ratio of read counts, the other is for normalized read counts, and the third one is a Bayesian approach coupled with an on-line change point model for read counts data. Note that the read counts discussed hereinafter are all pre-processed following standard NGS data processing protocol and are put into bins of certain size (1K bp, 10K bp, etc.).

Modeling the Log Ratio of Read Counts NGS Data

In [34], a mean and variance change point model for the log ratio of read counts from target genome and reference genome is proposed and a Bayesian solution, based on [35], along with an R package, SeqBBS, was established. The strategy for searching multiple CNVs using the Bayesian approaches is to use a sliding window of certain size, say m bins, by assuming that within each window there is potentially at most one CNV. Specifically, let Υ_i represent the logarithm with base 2 of the normalized and GC-corrected read counts of the sample genome to that of the reference genome at the ith bin along the chromosome under study, where $i = 1, \ldots, m$, and m is the total number of bins in a window of size m. In addition, Υ_i is assumed to follow approximately a normal distribution with mean μ_i and variance σ_i^2. A change point is the unknown k, $k = 2, \ldots, m-1$, to be estimated after testing

$$H_0 : \mu_1 = \mu_2 = \ldots = \mu_m = \mu \text{ and } \sigma_1^2 = \ldots = \sigma_m^2$$

against

$$\begin{aligned} &H_1 : \mu_1 = \ldots = \mu_k \neq \mu_{k+1} = \ldots = \mu_m \text{ and} \\ &\sigma_1^2 = \ldots = \sigma_k^2 \neq \sigma_{k+1}^2 = \ldots = \sigma_m^2, \end{aligned} \tag{10}$$

with H_0 being rejected.

Assuming a discrete uniform distribution as the prior for the unknown change point location k and constant priors on the mean and variance parameters, the posterior distribution for k, $p_1(k)$, was obtained in [34] as in (11).

$$p_1(k) = \frac{p_1^*(k)}{\sum_{j=2}^{m-2} p_1^*(j)}, \tag{11}$$

for $k = 2, \ldots, m-2$, and

$$\begin{aligned} p_1^*(j) \quad = \quad & \Gamma\left(\frac{j-1}{2}\right)\Gamma\left(\frac{m-j-1}{2}\right) j^{\,j/2-1}(m-j)^{(m-j)/2-1} \\ & \times\left\{ j\sum_{i=1}^{j} Y_i^2 - \left(\sum_{i=1}^{j} Y_i\right)^2 \right\}^{-(j-1)/2} \\ & \times\left\{ (m-j)\sum_{i=j+1}^{m} Y_i^2 - \left(\sum_{i=j+1}^{m} Y_i\right)^2 \right\}^{-(m-j-1)/2}. \end{aligned} \tag{12}$$

The algorithm was written into a freely available R package, SeqBBS. The algorithm was applied to NGS data on the tumor cell line HCC1954 and the matched normal cell line BL1954 [7]. Figure 4 presents the analysis outcome of using the algorithm on 100K-binned NGS ratio data of HCC1954/BL1954.

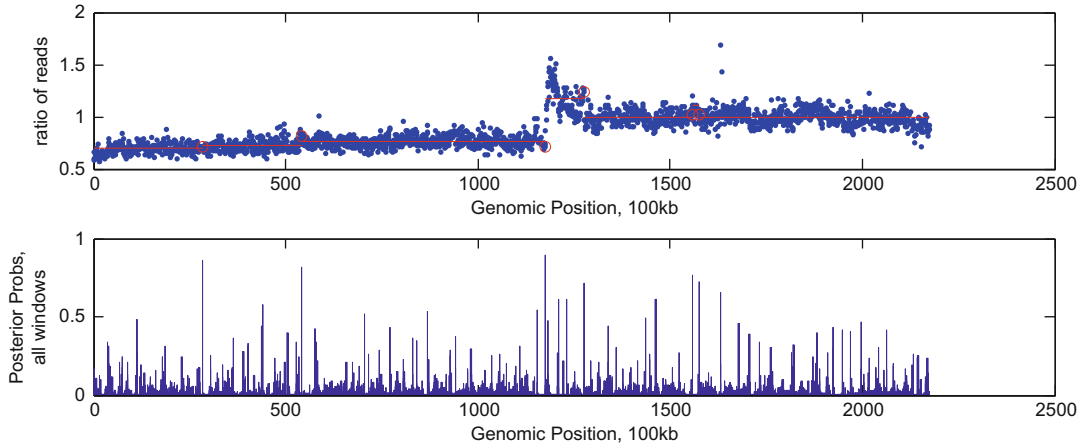


Fig. 4 HCC/BL1954 Chromosome 1 with CNV regions identified. Upper panel: a scatter plot of the log base 2 reads ratios with breakpoints identified as red circles and red horizontal line as the mean of each segment between two identified breakpoints for the threshold of 0.70 and window size of 20. Lower panel: the posterior probabilities for each position within each window. Reproduced from [3] with permission from John Wiley and Sons

Modeling the NGS Read Count Data Using Data Transformation

For the one sample NGS read count data, the problem of detecting one change point in a given window of size m for read counts $X_1, \ldots, X_m$ was formulated [36] as hypothesis testing for $H_0: X_1, \ldots, X_m \overset{i.i.d.}{\sim} \text{Poisson}(\lambda)$ (i.e., no change in the reads) vs. $H_1: X_1, \ldots, X_k \overset{i.i.d.}{\sim} \text{Poisson}(\lambda_1)$ and $X_{k+1}, \ldots, X_m \overset{i.i.d.}{\sim} \text{Poisson}(\lambda_2)$, where $1 \le k \le m-1$ (i.e., one change at an unknown position k). By using a variance stabilization transformation proposed in [37], let $\Upsilon_i = \sqrt{X_i + 3/8}$, then Υ_i's approximately follow normal distributions with means $\sqrt{\lambda_i + 1/8}$ and variance $1/4$. Thus, the hypothesis for testing one change point in the window is re-formulated as in (13) and (14).

$$H_0 : \Upsilon_1, \ldots, \Upsilon_k, \Upsilon_{k+1}, \ldots, \Upsilon_m \overset{i.i.d.}{\sim} N\left(\sqrt{\lambda + 1/8}, 1/4\right), \tag{13}$$

$$\begin{aligned} H_1 : \Upsilon_1, \ldots, \Upsilon_k &\overset{i.i.d.}{\sim} N\left(\sqrt{\lambda_1 + 1/8}, 1/4\right) \text{ and} \\ \Upsilon_{k+1}, \ldots, \Upsilon_m &\overset{i.i.d.}{\sim} N\left(\sqrt{\lambda_2 + 1/8}, 1/4\right). \end{aligned} \tag{14}$$

For Bayes method, a constant prior is considered for the change point location parameter k, i.e., $\pi_0(k) = 1/(m-1)$, where $k = 1, \ldots, m-1$. Two different priors are considered for the copy number parameters λ_1 and λ_2: one is the constant prior and the other is the Jeffreys prior.

Specifically, for constant prior, $\pi_0(\lambda_1, \lambda_2|k) \propto$ constant. The posterior probability of k being a change point given observations can be derived as in (15).

$$\pi_1(k|\Upsilon_i's) = \frac{\pi^*(k|\Upsilon_i's)}{\sum_{t=1}^{m-1} \pi^*(t|\Upsilon_i's)} \text{ for } k = 1, \ldots, m-1, \tag{15}$$

where

$$\begin{aligned} \pi^*(k|\Upsilon_i's) \quad &= \exp(-2SS_1)\left\{\frac{\exp(-k(\sqrt{1/8} - \bar{\Upsilon}_1)^2)}{2k}\right. \\ &\quad \left. + \sqrt{\frac{2\pi}{k}}\bar{\Upsilon}_1\left(1 - \Phi\left(\sqrt{\frac{k}{2}} - 2\sqrt{k}\bar{\Upsilon}_1\right)\right)\right\} \\ &\quad \times \exp(-2SS_2)\left\{\frac{\exp(-(m-k)(\sqrt{1/8} - \bar{\Upsilon}_2)^2)}{2(m-k)}\right. \\ &\quad \left. + \sqrt{\frac{2\pi}{m-k}}\bar{\Upsilon}_2\left(1 - \Phi\left(\sqrt{\frac{m-k}{2}} - 2\sqrt{m-k}\bar{\Upsilon}_2\right)\right)\right\}, \end{aligned} \tag{16}$$

$$\begin{aligned}\bar{\Upsilon} &= \frac{1}{k}\sum_{i=1}^{k}\Upsilon, \quad SS_1 = \sum_{i=1}^{k}(\Upsilon - \bar{\Upsilon})^2, \\ \bar{\Upsilon} &= \frac{1}{m-k}\sum_{i=k+1}^{m}\Upsilon \text{ and } SS_2 = \sum_{i=k+1}^{m}(\Upsilon - \bar{\Upsilon})^2.\end{aligned} \tag{17}$$

The Bayesian estimate of the change point location, k, is given by $\hat{k}$ such that the posterior distribution (15) attains its maximum, i.e., $\pi_1(\hat{k}|\Upsilon_i's) = \max_k \pi_1(k|\Upsilon_i's)$. If $\pi_1(\hat{k}|\Upsilon_i's)$ is no less than a pre-defined threshold c, then a change point is declared. Through simulation studies, Ji and Chen [36] suggested to use $c = 0.5$ as the cutoff.

Alternatively, a Jeffreys prior can be assumed for parameter λ's, i.e., $\pi_{J0}(\lambda_1, \lambda_2|k) \propto \sqrt{k}\sqrt{m-k}(\lambda_1 + 1/8)^{-1/2}(\lambda_2 + 1/8)^{-1/2}$. The posterior probability under the Jeffreys prior is

$$\pi_{J1}(k|\Upsilon_i's) = \frac{\pi_J^*(k|\Upsilon_i's)}{\sum_{t=1}^{m-1}\pi_J^*(t|\Upsilon_i's)} \text{ for } k = 1, \ldots, m-1, \tag{18}$$

where

$$\begin{aligned}\pi_J^*(k|\Upsilon_i's) &= \exp(-2SS_1) \times \left(1 - \Phi\left[\sqrt{4k}\left(\sqrt{\frac{1}{8}} - \bar{\Upsilon}_1\right)\right]\right) \\ &\times \exp(-2SS_2) \times \left(1 - \Phi\left[\sqrt{4(n-k)}\left(\sqrt{\frac{1}{8}} - \bar{\Upsilon}_2\right)\right]\right).\end{aligned} \tag{19}$$

The Bayesian estimate of the change point location is the k such that the posterior probability is the largest among all possible locations in a window other than the last position, i.e., $\pi_{J1}(\hat{k}|\Upsilon_i's) = \max_k \pi_{J1}(k|\Upsilon_i's)$, where $k = 1, \ldots, m-1$.

A moving window algorithm is designed for searching multiple change points, where a chromosome or a whole genome of interest is segmented into multiple windows with overlapped markers at the ends of adjacent windows. A change point is searched in each window, and inferences about change points are based on posterior probabilities.

A lung cancer cell line, NCI-H2347, NGS data from the National Center for Biotechnology Information (NCBI) Short Read Archive: SRP000246 (sequence reads) as referenced in [7] was used for illustration. Figure 5 displays change points detected by using this method [36] on chromosome 8 from 8000 to 13,000 kb with cutoff $c = 0.5$. The solid vertical lines in the upper part of the figure indicate the change point locations, while the vertical lines in the lower part of the figures indicate the posterior probabilities.

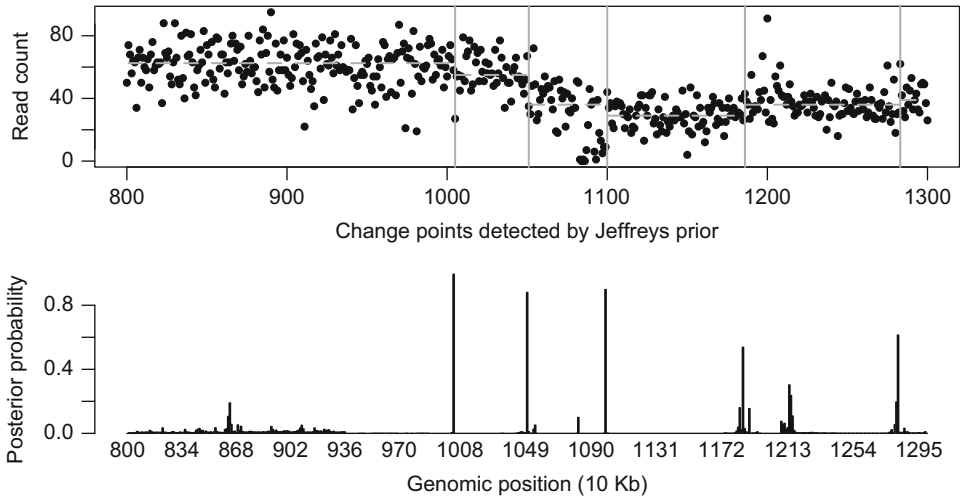


Fig. 5 NCI-H2347 Chromosome 8 (positions from 8000 to 13,000 kb) with identified CNV regions (window size of 130). Solid vertical line segment indicates the change point in the read counts. Reproduced from [3] with permission from John Wiley and Sons

Modeling the NGS Read Count Data Using an On-Line Change Point Model

The NGS read counts were modeled in [38] using a so-called on-line change point model introduced in [39], and they [38] proposed a normal distribution assumption for the normalized and GC content corrected read counts, and then provide the on-line breakpoint detection algorithm to find CNVs in NGS data. This approach is different from both the sliding window and segmentation approaches, and is an effective alternative algorithm in searching CNVs, especially in NGS data. We entail the method hereafter.

Let $\Upsilon_1, \Upsilon_2, \ldots, \Upsilon_n$ be the normalized and GC-corrected read counts (*see* subheading "Modeling the Log Ratio of Read Counts NGS Data" for examples of such read counts data), the inference of the existence of CNVs can be done by searching change points (or breakpoints) such that the read counts are about the same within a pair of breakpoints and are different from the previous pair and the next pair of breakpoints. To phrase this symbolically, we let $0 < \tau_1 < \cdots < \tau_m < n$ be a sequence of unknown breakpoint locations and define $\tau_0 = 0$ and $\tau_{m+1} = n$. We hypothesis that $\Upsilon_1, \Upsilon_2, \ldots, \Upsilon_n$ are homogeneous from $\tau_i + 1$ to τ_{i+1}, with $\Upsilon_{\tau_i+1}, \ldots, \Upsilon_{\tau_{i+1}} \sim N(\mu_i, \sigma^2)$ for $i = 0, 1, \ldots, m$, but heterogeneous otherwise, or simply note that β_i's are not equal, where $\beta_i = (\mu_i, \sigma^2)$. We assume that the breakpoint positions follow a Markov process. This Markov process is determined by a set of transition probabilities,

$$P(\text{nextbreakpointat } w | \text{breakpointat } s), \tag{20}$$

where these transition probabilities depend only on the distance, $w - s$, between the two points s and w. Further, let $g(\cdot)$ be the

probability mass function (and $G(\cdot)$ the cumulative distribution function, CDF) for the distance between the two breakpoints and take $g(\cdot)$ to be the geometric distribution, which implies that there is a fixed probability of a breakpoint at any time point, independent of other breakpoints because of the memory-less property of the geometric distribution. With this modeling idea of the breakpoints for the read count data, we develop an algorithm to calculate the transition probabilities (1) and to estimate the unknown breakpoints, $\tau_1, \ldots, \tau_m$. This process is done by an iterative computing process via a Bayesian framework. The idea is that, we will start with a prior distribution for the unknown parameters, find the posterior distribution at an initial time point s based on all the observations up to time s, then at each new time point w, we update the posterior distribution at the previous time point s to w by making use of the new observations came after time s. Because of the nature of the process described above, this change point detection process is also called an on-line change point detection method [39].

Specifically, we define a state, C_t (or run length), at time t to be the time of the most recent breakpoint before t and let $C_t = 0$ if there were no change points before time t. For example, given $C_t = j$, $j \leq t-1$, then $C_{t+1} = j$ means that there is no breakpoint at time t but $C_{t+1} = t$ means that there is a breakpoint at time t. The state C_t can take values $0, 1, \ldots, t-1$, and $C_1, C_2, \ldots, C_t, \ldots$ is a Markov chain. Taking s in (1) to be t, and w in (1) to be $t+1$, what we need is to calculate the posterior distribution for C_{t+1} given the observations $y_1, \ldots, y_t$. The transition probabilities (20) in this case can be calculated as

$$P(C_{t+1} = j | C_t = i) = \begin{cases} \dfrac{1 - G(t-i)}{1 - G(t-i-1)}, & \text{if } j = i \\ \dfrac{G(t-i) - G(t-i-1)}{1 - G(t-i-1)}, & \text{if } j = t \\ 0, & \text{otherwise} \end{cases}, \tag{21}$$

where $G(\cdot)$ is the CDF of the geometric distribution with parameter θ, that is, $G(t) = 1 - \theta^t$, and θ is the probability of occurrence of a change point. Yiğiter et al. [38] developed an iterative computing process to accomplish the goal set in the statistical inference of a breakpoint (change point) via a Bayesian framework.

Denote $\Upsilon_1, \Upsilon_2, \ldots, \Upsilon_t$ by $\Upsilon_{1:t}$, and assume that $\Upsilon_k \sim N(\mu_k, \sigma_k^2)$ for $k = 0, 1, \ldots, n$. Let the prior distributions of the parameters μ and σ^2 be, respectively,

$$f(\mu|\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{\mu^2}{2\sigma^2}\right\},$$

and

$$f(\sigma^2) \propto \left(\frac{1}{\sigma^2}\right)^{\alpha+1} \exp\left\{-\frac{\gamma}{\sigma^2}\right\},$$

where $\alpha > 0$ and $\gamma > 0$ are constants to be determined by prior experience. Then, the posterior distribution of the jth breakpoint being located at time $t+1$ given the observations $\Upsilon_{1:t+1}$ is

$$P(C_{t+1} = j|\Upsilon_{1:t+1}) \propto \begin{cases} w_{t+1}^{(j)} \dfrac{1 - G(t-j)}{1 - G(t-j-1)} P(C_t = j|\Upsilon_{1:t}) & \text{if } j < t \\ w_{t+1}^{(j)} \sum_{i=0}^{t-1} \dfrac{G(t-i) - G(t-i-1)}{1 - G(t-i-1)} P(C_t = i|\Upsilon_{1:t}) & \text{if } j = t \end{cases}, \tag{22}$$

with the weights $w_{t+1}^{(j)}$ given by

$$w_{t+1}^{(j)} = \begin{cases} \dfrac{\sqrt{t-j+1}}{\sqrt{\pi(t-j+2)}} \dfrac{\Gamma((t-j+2\alpha+1)/2)}{\Gamma((t-j+2\alpha)/2)} \dfrac{(B+2\gamma)^{-(t-j+2\alpha+1)/2}}{(A+2\gamma)^{-(t-j+2\alpha)/2}} & \text{if } j < t \\ (\pi)^{-1/2} 2^{\alpha-1/2} \Gamma\left(\dfrac{2\alpha+1}{2}\right) \left(\dfrac{\Upsilon_{t+1}^2}{2} + 2\gamma\right)^{-(2\alpha+1)/2} & \text{if } j = t \end{cases}, \tag{23}$$

where

$$A = \sum_{i=j+1}^{t} \Upsilon_i^2 - \frac{\left(\sum_{i=j+1}^{t} y_i\right)^2}{t-j+1}$$

and

$$B = \sum_{i=j+1}^{t+1} y_i^2 - \frac{\left(\sum_{i=j+1}^{t+1} y_i\right)^2}{t-j+2}.$$

They [38] presented the following recursive algorithm that helps the searching of multiple breakpoints in the sequence of normalized read counts.

The On-Line CNV Detection Algorithm:

- *Step 1*. Initialization: Start with an artificial starting time of $t = 0$ with $P(C_0 = 0) = 1$ and an arbitrary value for θ, with $\theta = 0.9$, 0.95, or 0.98 being strongly recommended, and a suggested choice of $\alpha = 1$ and $\gamma = 1$.
- *Step 2*. Observe new datum Υ_t according to (i) Right after initialization, t starts from 1, or (ii) t is the first breakpoint identified from the previous cycle.
- *Step 3*. Evaluate the weight $w_{t+1}^{(j)}$ according to Eq. 23.
- *Step 4*. Calculate the transition probabilities $P(C_{t+1} = j|C_t = i)$ according to Eq. 21.

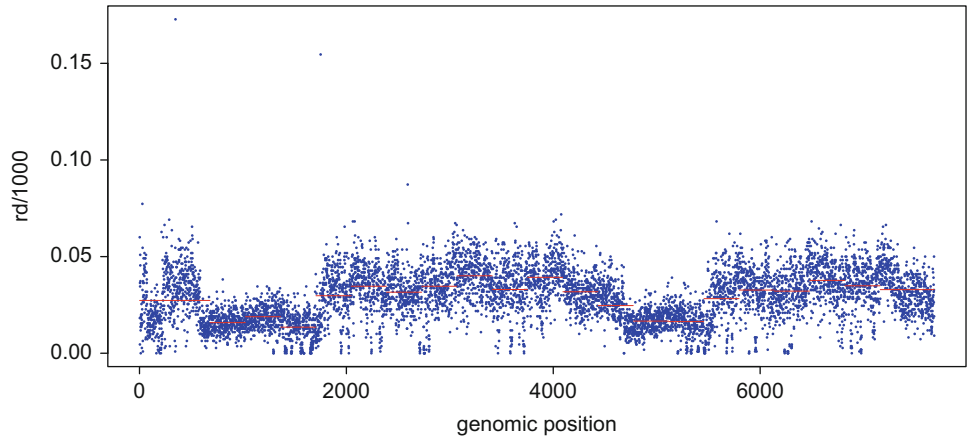


Fig. 6 NCI-H2347 Chromosome 17 with identified CNV regions. Solid horizontal line segment indicates the average read counts in each segment. Reproduced from [38] with permission from Taylor & Francis

- *Step 5*. Calculate the breakpoint posterior probabilities, $P(C_{t+1} = j | \Upsilon_{1:t+1})$, according to Eq. 22 for all $j \leq t$. If the maximum of these posterior probability is attained at j_1 and is greater than a threshold p_0, then $\tau_1 = j_1$ and go to Step 6. Otherwise, stop. There are no more breakpoints.
- *Step 6*. Return to step 2 (ii)—Step 5 with new $t = j_1$ until no further breakpoints are identified. The collection of j_1, $j_2, \ldots, j_m$ will be the final estimates of the CNV loci.

The above described algorithm [38] was applied to the normalized read counts data (divided by 1000, to avoid exponentiation of large number) of the lung cancer cell line, NCI-H2347, with $\alpha = 1$ and $\gamma = 1$. Neighboring segments are merged if the beginning and ending breakpoints of them are adjacent to each other. The following Fig. 6 shows chromosome 17 with breakpoints identified by using a threshold of 0.50 for the posterior probabilities (that actual maximum posterior probability was greater than 0.75) with the red horizontal lines indicating the segment means of different segments, separated by the identified breakpoints.

4.4 Penalized Regression Approach

4.4.1 For Single Subject Profile

As pointed out in [40], biologically it was thought that neighboring genomic regions may contain similar copy numbers. Several studies have considered this biologically structural dependency of copy numbers and applied penalized regression models based on the Least Absolute Shrinkage and Selection Operator (LASSO, [41]) approach to CNVs detection. In [40], an approach was proposed to incorporate a multiple change-point model with a fussed LASSO [42] and a modified information criterion to identify multiple CNV loci in the NGS ratio of read counts data. This approach avoids the drawbacks of the segmentation and sliding

window approaches and is effective in identifying multiple CNV regions in real applications and in simulation studies.

The fused LASSO method is suitable to find the multiple change points because when the tuning parameter λ changes in the range of $[0, \infty]$, the fused LASSO solution path is able to estimate the best piecewise solution according to the number change points. Since the publication of the seminal paper of [41] on using l-1 norm penalty to least squares regression, many researchers have developed algorithms to solve the LASSO problem. The generalized LASSO approach is one of them which is particularly useful for change-point detection.

Specifically, let $\Upsilon \in \mathbb{R}^{n+1}$ be a response vector and $\boldsymbol{\beta} \in \mathbb{R}^{n+1}$ be a vector of regression coefficients. Each observation y_i corresponds to the coefficient β_i, $i = 1, \ldots, n$. The 1d fused LASSO problem [43] is then written as

$$\min_{\boldsymbol{\beta} \in \mathbb{R}^n} \frac{1}{2} ||\mathbf{y} - \boldsymbol{\beta}||_2^2 + \lambda ||\mathbf{D}\boldsymbol{\beta}||_1, \tag{24}$$

where $\lambda \geq 0$ is a tuning parameter, and D is a $(n-1) \times n$ penalty matrix with

$$\mathbf{D} = \begin{bmatrix} -1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & -1 & 1 & \cdots & 0 & 0 \\ & & & \cdots & & \\ 0 & 0 & 0 & \ldots & -1 & 1 \end{bmatrix}.$$

$||\cdot||_2^2$ and $\|\cdot\|_1$ denote the l-2 are l-1 norms, respectively.

In CNV studies, copy numbers between neighboring genomic regions have a similar inherent structure along a chromosome. Thus, the penalty term $||\mathbf{D}\boldsymbol{\beta}||_1$ is desirable in CNV detection because it encourages the differences between neighboring coefficients to be small. In addition, by using the l-1 norm, when the estimated values of neighboring coefficients are close to each other, the penalty term makes them to be exactly equal. Consequently, the solution, $\hat{\boldsymbol{\beta}} = (\hat{\beta}_1, \hat{\beta}_2, \ldots, \hat{\beta}_n)^T$, appears a piecewise constant shape. Then, the 1d fused LASSO solution can capture the exact locations of change points, which is suitable in the multiple change point detection problem. To estimate $\boldsymbol{\beta}$, we use the dual path algorithm developed by [43]. The path provides the $n-1$ solutions according to $n-1$ candidate λ values. In terms of change point problems, the path identifies the change point locations according to the number of change points. The generalized LASSO dual path algorithm is implemented in the R package *genLASSO* on the CRAN website. In the true underlying model, only small number of regions may have CNVs among thousands of genomic locations. The sparsity in the 1d fused LASSO depends on the tuning parameter λ selected. As λ increases, the $\|\mathbf{D}\boldsymbol{\beta}\|_1$ encourages neighboring coefficients to be the

same, leading a small number of change points (i.e., a sparse model); as λ decreases, neighboring coefficients tend to be unequal, leading a large number of change points (i.e., a dense model). The l-1norm penalty term in the 1d fused LASSO actually forces sparsity of the differences between neighboring coefficients.

Given the number of copy number changes, their locations are then estimated by fitting the 1d fused LASSO using the dual path algorithm proposed by Tibshirani and Taylor [43]. However, the main challenge is to determine the number of change points. Although several scholars have applied LASSO-based methods to the CNV detection problem, they have not yet intensely focused on how to determine the number of copy number changes. In [40], a modified information criterion, JMIC, was introduced to help determine the number of CNVs by selecting the optimal λ among $n-1$ candidate λ values. JMIC is defined as

$$\mathrm{JMIC}(\lambda_K) = -2\ln L(\hat{\theta}, \hat{\mathbf{t}}_K, K) + d(K+1)^{\gamma} n^{\alpha}, \tag{25}$$

where $1<\gamma<2$, and $0<\alpha<1$. Note that if let $n^{\alpha}=\rho_n n$ with $\rho_n = n^{\alpha-1}$, the consistency of JMIC can be established following similar work in [44]. Using JMIC, given by Eq. 25, to select the tuning parameter in the 1d fused LASSO, we can estimate $\hat{\mathbf{t}}_K$ by the dual path algorithm along with the 1d fused LASSO at $n-1$ critical λ values. Then, based on the principle of minimum information criterion, we select the optimal $\hat{\lambda}_{\hat{K}}$, where $\hat{K} = arg\left\{\min_{0\le K\le n-2} \mathrm{JMIC}(\lambda_K)\right\}$. The implementation of the approach was given in [40].

The 1d fused LASSO approach with JMIC for tuning parameter selection was applied to the log ratio of NGS reads data of HCC1954 and its matched normal cell line BL 1954. The following Fig. 7 illustrates the result for Chromosome 5 [40].

4.4.2 For Multiple Subject Profiles

When there are multiple patients' or subjects' DNA-sequencing data available, researchers often desire to find the respective common CNV regions shared by different groups of subjects and simultaneously find distinct CNV regions that uniquely appear in one of the subjects. To answer this type of questions, recently, the framework of a fused Lasso latent feature model [45] was proposed in [46] to help detect boundaries of CNV regions using the DNA-sequencing data from multiple subject samples. Let $y_{\ell s}$ be the normalized sequencing read count (normalized to copy number 2, *see* [29]) at genomic location $\ell(\ell = 1,\ldots,\mathcal{L})$ for subject s, $s=1,\ldots,S$. The latent feature model was proposed to represent $y_{\ell s}$:

$$Y_{\ell s} = \sum_{j=1}^{J} \beta_{\ell j}\theta_{js} + \varepsilon_{\ell s}, \tag{26}$$

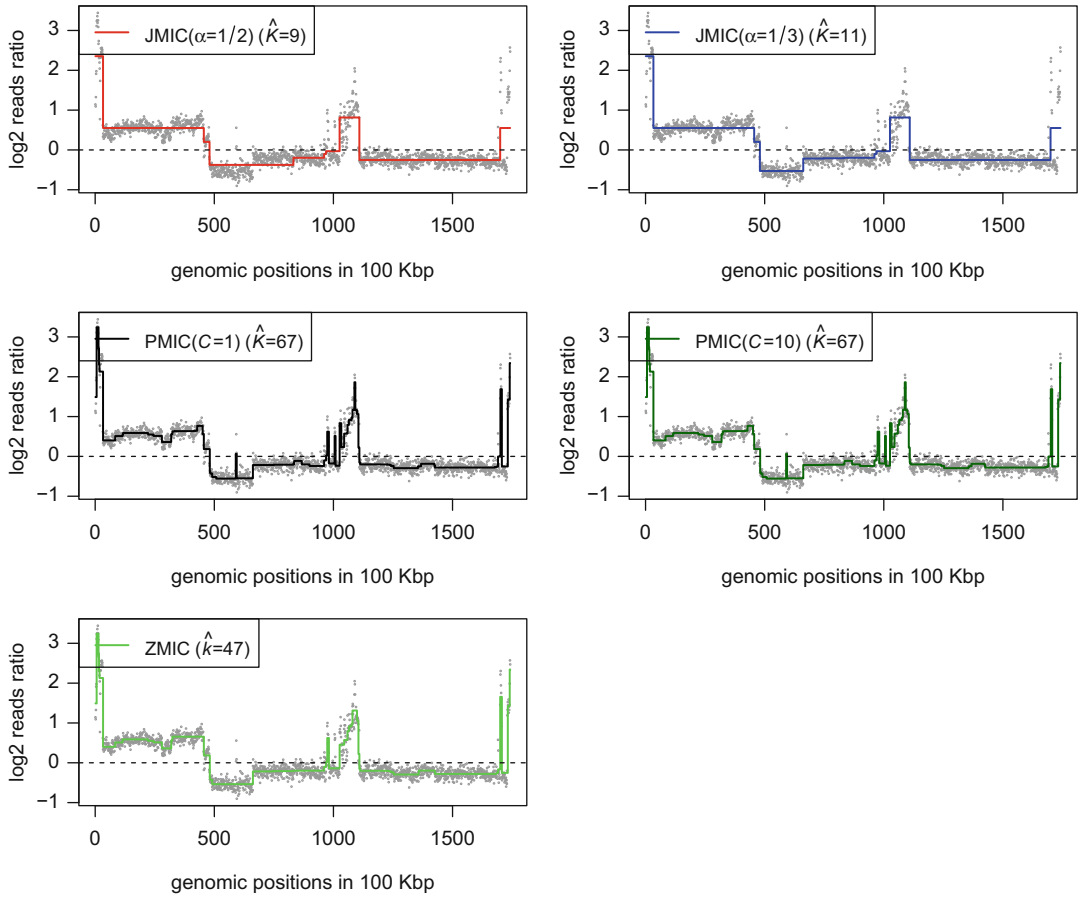


Fig. 7 Chromosome 5 with the identified change points by the 1d fused LASSO models based on JMIC($\alpha = 1/2$, 1/3), PMIC($C = 1$, 10), ZMIC. $\hat{K}$ denotes the estimated number of change points. The solid line represents the average reads ratio in each segment. The dashed line is for $\hat{y} = 0$, indicating no copy number change. Reproduced from [40] with permission from

where $(\beta_{1j}, \ldots, \beta_{\mathcal{L}j})' \triangleq \boldsymbol{\beta}_{\cdot j}$, for $j = 1, \ldots, J$, represent the J latent features, $(\theta_{1s}, \ldots, \theta_{Js})' \triangleq \boldsymbol{\theta}_{\cdot s}$ represents the weights on the features for sample s, $s = 1, \ldots, S$, and $(\varepsilon_{1s}, \ldots, \varepsilon_{\mathcal{L}s})' \triangleq \boldsymbol{\varepsilon}_{\cdot s}$ is the random error term for sample s, whose components are correlated with each other.

The above Model (26) can also be expressed in the following matrix form,

$$\boldsymbol{\Upsilon} = \boldsymbol{B}\boldsymbol{\Theta} + \boldsymbol{E}, \tag{27}$$

where $\boldsymbol{\Upsilon}_{\mathcal{L}\times S} = (y_{\ell s})$, $\boldsymbol{B}_{\mathcal{L}\times J} = (\beta_{\ell j})$, $\boldsymbol{\Theta}_{J\times S} = (\theta_{js})$ with the L_2 restriction being $\sum_{s=1}^{S} \theta_{js}^2 \le 1$, for each j, and $\boldsymbol{E}_{\mathcal{L}\times S} = (\varepsilon_{\ell s})$. Specifically, $\boldsymbol{y}_{\cdot s} = (y_{1s}, \ldots, y_{\mathcal{L}s})'$ represents the vector consisting of the normalized sequencing read counts for a given sample s on the genome (or a specific chromosome). The idea is to view normalized read counts in each sample as a weighted linear combination of the

J latent features in the presence of some random noises, while each feature represents a particular pattern of CNV and the weights, θ's, for a given sample determine how much each feature contributes to that sample.

Notice that CNV regions appear within a neighborhood on the genome and the probe locations within a region tend to have the common copy number, while outside of CNV regions, such normalized copy number should be zero.

Therefore, β_{lj} and θ_{js} in model (27) can be estimated by using the fused LASSO signal approximator (FLSA) [47], or by minimizing the following function:

$$F(\boldsymbol{B},\boldsymbol{\Theta}) = \sum_{s=1}^{S}\sum_{\ell=1}^{\mathcal{L}}\left(y_{\ell s} - \sum_{j=1}^{J}\beta_{lj}\theta_{js}\right)^2 + \sum_{j=1}^{J} P_{\lambda_1,\lambda_2}(\boldsymbol{\beta}_{\cdot j}), \tag{28}$$

where $P_{\lambda_1,\lambda_2}(\boldsymbol{\beta}_{\cdot j}) = \lambda_1\sum_{\ell=1}^{\mathcal{L}}|\beta_{\ell j}| + \lambda_2\sum_{\ell=1}^{\mathcal{L}}|\beta_{\ell j} - \beta_{\ell-1,j}|$ is the penalty function, with the first penalty term (with parameter λ_1) encouraging sparsity and the second term (with parameter λ_2) encouraging smoothness, and $\sum_{s=1}^{S}\sum_{\ell=1}^{\mathcal{L}}\left(y_{\ell s} - \sum_{j=1}^{J}\beta_{lj}\theta_{js}\right)^2 \triangleq \boldsymbol{\Upsilon} - \boldsymbol{B\Theta}_F^2$ is the usual sum of squared errors.

As in all versions of LASSO type of computing, the estimation of the model parameters and the selection of tuning parameters are the essential elements for achieving stable estimates. Specifically, the following modified information criterion, $\mathrm{MBIC}_1(\gamma)$, was proposed in [46] to select the tuning parameters λ_1 and λ_2, in the forms of α $(0<\alpha<1)$ and λ_0 (where $\lambda_1 = \alpha\lambda_0$, $\lambda_2 = (1-\alpha)\lambda_0$),

$$\begin{aligned}\mathrm{MBIC}_1(\gamma) \triangleq\ & (S\mathcal{L})\cdot\log\left(\frac{\|\boldsymbol{\Upsilon} - \hat{\boldsymbol{B}}\hat{\boldsymbol{\Theta}}\|_F^2}{S\mathcal{L}}\right) \\ & +(1 + k_{\alpha,\lambda_0}(\hat{\boldsymbol{B}}) + k_{\alpha,\lambda_0}(\hat{\boldsymbol{\Theta}}))^{\gamma}\log(S\mathcal{L}),\end{aligned} \tag{29}$$

where $\gamma \geq 1$ is a constant, $k_{\alpha,\lambda_0}(\hat{\boldsymbol{B}}) = \sum_{j=1}^{J} k_{\alpha,\lambda_0}(\hat{\boldsymbol{\beta}}_{\cdot j})$ with $k_{\alpha,\lambda_0}(\hat{\boldsymbol{\beta}}_{\cdot j})$ being the number of unique nonzero elements in the jth estimated feature, $\hat{\boldsymbol{\beta}}_{\cdot j}$, and $k_{\alpha,\lambda_0}(\hat{\boldsymbol{\Theta}}) = \sum_{j=1}^{J} k_{\alpha,\lambda_0}(\hat{\boldsymbol{\theta}}_{j\cdot})$ with $k_{\alpha,\lambda_0}(\hat{\boldsymbol{\theta}}_{j\cdot})$ being the number of unique nonzero elements in the weights of jth estimated feature, $\hat{\boldsymbol{\theta}}_{j\cdot}$.

The estimate of J is needed and the choice of J, the number of features, and the choice of tuning parameters λ_1 and λ_2 mutually influence each other. Numerical experiments [46] suggest that a plausible way to choose J is to use a semiautomatic process, suggested by Nowak et al. [45] based on the percentage of variation explained (PVE), defined as

$$\mathrm{PVE}(J^*) = 1 - \frac{\sum_{s=1}^{S}\sum_{\ell=1}^{\mathcal{L}}\left(y_{\ell s} - \sum_{j=1}^{J^*}\hat{\beta}_{\ell j}\hat{\theta}_{js}\right)^2}{\sum_{s=1}^{S}\sum_{\ell=1}^{\mathcal{L}}(y_s - \bar{y}_{\cdot s})^2} \tag{30}$$

for given J^*, where $\bar{y}_{.s} = \sum_{\ell=1}^{\mathcal{L}} y_{\ell s}/\mathcal{L}$, and $\hat{\beta}_{lj}$ and $\hat{\theta}_{js}$ are the estimates obtained from the aforementioned estimation procedure with $\text{MBIC}_1(\gamma)$ criterion. For the potential values of J, $\{1, \ldots, S\}$, the PVEs can be obtained. J is then chosen as the plateau point of the PVEs, that is, any additional features will not significantly improve the estimated fit or increase the PVEs. With the found plateau point J, the optimal tuning parameters along with the estimated values for β_{lj}s and θ_{js}s can be obtained.

Implementation Steps of the Method: The process of detecting CNVs for multiple subjects can be implemented through the following three steps:

- *Step 1.* **Optimizing the choice of** J: Let J be $1, \ldots, S$, respectively. For each J, the optimized tuning parameters $\hat{\alpha}$ and $\hat{\lambda}_0$, thus $\hat{\lambda}_1$ and $\hat{\lambda}_2$, along with $\hat{\boldsymbol{\beta}}$ and $\hat{\boldsymbol{\theta}}$ are obtained from the FLLat method with $\text{MBIC}_1(\gamma)$ for $\gamma \in \{1, 1.1, \ldots, 1.9, 2.0\}$. For each γ, $\text{PVE}(J)$ is calculated for each J by using the parameter estimates obtained above and the plateau point $J_0(\gamma)$ for PVEs is found. Declare $J_0(\gamma)$ as the optimal value of J for this γ.
- *Step 2.* **Final Parameter estimation**: Let γ_0 be the one such that the optimal $J_0(\gamma)$ obtained from the first step gives the closest value to the true value of J, with simultaneous products of $\hat{\lambda}_1$, $\hat{\lambda}_2$, $\hat{\boldsymbol{\beta}}$s, and $\hat{\boldsymbol{\theta}}$s.
- *Step 3.* **Visualization and common/individual CNV region detection**: Use the $\hat{\boldsymbol{\beta}}$s and $\hat{\boldsymbol{\theta}}$s obtained in Step 2 to obtain the predicted $\hat{y}_{\ell s} = \sum_{j=1}^{J_0} \hat{\beta}_{\ell j} \hat{\theta}_{js}$. Then input gains and losses based on $\hat{y}_{\ell s}$ to the STAC analysis to obtain visualization of the detected common/individual CNV regions and the endpoints of the regions.

As an example, the above implementation process was applied [46] to the analysis of 18 subjects, one male (M) and one female (F), from each of the nine ethnic group, whose sequencing reads are downloaded from the **1000 Genome Project Consortium** [48]. These subjects are abbreviated as: GBR—British in England and Scotland, FIN—Finnish in Finland, CHS—Southern Han Chinese, PUR—Puerto Ricans from Puerto Rico, CEU—Utah residents with Northern and Western European Ancestry, YRI—Yoruba in Ibadan, Nigeria, CHB—Han Chinese in Beijing, China, JPT—Japanese in Tokyo, Japan, and TSI—Toscani in Italia with these specific subjects IDs: HG00125-GBR-F, HG00136-GBR-M, HG00276-FIN-F, HG00280-FIN-M, HG00449-CHS-F, HG00463-CHS-M, HG00638-PUR-F, HG00637-PUR-M, NA12878-CEU-F, NA12043-CEU-M, NA18507-YRI-M, NA18511-YRI-F, NA18561-CHB-M, NA18570-CHB-F, NA18948-JPT-M, NA18956-JPT-F, NA20755-TSI-M, and

NA20756-TSI-F. For this example, we did not consider possible difference due to gender.

Let $\tilde{y}_{\ell s}$ represent the GC-corrected read count values for individual s at the ℓ-th genome location. Let $\mu_{\cdot s} = \frac{1}{\mathcal{L}}\sum_{\ell=1}^{\mathcal{L}} \tilde{y}_{\ell s}$ denote the whole genome mean for the copy number read counts for individual s. Then the copy number read count $\tilde{y}_{\ell s}$ can be normalized to copy number 2 by using $y_{\ell s} = \tilde{y}_{\ell s}/\mu_{\cdot s}$ for $s = 1, \ldots, S, \ell = 1, \ldots, \mathcal{L}$. These $y_{\ell s}$ values are the input data values in our analysis.

For the genomic positions from 150,401 to 150,900 kbp ($\mathcal{L} = 500$) on chromosome 1, the predicted values $\hat{y}_{\ell s}$ of $y_{\ell s}$ are obtained by the FLLat method with $\text{MBIC}_1(1.2)$. Let $\hat{\mu}_{\cdot s} = \frac{1}{\mathcal{L}}\sum_{\ell=1}^{\mathcal{L}} \hat{y}_{\ell s}$ for $s = 1, \ldots, S$, and $\hat{\sigma}_{\cdot s} = \sqrt{\frac{1}{\mathcal{L}-1}\sum_{\ell=1}^{\mathcal{L}} (\hat{y}_{\ell s} - \hat{\mu}_{\cdot s})^2}$. To visualize the profile patterns with the proposed estimate of J, the Significance Testing of Aberrant Copy number (STAC) [50] software was applied to the results. To do the STAC analysis, for regions with copy number gains, we set the probe location ℓ for sample s as 1 if $\hat{y}_{\ell s} - \hat{\mu}_{\cdot s} > 1.5\hat{\sigma}_{\cdot s}$ and as 0 otherwise. Similarly, for copy number losses, we set the probe location ℓ for sample s as 1 if $\hat{y}_{\ell s} - \hat{\mu}_{\cdot s} < -1.5\hat{\sigma}_{\cdot s}$ and as 0 otherwise. As an illustration, the STAC analysis results for gains (top) and losses (bottom) on the chromosome position from 150,401 to 150,900 kbp of chromosome 1 with the frequency statistic and the cutoff being 0.95 are given in Fig. 8.

The 1000 Genomes Project Consortium [48] aimed to discover human genome variation for different populations using

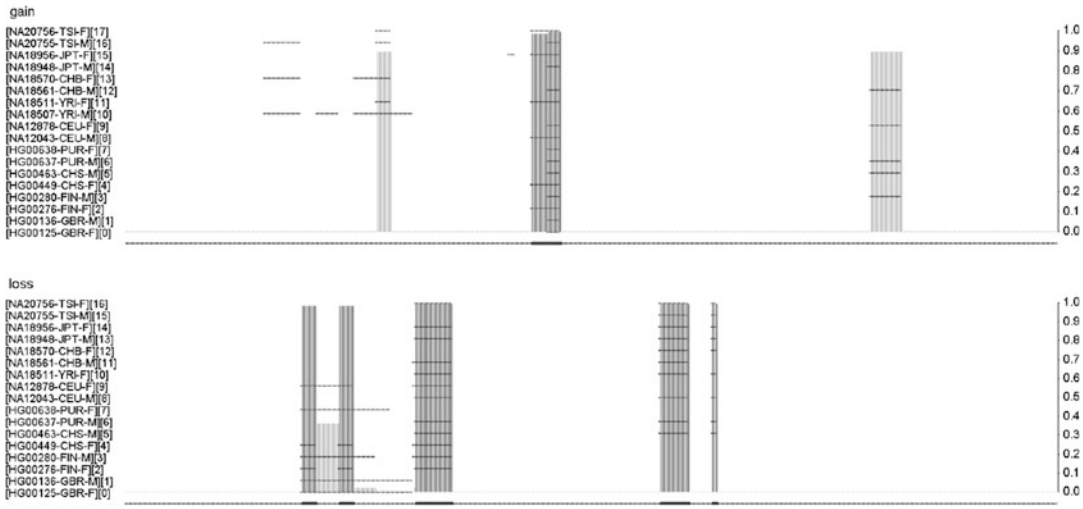


Fig. 8 The STAC analysis for gains (top) and losses (bottom) of copy numbers from position 150,401 to 150,900 kbp of chromosome 1 with the frequency statistic and the cutoff being 0.95. The 1-kb locations are plotted along the *x*-axis, and each sample having at least one interval of gain/loss along the chromosome is plotted on the *y*-axis. The gray bars track the maximum STAC confidence ($1 - P$-value), darker bars are those with confidence > 0.95. Reproduced from [46] with permission from Mary Ann Liebert, Inc. Publishers

sequencing technology, especially, NGS, and continued to provide NGS data of 2504 individuals from 26 populations worldwide [49]. The distribution of human genetic variations, pictured by the Consortium, sheds much light on studying human susceptibility to diseases due to their CNVs in common or different forms relevant to their ethnic identity. The Consortium data provide a rich library for population health data analytics. The sequencing data from the Consortium are certainly high dimensional and analyzing them requires a good statistical and computational approach. In this example, we illustrated how to use the sequencing reads data to infer common CNV regions shared by multiple subjects and in the meantime detect individual CNV regions, when the data have correlated structure.

5 Conclusions

To conclude this chapter, we provide Table 2 to the readers for a quick summary of the approaches for detecting CNVs using NGS data so that an appropriate method/algorithm will be used based on the NGS data one has in hand. As a rule of thumb, knowing the data and the feasibility of a particular model/method/algorithm in getting the most interpretable results from the data should be kept in the minds of data analysts. There is no one-size-fits-all type of method for analysis NGS data and knowing how the data were gathered is always a must to do work for statisticians.

We would like to point out that using read-depth information only for CNV detection makes it difficult to find the exact breakpoints [51]. Several researches have indicated that integrating multiple information sources together, such as using pair-end and split-read information as priors for Bayesian analysis on read-depth data, can improve the accuracy in finding breakpoints. Tools incorporating multiple information sources, for example, Lumpy [52] and Hydra-Multi [53], have shown improved CNV detection results. In addition, large consortia, such as the 1000 Genome Project, are combining different methods together to obtain final results.

Another unsolved issue of current pipelines is to genotype copy numbers for complex and multi allelic regions after CNVs are identified. Though some pipelines, such as CNVnator [54] and cn.MOPS [55], incorporate the genotyping step, it is challenging when the copy number is large, for example, larger than four. Recent developments have begun to focus on this issue. For example, GenomeStrip [56] detects CNVs and genotypes in simple and complex regions; CNVrd2 [57] genotypes multiallelic regions associated with complex diseases. In the future, if more sophisticated statistical methods can be developed for genotyping complex and multiallelic regions, then the association between complex diseases and CNVs at these regions can be better answered [58, 59].

Table 2
Summary of Statistical and Computational Approaches for CNV Detection using NGS Data. There are three types of data input: (1) one sample that contains the corrected target sample data, (2) two samples that contain one control and one target sample, and (3) multiple samples. Methods are sorted by the number of samples in the data input and the year the work was published. Algorithms described in this chapter with details are marked by *

Tool	Reference	Model/Algorithm	Data type
mrCaNaVar	[60]	Detect large CNVs	One sample
ReadDepth*	[29]	Circular Binary Segmentation	One sample
CNVnator	[54]	Mean-shift approach	One sample
WaveCNV	[61]	Wavelet transformation	One sample
Ji and Chen*	[36]	Poisson change point model	One sample
Yiğiter et al.*	[38]	On-line change point model	One Sample
Lee and Chen*	[40]	1d fused LASSO	One Sample
FREEC	[62]	Moving window algorithm	One/Two samples
SegSeq*	[7]	Log-ratio test	Two samples
CNV-seq	[63]	Observed ratio difference	Two samples
CNAseg*	[17]	Hidden Markov Model	Two samples
rSW-seq*	[8]	Dynamic programming	Two samples
BIC-seq	[64]	Bayesian information criterion	Two sample
SeqCBS*	[30]	Poisson change point model	Two Samples
CNVnorm	[65]	Cell comparison	Two Samples
SeqBBS*	[34]	Normal change point model	Two samples
m-HMM*	[18]	Hidden Markov Model	Two samples
RDXplorer	[66]	Event-wise test algorithm	Multiple samples
CMDS	[67]	Correlation analysis	Multiple samples
JointSLM*	[19]	Shifting level model	Multiple samples
cn.MOPS	[55]	Mixture Poisson model	Multiple samples
CNVeM	[68]	EM algorithm	Multiple samples
CNV-CH	[69]	Convex hull segmentation	Multiple samples
Chen and Deng*	[46]	Panelized regression model	Multiple samples

References

1. Redon R, Ishiwaka S, Fitch KR, Feuk L, Perry GH, Andrews D, Fiegler H, Shapero MH, Carson AR, Chen W, Cho EK, Dallaire S, Freeman JL, Gonzalez JR, Gratacos M, Huang J, Kalaitzopoulos D, Komura D, MacDonald JR, Marshall CR, Mei R, Montgomery L, Nishimura K, Okamura K, Shen F, Somerville MJ, Tchinda J, Valsesia A, Woodwark C, Yang F, Zhang J, Zerjal T, Zhang J, Armengol L, Conrad DF, Estivill X, Tyler-Smith C, Carter NP, Aburatani H, Lee C, Jones KW, Scherer SW, Hurles ME (2006) Global variation in copy number in the human genome. Nature 444:444–454
2. Stranger B, Forrest M, Dunning M, Ingle C, Beazley C, Thorne N, Redon R, Bird C, de Grassi A, Lee C, Tyler-Smith C, Carter N, Scherer SW, Tavaré S, Deloukas P, Hurles ME, Dermitzakis ET (2007) Relative impact of nucleotide and copy number variation on gene expression phenotypes. Science 315:848
3. Ji T, Chen J (2016) Statistical methods for DNA copy number variation detection using the next generation sequencing data. Aust N Z J Stat 58:473–491
4. Langmead B, Trapnell C, Pop M, Salzberg SL (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome Biol 10:R25
5. Cheung MS, Down TA, Latorre I, Ahringer J (2011) Systematic bias in high-throughput sequencing data and its correction by BEADS. Nucleic Acids Res 39:e103
6. Benjamini Y, Speed T (2011) Estimation and correction for GC-content bias in high throughput sequencing. Technical Report 804, Department of Statistics, University of California, Berkeley
7. Chiang DY, Getz G, Jaffe DB, O'Kelly MJ, Zhao X, Carter SL, Russ C, Nusbaum C, Meyerson M, Lander ES (2009) High-resolution mapping of copy-number alterations with massively parallel sequencing. Nat Methods 6:99–103
8. Kim TM, Luquette LJ, Xi R, Park PJ (2010) rSW-seq: algorithm for detection of copy number alterations in deep sequencing data. BMC Bioinf 11(432):1471–2105
9. Smith TF, Waterman MS (1981) Identification of common molecular subsequences. J Mol Biol 147:195–197
10. Price TS, Regan R, Mott R, Hedman A, Honey B, Daniels RJ, Smith L, Greenfield A, Tiganescu A, Buckle V, Ventress N, Ayyub H, Salhan A, Pedraza-Diaz S, Broxholme J, Ragoussis J, Higgs DR, Flint J, Knight SJ (2005) SW-ARRAY: a dynamic programming solution for the identification of copy-number changes in genomic DNA using array comparative genome hybridization data. Nucleic Acids Res 33(11):3455–3464
11. Baum LE, Petrie T (1966) Statistical inference for probabilistic functions of finite state Markov chains. Ann Math Stat 37(6):1554–1563
12. Baum LE, Eagon JA (1967) An inequality with applications to statistical estimation for probabilistic functions of a Markov process and to a model for ecology. Bull Am Math Soc 73 (3):360–363
13. Baum LE, Petrie T, Soules G, Weiss N (1970) A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. Ann Math Stat 41(1):164–171
14. Baum LE (1972) An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes. In: Shisha O (ed) Proceedings of the third symposium on inequalities. Academic, New York, pp 1–8
15. Guha S, Li Y, Neuberg D (2008) Bayesian hidden Markov modeling of array CGH Data. J Am Stat Assoc 103:485–497
16. Marioni JC, Thorne NP, Tavare S (2006) BioHMM: a heterogeneous Hidden Markov model for segmenting array CGH data. Bioinformatics 22:1144–1146
17. Ivakhno S, Royce T, Cox AJ, Evers DJ, Cheetham RK, Tavare S (2010) CNAseg – a novel framework for identification of copy number changes in cancer from second-generation sequencing data. Bioinformatics 26:3051–3058
18. Wang H, Nettleton D, Ying K (2014) Copy number variation detection using next generation sequencing read counts. BMC Bioinf 15:109
19. Magi A, Bnelli M, Yoon S, Roviello F, Torricelli F (2011) Detecting common copy number variants in high-throughput sequencing data by using JointSLM algorithm. Nucleic Acids Res 39:e65
20. Magi A, Benelli M, Marseglia G, Nannetti G, Scordo MR, Torricelli F (2010) A shifting level model algorithm that identifies aberrations in array-CGH data. Biostatistics 11:265–280
21. Shaban SA (1980) Change-point problem and two phase regression: an annotated bibliography. Int Stat Rev 48:83–93

22. Basseville M (1988) Detecting changes in signals and systems – a survey. Automatica 24:309–326
23. Chen J, Gupta AK (2012) Parametric statistical change point analysis - with applications to genetics, medicine, and finance, 2nd edn. Birkhauser, New York
24. Olshen AB, Venkatraman ES, Lucito R, Wigler M (2004) Circular binary segmentation for the analysis of array-based DNA copy number data. Biostatistics 5(4):557–572
25. Venkatraman ES, Olshen AB (2007) A faster circular binary segmentation algorithm for the analysis of array CGH data. Bioinformatics 23:657–663
26. Siegmund DO (1988) Approximate tail probabilities for the maxima of some random fields. Ann Probab 16:487–501
27. Yao Q (1989) Large deviations for boundary crossing probabilities of some random fields. J Math Res Expo 9:181–192
28. Yao Q (1993) Tests for change-points with epidemic alternatives. Biometrika 80:179–191
29. Miller CA, Hampton O, Coarfa C, Milosavljevic A (2011) ReadDepth: a parallel R package for detecting copy number alterations from short sequencing reads. PLoS One 6(1): e16327
30. Shen JJ, Zhang NR (2012) Change-point model on nonhomogeneous Poisson process with application in copy number profiling by next-generation DNA sequencing. Ann Appl Stat 6(2):476–496
31. Rabinowitz D (1994) Detecting clusters in disease incidence. In: Change-point problems (South Hadley, MA, 1992). Institute of Mathematical Statistics Lecture Notes–Monograph Series, vol 23. IMS, Hayward, pp 255–275
32. Zhang NR, Siegmund DO (2007) A modified Bayes information criterion with applications to the analysis of comparative genomic hybridization data. Biometrics 63:22–32
33. Schwarz G (1978) Estimating the dimension of a model. Ann Stat 6:461–464
34. Li H, Vallandingham J, Chen J (2013) SeqBBS: a change-point model based algorithm and R package for searching CNV regions via the ratio of sequencing reads. In: Proceedings of the 2013 IEEE international workshop on genomic signal processing and statistics, pp 46–49
35. Chen J, Yiğiter A, Chang KC (2011) A Bayesian approach to inference about a change point model with application to DNA copy number experimental data. J Appl Stat 38:1899–1913
36. Ji T, Chen J (2015) Modeling the next generation sequencing read count data for DNA copy number variant study. Stat Appl Genet Mol Biol 14:361–374
37. Anscombe FJ (1948) The transformation of Poisson, binomial and negative-binomial data. Biometrika 35:246–254
38. Yiğiter A, Chen J, Lingling An L, Danacioğlu N (2015) An on-line CNV detection method for short sequencing reads. J Appl Stat 42 (7):1556–1571
39. Fearnhead P, Liu Z (2007) On-line inference for multiple changepoint problems. J R Stat Soc B 69:589–605
40. Lee J, Chen J (2019) A penalized regression approach for DNA copy number study using the sequencing data. Stat Appl Genet Mol Biol 18(4). https://doi.org/10.1515/sagmb-2018-0001
41. Tibshirani RJ (1996) Regression shrinkage and selection via the LASSO. J R Stat Soc Ser B 58 (1):267–288
42. Tibshirani R et al (2005) Sparsity and smoothness via the fused LASSO. J R Stat Soc Ser B (Stat Methodol) 67:91–108
43. Tibshirani RJ, Taylor J (2011) The solution path of the generalized LASSO. Ann Stat 39:1335–1371
44. Qian J, Su L (2016) Shrinkage estimation of regression models with multiple structural changes. Economet Theory 32(6):1376–1433
45. Nowak G, Hastie T, Pollack JR, Tibshirani R (2011) A fused lasso latent feature model for analyzing multi-sample aCGH data. Biostatistics 12(4):776–791
46. Chen J, Deng S (2018) Detection of copy number variation regions using the DNA-sequencing data from multiple profiles with correlated structure. J Comput Biol 25:1128–1140
47. Tibshirani R, Saunders M, Rosset S, Zhu J, Knight K (2005) Sparsity and smoothness via the fused lasso. J R Stat Soc Ser B (Stat Methodol) 67:91–108
48. The 1000 Genomes Project Consortium (2010) A map of human genome variation from population scale sequencing. Nature 467 (7319):1061–1073
49. The 1000 Genomes Project Consortium (2015) A global reference for human genetic variation. Nature 526(7571):68–74
50. Diskin SJ, Eck T, Greshock J, Mosse YP, Naylor T, Stoeckert CJ, Weberm BL, Maris JM, Grant GR (2006) STAC: a method for testing the significance of DNA copy number aberrations across multiple array-CGH experiments. Genome Res 16(9):1149–1158
51. Zhao M, Wang Q, Wang Q, Jia P, Zhao Z (2012) Computational tools for copy number variation (CNV) detection using next-

generation sequencing data: features and perspectives. BMC Bioinf 14(Suppl 11):S1

52. Layer RM, Chiang C, Quinlan AR, Hall IM (2014) LUMPY: a probabilistic framework for structural variant discovery. Genome Biol 15:R84
53. Lindberg MR, Hall IM, Quinlan AR (2015) Population-based structural variation discovery with Hydra-Multi. Bioinformatics 31:1286–1289
54. Abyzov A, Urban AE, Snyder M, Gerstein M (2011) CNVnator: an approach to discover, genotype, and characterize typical and atypical CNVs from family and population genome sequencing. Genome Res 21(6):974–984
55. Klambauer G, Schwarzbauer K, Mayr A et al (2012) cn.MOPS: mixture of Poissons for discovering copy number variations in next-generation sequencing data with a low false discovery rate. Nucleic Acids Res 40:e69
56. Handsaker RE, Van Doren V, Berman JR et al (2015) Large multiallelic copy number variations in humans. Nat Genet 47:296–303
57. Nguyen HT, Merriman TR, Black MA (2014) The CNVrd2 package: measurement of copy number at complex loci using high-throughput sequencing data. Front Genet 5:248
58. Hollox EJ (2009) Beta-defensins and Crohn's disease: confusion from counting copies. Am J Gastroenterol 105:360–362
59. Shrestha S, Tang J, Kaslow RA (2009) Gene copy number: learning to count past two. Nat Med 15:1127–1129
60. Alkan C, Kidd JM, Marques-Bonet T et al (2009) Personalized copy number and segmental duplication maps using next-generation sequencing. Nat Genet 41:1061–1067
61. Holt C, Losic B, Pai D, Zhao Z, Trinh Q, Syam S, Arshadi N, Jang GH, Ali J, Beck T, McPherson J, Muthuswamy LB (2014) WaveCNV: allele-specific copy number alterations in primary tumors and xenograft models from next-generation sequencing. Bioinformatics 30(6):768–774
62. Boeva V, Zinovyev A, Bleakley K, Vert JP, Janoueix-Lerosey I, Delattre O, Barillot E (2011) Control-free calling of copy number alterations in deep-sequencing data using GC-content normalization. Bioinformatics 27 (2):268–269
63. Xie C, Tammi MT (2009) CNV-seq, a new method to detect copy number variation using high-throughput sequencing. BMC Bioinf 10:80
64. Xi R, Hadjipanayis AG, Luquette LJ, Kim TM, Lee E, Zhang J, Johnson MD, Muzny DM, Wheeler DA, Gibbs RA, Kucherlapati R, Park PJ (2011) Copy number variation detection in whole-genome sequencing data using the Bayesian information criterion. Proc Natl Acad Sci 108:E1128–E1136
65. Gusnanto A, Wood HM, Pawitan Y, Rabbitts P, Berri S (2012) Correcting for cancer genome size and tumour cell content enables better estimation of copy number alterations from next-generation sequence data. Bioinformatics 28:40–47
66. Yoon S, Xuan Z, Makarov V, Ye K, Sebat J (2009) Sensitive and accurate detection of copy number variants using read depth of coverage. Genome Res 19:1586–1592
67. Zhang Q, Ding L, Larson DE, Koboldt DC, McLellan MD, Chen K, Shi X, Kraja A, Mardis ER, Wilson RK, Borecki IB, Province MA (2010) CMDS: a population-based method for identifying recurrent DNA copy number aberrations in cancer from high-resolution data. Bioinformatics 26(4):464–469
68. Wang Z, Hormozdiari F, Yang WY, Halperin E, Eskin E (2013) CNVeM: copy number variation detection using uncertainty of read mapping. J Comput Biol 20(3):224–236
69. Sinha R, Samaddar S, De RK (2015) CNV-CH: a convex hull based segmentation approach to detect copy number variations (CNV) using next-generation sequencing data. PLOS One 10(8):e0135895

Chapter 3

Applications of Community Detection Algorithms to Large Biological Datasets

Itamar Kanter, Gur Yaari, and Tomer Kalisky

Abstract

Recent advances in data acquiring technologies in biology have led to major challenges in mining relevant information from large datasets. For example, single-cell RNA sequencing technologies are producing expression and sequence information from tens of thousands of cells in every single experiment. A common task in analyzing biological data is to cluster samples or features (e.g., genes) into groups sharing common characteristics. This is an NP-hard problem for which numerous heuristic algorithms have been developed. However, in many cases, the clusters created by these algorithms do not reflect biological reality. To overcome this, a Networks Based Clustering (NBC) approach was recently proposed, by which the samples or genes in the dataset are first mapped to a network and then community detection (CD) algorithms are used to identify clusters of nodes.

Here, we created an open and flexible python-based toolkit for NBC that enables easy and accessible network construction and community detection. We then tested the applicability of NBC for identifying clusters of cells or genes from previously published large-scale single-cell and bulk RNA-seq datasets.

We show that NBC can be used to accurately and efficiently analyze large-scale datasets of RNA sequencing experiments.

Key words Networks based clustering, Community detection, Single-cell RNA sequencing, Big data

1 Introduction

Advances in high-throughput genomic technologies have revolutionized the way biological data is being acquired. Technologies like DNA sequencing (DNA-seq), RNA sequencing (RNA-seq), chromatin immunoprecipitation sequencing (ChIP-seq), and mass cytometry are becoming standard components of modern biological research. The majority of these datasets are publicly available for further large-scale studies. Notable examples include the Genotype-Tissue Expression (GTEx) project [1], the cancer

genome atlas (TCGA) [2], and the 1000 genomes project [3]. Examples of utilizing these datasets include studying allele-specific expression across tissues [4, 5], characterizing functional variation in the human genome [6], finding patterns of transcriptome variations across individuals and tissues [7], and characterizing the global mutational landscape of cancer [8]. Moreover, some of these genomic technologies have recently been adapted to work at the single-cell level [9]. While pioneering single-cell RNA sequencing (scRNA-seq) studies were able to process relatively small numbers of cells (42 cells in [10] and 18 cells in [11]), recent single-cell RNA-seq studies taking advantage of automation and nanotechnology were able to produce expression and sequence data from many thousands of individual cells (~1500 cells in [12] and ~40,000 cells in [13]). Hence, biology is facing significant challenges in handling and analyzing large complex datasets [14, 15].

1.1 Clustering Analysis

One of the common methods used for making sense of large biological datasets is cluster analysis: the task of grouping similar samples or features [16]. For example, clustering analysis has been used to identify subtypes of breast tumors [17, 18] with implications to treatment and prognosis. More recently, clustering analysis was used to identify and characterize cell types in various tissues and tumors in the colon [19], brain [20], blood [12], and lung [21], with the overall aim of finding key stem and progenitor cell populations involved in tissue development, repair, and tumorigenesis. Another application is to find sets of coordinately regulated genes in order to find gene modules [11, 22, 23]. Such clusters of genes (or other features such as single-nucleotide polymorphism (SNPs) [24]) can be further analyzed by gene set enrichment approaches to identify gene annotations [25] (e.g., GO [26], KEGG [27], and OMIM [28]) that are over-represented in a given cluster, and thus shed light on their biological functionalities [29]. Two of the most common clustering methods used in biology are K-means clustering, which groups data points into K prototypes (where K is a predetermined number of clusters), and hierarchical clustering, which builds a hierarchy of clusters from all data points [30]. Other methods for clustering include self-organizing map (SOM) [31], spectral clustering [32], and density-based methods [33] (for a comprehensive review on clustering *see* [30, 34, 35]).

1.2 Networks and Community Detection

Another way to model biological systems is through network science. Networks (also known as graphs) are structures composed of nodes that are connected by edges, which may be weighted and/or directional. Networks have been used for modeling interactions between components of complex systems such as users in social media platforms (Facebook [36] and Twitter [37]) or proteins [38] and genes [39] in a cell. Often, networks contain communities of

nodes that are tightly interconnected with each other, which is an indication of common features. For example, communities in social networks [40] are composed, in most cases, of people with common interests or goals that are interacting with each other. Likewise, proteins with related functionalities interact with each other, and as a result form close-knit communities in protein–protein interactions (PPI) networks [41]. Similarly, web pages with similar content in the World Wide Web [42] usually have links to each other.

The problem of community detection (CD), which can be viewed as a network's equivalent for clustering, is not rigorously defined. As a consequence, there are numerous CD algorithms that solve this problem reasonably well using different strategies [43]. An intuitive way to define communities in a network is to divide the nodes into groups that have many in-group edges and few out-group edges. This can be achieved by maximizing the network modularity—a measure that quantifies edge density within communities compared to edge sparseness between communities [44, 45] (*see* Subheading 2 for formal definition).

Numerous community detection algorithms were developed during the last two decades [43, 46]. Newman defined a measure for the modularity of a weighted network [45]. Clauset et al. developed a fast greedy algorithm to partition the nodes of the network in a way that maximizes the modularity by hierarchical agglomeration of the nodes [47]. Reichardt et al. proposed an approach based on statistical mechanics. Their algorithm models the network as a spin glass and aims to find the partition of nodes with maximal modularity by finding the minimal energy state of the system [48]. Rosvall et al. and Yucel et al. proposed methods to find the community structure by approximating the information flow in the network as a random walk [49, 50]. Jian et al. developed SPICi—a fast clustering algorithm for large biological networks based on expanding clusters from local seeds [51].

A popular community detection algorithm called the *Louvain* algorithm was proposed by Blondel et al. [52] (*see* Subheading 2 for details). The *Louvain* algorithm starts by defining each node as a separate community and then performs modularity optimization by an iterative heuristic two-step process. In the first step, the algorithm goes over all nodes of the network and checks, for each individual node, if the network modularity can be increased by removing it from its present community and joining it to one of its neighboring communities. The process is repeated until no further increase in modularity can be achieved. This approach is called the "local moving heuristic." In the second step, a new meta-network, whose nodes are the communities identified by the first step, is constructed. The two steps are repeated until maximum modularity is attained. It was shown that this algorithm can be improved further by modifying the "local moving heuristic".

Table 1
Examples of previously published methods for NBC

Name	Reference	Description
PhenoGraph	Manuscript: [54] Software: [56]	A KNN network is constructed using Euclidean distance. Then, a SNN network is constructed by using the number of shared neighbors between every two nodes as a new similarity measure between them. Communities are found using the *Louvain* algorithm.
SNN-Cliq	Manuscript: [55] Software: [57]	A KNN network is constructed using Euclidean distance (or similar). Then, a SNN network is constructed using the number of shared neighbors between every two nodes, as well as their distances to the two nodes, as a new similarity measure. Communities are found using a heuristic approach to find "quasi-cliques" and merge them.
Seurat	Manuscript: [58] Software: [59]	The algorithm constructs a KNN network and then a SNN network. Communities are found using the *Louvain* [52] or the SLM [53] algorithms.

SLM, for example, attempts to split each community into sub-communities prior to construction of the meta-network. In this way, communities can be split up and sets of nodes can be moved from one community to another for improving the overall modularity score [53].

Recently, several networks based clustering algorithms were developed specifically for single-cell gene expression datasets (*see* Table 1). Pe'er and colleagues developed PhenoGraph [54]. This method first builds a k-nearest neighbors (KNN) network of cells, where each cell is connected to its K nearest cells in Euclidean space. In order to better resolve rare or non-convex cell populations, the algorithm then constructs a second, shared nearest neighbors (SNN) network, in which the similarity between every two nodes is determined by the number of neighboring nodes that are connected to both of them. Finally, the *Louvain* community detection algorithm is used to find groups of cells with similar gene expression profiles. Applying their method to mass cytometry data from 30,000 human bone marrow cells, they were able to cluster single-cell expression profiles into different immune cell types. Su and Xu developed another algorithm called SNN-cliq [55]. This algorithm also constructs a KNN network, and then constructs a SNN network in which the weight between every two nodes is determined not only by the number of shared nearest neighbors, but also their distances to the two nodes. Communities then are detected using a quasi-clique-based clustering algorithm. When applying their method to several single-cell transcriptomics datasets, they found it to be more robust and precise than traditional clustering approaches.

In this chapter, we introduce an accessible and flexible python-based toolkit for Networks Based Clustering (NBC) for large-scale RNA-seq datasets in the form of an *IPython* notebook with a self-contained example. This toolkit allows the user to follow and modify various aspects of the algorithms detailed above [52–55, 58], that is, to map a given dataset into a KNN network, to visualize the network, and to perform community detection using a variety of similarity measures and community detection algorithms of his choice. This flexibility is important since, from our experience, different parameters and algorithms might work best for different datasets according to their specific characteristics. Using this toolkit, we tested the performance of NBC on previously published large-scale single-cell and bulk RNA-seq datasets.

2 Materials

2.1 Single-Cell and "Bulk" RNA Sequencing Datasets

We used Four datasets in this study:

1. Single-cell RNA-seq data from Patel et al. [60] containing single-cell gene expression levels from five patients with glioblastoma and two gliomasphere cell lines that were acquired using the SMART-SEQ protocol. We downloaded the preprocessed data from GEO [61]. Altogether, this dataset contains 543 cells by 5948 genes.
2. Single-cell RNA-seq data from Klein et al. [62] containing single-cell gene expression levels from mouse embryonic stem cells at different stages of differentiation that were acquired using the inDrop protocol. In that experiment, samples were collected along the differentiation timeline by sequencing single cells at 0, 2, 4, 7 days after withdrawal of leukemia inhibitory factor (LIF). We downloaded the preprocessed data from GEO [63] and removed genes with zero expression levels, resulting in a dataset of 8669 cells by 24,049 genes. For the analysis presented in Fig. 3, we first removed technical replicates and data from a control cell line, resulting in a total number of 2717 cells.
3. "Bulk" RNA sequencing datasets from the Genotype-Tissue Expression (GTEx) database [1]. We downloaded the data from the GTEx website [64] version 6. This dataset includes 8555 samples taken from 30 tissue types (according to the SMTS variable) of 570 individuals. Gene names were translated from Ensemble gene ID into HGNC symbols using the *BioMart* Bioconductor package [65]. In cases where we found multiple matches of the Ensemble gene ID's corresponding to a single HGNC symbol, the Ensemble gene ID with maximum average intensity across all samples was chosen. To

compare different clustering methods (Fig. 4a) we chose only samples originating from a single tissue type. Moreover, we omitted tissues having multiple detailed tissue types (according to the SMTSD variable) even if they had a single tissue type (indicated by the SMTS variable). Likewise, genes with zero expression were omitted, resulting in a dataset of 3174 samples by 33,183 genes from 21 tissue types.

4. Single-cell RNA-seq data from Macosko et al. [13] containing single-cell gene expression levels from a P14 mouse retina that were acquired using the Drop-Seq protocol. We downloaded the preprocessed data from GEO [66] and removed genes with zero expression, resulting in a dataset of 49,300 cells by 24,071 genes. This dataset was used to compare the performance, in terms of CPU time, of NBC, K-means, and hierarchical clustering as shown in Fig. 4b–c.

2.2 KNN Network Construction and Visualization

A KNN network with cosine similarity was constructed using the *scikit-learn* package [67] for machine learning in Python. Since the cosine distance was not directly available in the *scikit-learn* BallTree () function, we used a two-step implementation as follows: First, each sample was mean-centered and standardized such that it will have zero mean and unit length (L2 normalization). Next, the ball tree algorithm [68] was applied with Euclidean distance to find the K nearest neighbors of each sample and construct a KNN network. Then, the Euclidean distances between the nodes (=samples) were transformed to cosine similarities that were used as the edges weights for community detection.

We calculated the cosine similarity from the Euclidean distance as follows. The cosine similarity between two vectors A and B is defined as:

$$sim_{\cos}(A,B) \equiv 1 - \frac{\theta(A,B)}{\pi},$$

where $\theta(A, B)$ is the angle between A and B. If A and B are also of unit length (L2 normalized), then this angle is related to the Euclidean distance $D_{euc}(A, B)$ according to:

$$\theta(A,B) = \cos^{-1}\left[1 - \frac{D_{euc}(A,B)^2}{2}\right]$$

or: $D_{euc}(A,B) = \sqrt{2 - 2\cos\theta(A,B)}$.

Network layouts for visualization were created by the *fruchterman–reingold* algorithm [69] as implemented in the *igraph* Python and R packages [70]. For correlation similarity we calculated the full *spearman* correlation matrix $\rho(A, B)$ between any two vectors A and B using the *corr* function in R.

2.3 Community Detection Algorithms

In this chapter we generally used the *Louvain* [52] algorithm for community detection as implemented by the *igraph* Python and R packages for network analysis [70], apart from Fig. 3 in which we used the *fast greedy* [47] algorithm.

The *Louvain* method partitions the nodes of the network into communities $c1$, $c2$, $c3$, ..., such that network modularity score

$$Q = \frac{1}{2m}\sum_{i,j}\left[A_{ij} - \frac{k_i \cdot k_j}{2m}\right]\delta(c_i, c_j),$$

is maximized. In the above formula, A_{ij} is the edge weight between nodes i and j, k_i is the degree of node i (that is, the sum of the weights of all the links emanating from node i), m is the overall sum of weights, $m = \frac{1}{2}\sum_{ij} A_{ij}$, and $\delta(c_i, c_j)$ is the Kronecker delta function. The network modularity score is actually the difference between the number of links connecting nodes within the same community and the expected number of links in a network with randomly shuffled links.

Briefly, the algorithm starts by assigning a separate community to each node. Then, the algorithm iterates between two steps: In the first step, the modularity is maximized by repeatedly iterating over all nodes in the network. For each node, we evaluate the gain in modularity that will take place by removing it from its present community and assigning it to one of its neighboring communities. If the overall modularity can be improved, the community of the node is reassigned accordingly. This process is repeated until a local maximum is reached. In the second step, the algorithm constructs a meta-network in which the nodes are communities from the first step and the edges are the edges between the communities. At this point, the first step is repeated on the nodes of the new meta-network in order to check if they can be merged into even larger communities. The algorithm stops when there is no more improvement in the modularity score.

2.4 Statistical Measures for Comparing NBC and Other Common Clustering Algorithms

To compare the performance of NBC, hierarchical clustering, K-means, and spectral clustering, we used the *F-measure*, which is the harmonic mean between the precision P and sensitivity (recall) R:

$$F \equiv \frac{2}{\frac{1}{P}+\frac{1}{R}} = 2 * \frac{\mathrm{P} * \mathrm{R}}{\mathrm{P}+\mathrm{R}},$$

where $P \equiv Precision \equiv \frac{TP}{TP+FP}$, and $R \equiv Recall \equiv \frac{TP}{TP+FN}$ (TP—true positive, FP—false positive, FN—false negative). To calculate precision and sensitivity for each clustering algorithm, we also used the R package *clusterCrit* [71] that compares the labels from the original publication to the labels inferred by the algorithm.

Another requirement for evaluating and comparing the different clustering algorithms was to require all of them to find the same number of clusters. Therefore, the number of required clusters was set to the number of distinct groups from the original publication (7 clusters in Fig. 2, 4 clusters in Fig. 3, 21 clusters in Fig. 4a, etc.). We used the *stat* package in R [72] to run hierarchical clustering and K-means clustering with default parameters (Euclidean distance), apart from the linkage in hierarchical clustering which was set to *average* linkage. For spectral clustering we used the *specc* function from the *kernlab* R package [73] with default parameters. Generally, all parameters were chosen as default unless otherwise specified.

All computations were done on a standard PC with i7-4600 CPU with 2.10 GHz and 16 GB of RAM memory.

2.5 Tissue-Specific Reference Genes Lists from the Human Protein Atlas

Tissue-specific reference lists of genes were obtained from the Human Protein Atlas [74] version 14 [75]. Altogether, the housekeeping genes (HKG) reference list is composed of 8588 genes, and the liver-specific, pancreas-specific, and spleen-specific reference genes lists are composed of 436, 234, and 95 genes respectively. Genes that do not appear in the dataset or genes that appear in more than one tissue-specific list were removed, resulting in 8331, 403, 210, and 87 genes in the HKG, liver-specific, pancreas-specific, and spleen-specific reference genes lists respectively.

3 Methods

3.1 A Workflow for Networks Based Clustering (NBC)

A typical Networks Based Clustering workflow can be divided into four steps (Fig. 1). The given dataset, in the form of a matrix of N samples (e.g., cells) by P features (e.g., genes), is first preprocessed by normalizing the samples to each other [76, 77] and filtering less informative samples or features [77, 78]. This step is especially important in single-cell data since it is typically noisier than bulk samples. Likewise, in meta-analysis, it is important to normalize samples obtained from different sources in order to mitigate bias due to batch effects [79, 80]. In our *IPython* notebook we took a dataset that was collected and pre-filtered by Patel et al. [60] and normalized each sample such that it will have zero mean and unit length (L2 normalization).

Next, a similarity measure is defined between the samples (or alternatively, features) and a KNN (K-nearest neighbors) network is constructed as follows: First, each sample (or feature) is represented as a node. Then each node is linked to its K nearest neighboring nodes. Constructing a KNN network using the naïve algorithm has a complexity of $O(N^2)$ which is slow for large N. We therefore use the more efficient ball tree algorithm [68], whose complexity scales as $O(N \log(N))$ if supplied with a true distance

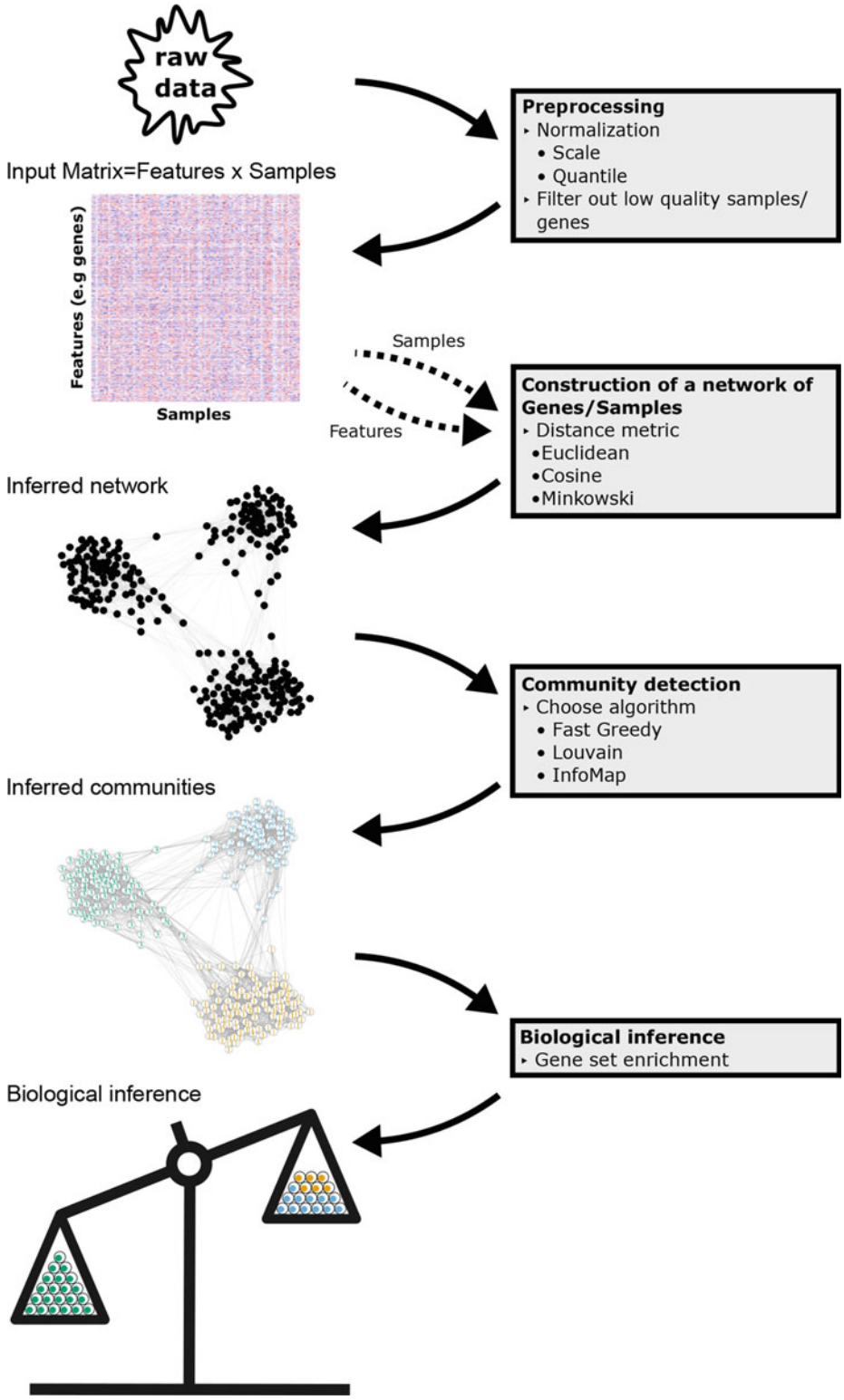


Fig. 1 A typical workflow for Networks Based Clustering (NBC). The raw data is first preprocessed to form a gene expression matrix. From this matrix, a weighted KNN network is constructed, in which each node represents a sample (e.g., a single cell) or a feature (e.g., a gene). Then, a community detection algorithm is applied to partition the nodes into closely knit communities, which can be characterized using enrichment strategies

metric that satisfies the triangle inequality [81]. There are various such distance-like measures (each based on a similarity measure) that can be used [82] to construct the network. For example, the Euclidean distance, the Minkowski distance, and the cosine distance. In our solved example we used the cosine similarity (*see* Subheading 2). Note that the popular correlation distance does not satisfy the triangle inequality [81] and hence is not a true metric. Following network construction, a community detection (CD) algorithm is performed, resulting in an assignment of each node (sample or feature) to a distinct community.

Once communities have been identified, each community can be characterized to infer its biological meaning. For example, communities of cells may represent cell sub-populations in complex tissues or tumors and can be identified using previously known markers [83, 84]. Similarly, the biological functionality of communities of genes (or of gene sets that are over-expressed in specific cell communities) can be inferred using enrichment analysis at the gene and gene set levels [29, 85–89].

3.2 NBC Accurately Resolves Seven Cell Types from a Glioblastoma Single-Cell RNA-seq Dataset

To test the performance of NBC, we analyzed single-cell RNA-seq datasets originally published by Patel et al. [60], for which the biological interpretation was known a priori. These datasets were previously obtained from five glioblastoma patients and two glioma sphere cell lines and were found to contain 7 biologically distinct cell types. A 2D representation of the data by PCA and tSNE can be found in the supplementary *IPython* notebook. We first calculated the distance between individual cells according to the cosine similarity and then constructed a KNN network with $K = 40$ (for details *see* Subheading 2). We applied the *Louvain* algorithm [52], detected communities of cells, and used the *F-measure* (the harmonic mean between precision and sensitivity) to check the degree to which the inferred communities reproduce the known cell types from the original publication. We found that NBC resolves the original seven cell types with high precision and sensitivity (Fig. 2a and b, *F-measure* = 0.93). Constructing the network with $K = 10$ resulted in a slightly lower *F-measure* (Fig. 2c and d, *F-measure* = 0.81), mainly due to the separation of one original cluster (indicated by light blue in Fig. 2c) into two inferred clusters (indicated by light blue and orange in Fig. 2d). Evaluating the precision and sensitivity of NBC for a wide range of K's shows that, for this dataset, NBC is quite robust to the choice of K for values larger than $K \approx 18$ (Fig. 2e). Using the correlation similarity for constructing the KNN network results in a similar performance in terms of the *F-measure* (Fig. 2e). In this dataset, NBC outperformed other common clustering methods (Table 2).

We performed a similar analysis on another single-cell RNA-seq dataset published by Klein et al. [62] containing 2717 mouse embryonic stem cells collected from four consecutive

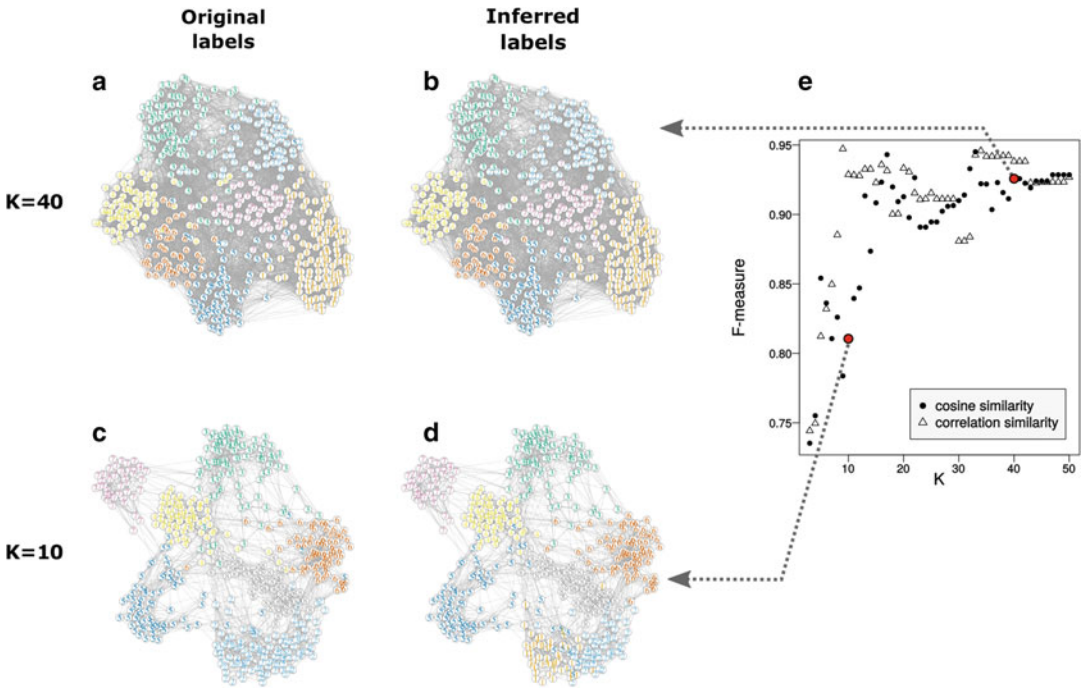


Fig. 2 NBC accurately resolves seven cell types from a glioblastoma single-cell RNA-seq dataset. We applied NBC to single-cell RNA-seq data published by Patel et al. [60] containing single cells collected from five patients with glioblastoma and two gliomasphere cell lines. A KNN network constructed using cosine similarity with $K = 40$ is shown in (**a**) and (**b**). A similar KNN network with $K = 10$ is shown in (**c**) and (**d**). Nodes shown in (**a**) and (**c**) are color-coded according to cell types reported by the original publication, while nodes in (**b**) and (**d**) are color-coded according to communities inferred by the *Louvain* algorithm. (**e**) The *F-measure*, which is the harmonic mean of precision and recall, is plotted against K for networks constructed with cosine (full circles) and correlation (empty triangles) similarities. Specific points corresponding to $K = 10$ and $K = 40$ are highlighted in red

Table 2
Comparison between NBC, K-means, hierarchical clustering, and spectral clustering in terms of the *F-measure* for two single-cell RNA-seq datasets. The *F-measure*, which is the harmonic mean between precision and sensitivity, measures the degree to which the inferred communities reproduce the known cell types from the original publication

Method	Patel et al. [60] dataset (Fig. 2)	Klein et al. [62] dataset (Fig. 3)
NBC (Cosine similarity, *Louvain* CD)	0.93 ($K = 40$)	0.81 ($K = 30$)
NBC (Correlation similarity, *Louvain* CD)	0.94 ($K = 40$)	0.90 ($K = 30$)
K-means (Euclidean similarity)	0.76	0.84
Hierarchical clustering (Euclidean similarity)	0.86	0.44
Hierarchical clustering (Correlation similarity)	0.79	0.44
Spectral clustering	0.47	0.80

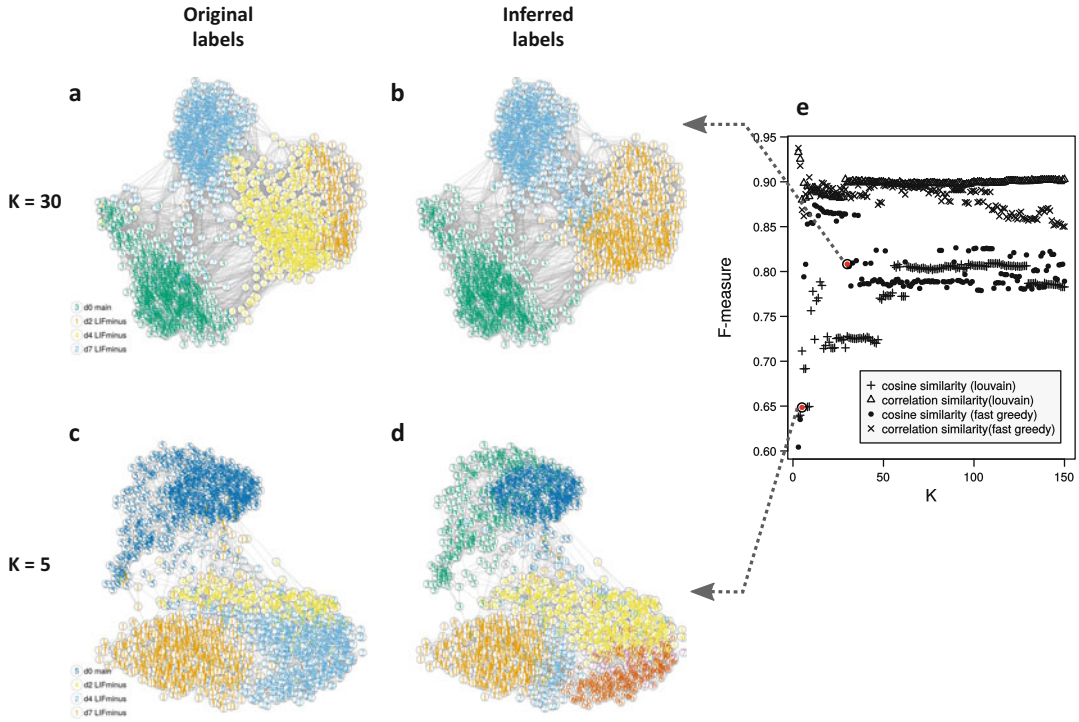


Fig. 3 NBC resolves four differentiation stages in a single-cell RNA-seq dataset from mouse embryonic stem cells. We applied NBC to single-cell RNA-seq data published by Klein et al. [62] containing single mouse embryonic stem cells collected from four consecutive developmental stages. A KNN network constructed using cosine similarity with $K = 30$ is shown in (**a**) and (**b**). A similar KNN network with $K = 5$ is shown in (**c**) and (**d**). Nodes shown in (**a**) and (**c**) are color-coded according to the differentiation stage reported by the original publication, while nodes in (**b**) and (**d**) are color-coded according to communities inferred by the *fast greedy* algorithm. (**e**) The *F-measure*, which is the harmonic mean of precision and recall, is plotted against K for networks constructed with cosine and correlation similarities and communities inferred by the *fast greedy* and *Louvain* algorithms. Specific points corresponding to $K = 5$ and $K = 30$ are highlighted in red

developmental stages. We found that also for this dataset, NBC performs well relative to other common clustering methods (Fig. 3 and Table 2). However, here we found that, for sufficiently large K ($K > 50$), correlation similarity had better performance than cosine similarity (*F-measure* = 0.90 for correlation similarity and 0.81 for cosine similarity, in both cases using the *Louvain* algorithm, Fig. 3e and Table 2).

3.3 Comparing NBC with Other Common Clustering Methods

We compared NBC with three widely used clustering algorithms: K-means, hierarchical clustering, and spectral clustering. As a reference dataset, we used 3174 human tissue-specific gene expression samples from the GTEx project [1] that were collected from 21 tissue types. Based on the number of tissue types in the original samples, the number of clusters was set to 21 for all clustering algorithms (*see* Subheading 2). A KNN network was constructed

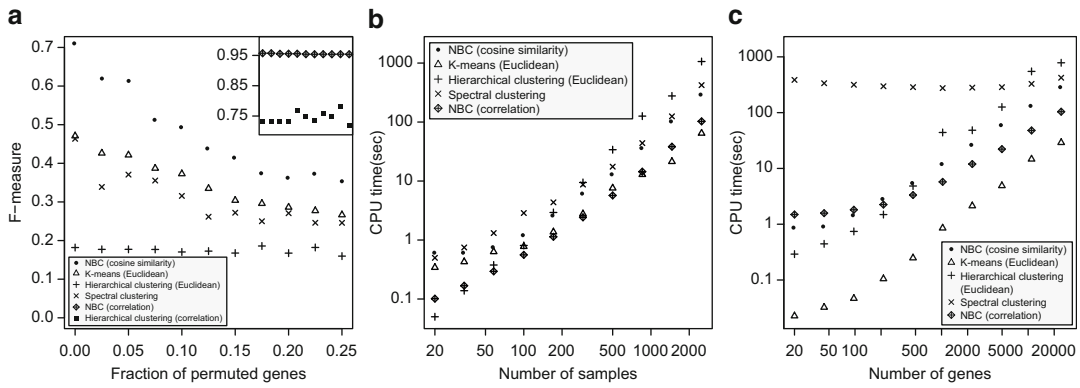


Fig. 4 Comparing NBC with other common clustering methods. (**a**) Shown is a comparison between NBC, K-means, hierarchical clustering, and spectral clustering in terms of the *F-measure* as a function of the effective number of genes. We effectively reduced the number of genes by randomly permuting the sample labels for a fraction of the genes. The inset shows results for NBC and hierarchical clustering with correlation similarity. (**b**, **c**) Shown is a comparison of the CPU times required by the five clustering methods, once as a function of the number of randomly chosen samples while keeping the number of genes fixed to 24,071 (**b**), and once as a function of the number of randomly chosen genes while keeping the number of samples fixed to 2500 (**c**). Each point shown is an average of three iterations. Data used in (**a**) was downloaded from the GTEx consortium [1], and data used in (**b**) and (**c**) was taken from a single-cell RNA-seq dataset published by Macosko et al. [13]

with $K = 50$ and the *Louvain* algorithm was applied to infer sample communities. In order to compare the algorithms and test their robustness to different levels of "noise," a fraction of the genes was randomly permuted by shuffling their sample labels (Fig. 4a), thereby maintaining the original distribution for each gene. This actually mimics different levels of non-informative features. We observed that for this data set, NBC out-performs K-means, spectral, and hierarchical clustering in terms of the *F-measure* over a wide range of noise levels (Fig. 4a).

We performed a similar comparison of the CPU time required to detect communities using a single-cell RNA-seq dataset that was published by Macosko et al. [13]. We randomly chose subsets of varying sizes of samples (cells) and genes in order to test the dependency of the running-time on the size of the dataset. We found that NBC falls in-between K-means, Hierarchical clustering, and spectral clustering for a wide range of samples and genes (Fig. 4b and c).

3.4 NBC Can Be Used to Resolve Tissue-Specific Genes

NBC can also be used to detect communities of genes from large gene expression datasets. To demonstrate this, we analyzed a dataset composed of 394 GTEx samples collected from three tissues: pancreas, liver, and spleen. First, a KNN network with $K = 10$ was constructed for the 394 samples. The resulting network contained three perfectly separated components, each corresponding to one

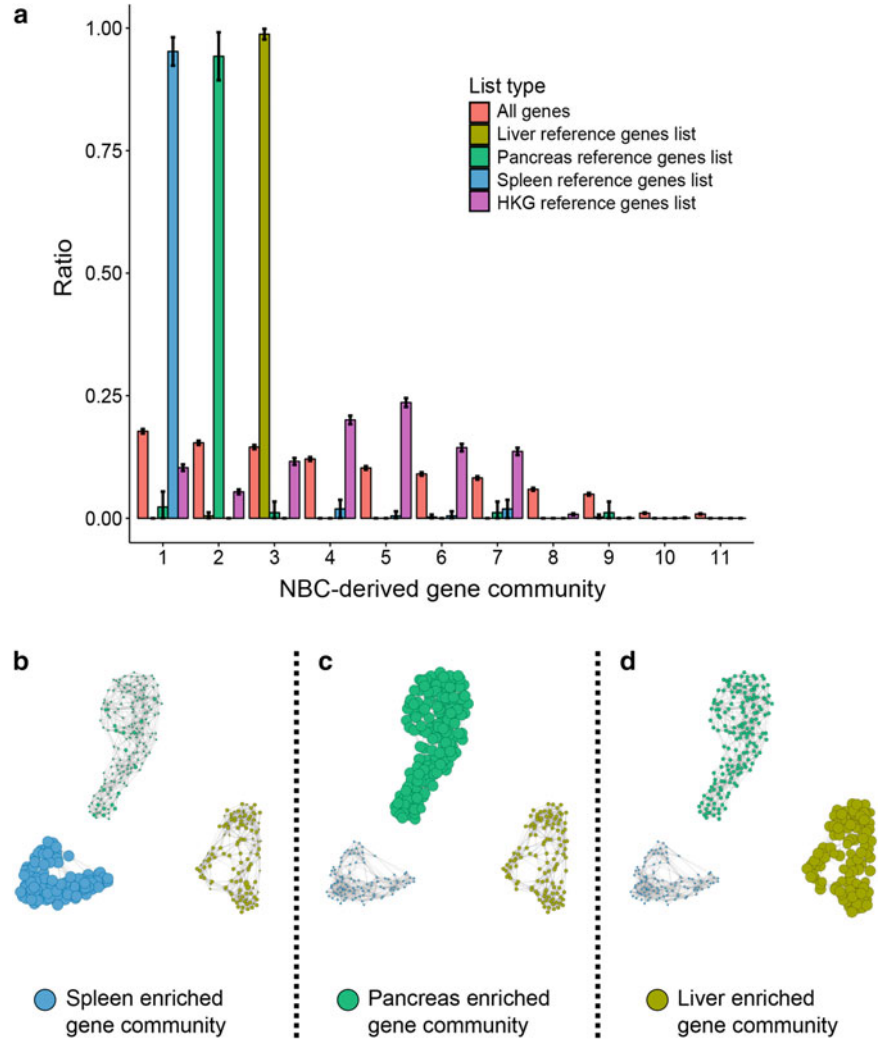


Fig. 5 NBC can be used to resolve tissue-specific genes. (**a**) NBC was applied to a mix of RNA-seq expression profiles from "bulk" samples of the human liver, pancreas, and spleen, that were obtained from the GTEx project [1]. Eleven communities of genes were detected by NBC using the *Louvain* algorithm. For each NBC-derived gene community, the relative fractions of all genes, housekeeping genes (HKG), and three tissue-specific reference genes lists are shown. The tissue-specific reference genes lists were downloaded from the Human Protein Atlas [74]. The NBC-derived gene communities are ordered according to their relative sizes, which is represented by the fraction of total genes that belong to each community (light red bars). (**b**–**d**) Shown is the network of samples, where in each panel the node size is proportional to the average log-transformed expression of the genes from NBC-derived community #1 (spleen-enriched, panel **b**), NBC-derived community #2 (pancreas-enriched, panel **c**), and NBC-derived community #3 (liver-enriched, panel **d**). The nodes are color-coded according to their respective tissue type

of the three tissue types (Fig. 5b–d, $F\text{-}measure = 1$). We then constructed another KNN network for the 27,838 genes with $K = 200$. The *Louvain* community detection algorithm was applied and 11 gene communities were detected.

Table 3
Summary statistics for NBC-derived gene communities

NBC-derived gene community no.	Total number of genes in community	Annotation of enriched tissue-specific "reference" genes list (from Human Protein Atlas)	Enrichment—number of genes from tissue-specific "reference" list (from the Human Protein Atlas) found in this NBC-derived community	*p*-Value for over-expression in corresponding tissue-specific samples (one side *t*-test with unequal variance)
1	4940	Spleen	200 of 210	2×10^{-61}
2	4284	Pancreas	82 of 87	2×10^{-250}
3	4043	Liver	397 of 403	2×10^{-124}

In order to explore the biological meaning of these 11 NBC-derived gene communities, we used three independently derived tissue-specific "reference lists" of genes from the Human Protein Atlas [74] that were found to be over-expressed in the pancreas, liver, and spleen. We compared these three reference lists to the 11 NBC-derived communities and found that each reference list was found predominantly in a single community (Fig. 5a). In community #1, 200 of the 210 spleen-specific reference genes were found, in community #2, 82 of the 87 pancreas-specific reference genes were found, and in community #3, 397 of the 403 liver-specific reference genes were found. On the contrary, a reference list of "House Keeping Genes" (HKG) was found to be distributed relatively uniformly among the different communities.

Another helpful feature of NBC is that it can be used to visualize families of genes within the network of samples. To demonstrate this, we measured the relative gene expression levels of genes from NBC-derived community no. 1 (enriched for spleen related genes) in each node (= sample) of the network and used this information to determine the size of that node (Fig. 5b). It can be seen that the average expression level of NBC-derived community #1, that is enriched for spleen-specific genes, was indeed much higher in the spleen samples compared to pancreas and liver samples (Table 3, one side *t*-test *p*-value$= 2 \times 10^{-61}$). We observed similar results when we repeated this analysis for NBC-derived communities 2 and 3 (pancreas-enriched and liver-enriched, Fig. 5c and d, Table 3).

4 Notes

To date, genomic datasets typically contain hundreds of samples with thousands of features each. FACS datasets may contain millions of samples with 10–30 features each. Improvements in

single-cell processing techniques (e.g., droplet-based [13] or magnetic-beads based methods [90]) are further increasing the number of samples in single-cell RNA-seq data. Therefore, tools for genomic data analysis need to perform efficiently on very large datasets. In this aspect, the major bottleneck of our toolkit is the KNN network construction step for which we use the ball tree algorithm [68]. Although efficient, this algorithm does not scale well with respect to memory usage and query time when the number of features increases. One possible solution is to use methods for approximating KNN networks, which might be utilized for this step after careful exploration of the error they introduce [91, 92]. Another possibility is to use parallel architectures to accelerate KNN network construction [93].

Moreover, when the number of features P is larger than the number of samples N ($P > N$), network construction can be made more efficiently by projecting the original matrix (embedded in $\mathbb{R}^P$) into a lower dimension space ($\mathbb{R}^N$) using PCA [94]. This projection preserves the original Euclidean distances between the samples. Other possible projections into lower dimensions are truncated SVD or approximated PCA methods [95]. However, these do not preserve the original Euclidean distances between the samples [95]. The original matrix can also be projected into lower dimensions by non-linear transformations like tSNE [96]. tSNE captures much of the local structure of the data and has been widely used for single-cell expression analysis [21, 97].

Note that Network-based methods themselves can be used for dimensionality reduction. Isomap [98], for instance, constructs a KNN graph that is used to approximate the geodesic distance between data points. Then, a multi-dimensional scaling is applied, based on the graph distances, to produce a low-dimensional mapping of the data that maintains the geodesic distances between all points.

NBC has much in common with the widely used density-based clustering method DBSCAN [33]. Although both methods explore the local structure of the data, NBC uses the K nearest neighbors, while DBSCAN defines clusters by their local densities. However in NBC, as opposed to DBSCAN, no minimal distance is required to define two samples as neighbors. In addition, DBSCAN does not produce the network explicitly, but rather just the connectivity component of each sample. This is in contrast to NBC that provides an explicit representation of the underlying weighted network that can be analyzed with different CD algorithms.

NBC requires the user to specify the following parameters: a similarity measure, a community detection algorithm, and the number of nearest neighbors K. For the datasets that we checked we found that NBC is not very sensitive to the choice of K given sufficiently large values ($K > 18$) (e.g., Fig. 2e); however, we found that the choice of the community detection algorithm and

especially the similarity measure may significantly influence its performance (e.g., Figs. 3e and 4a). Hence, these parameters should be chosen carefully when applying NBC to other data types. Similar to other machine learning approaches, NBC parameters can be optimized using a labeled training dataset prior to application on unlabeled data.

We created an open and flexible python-based toolkit for Networks Based Clustering (NBC) that enables easy and accessible KNN network construction followed by community detection for clustering large biological datasets, and used this toolkit to test the performance of NBC on previously published single-cell and bulk RNA-seq datasets. We find that NBC can identify communities of samples (e.g., cells) and genes, and that it performs better than other common clustering algorithms over a wide range of parameters.

In practice, given a new dataset, we recommend to carefully test different alternatives for network construction and community detection since results may vary among different datasets according to their unique characteristics. We believe that the open and flexible toolkit that we introduced here can assist in rapid testing of the many possibilities.

References

1. The GTEx Consortium (2015) The Genotype-Tissue Expression (GTEx) pilot analysis: multitissue gene regulation in humans. Science 348, 648–660. https://doi.org/10.1126/science.1262110, http://www.ncbi.nlm.nih.gov/pubmed/25954001
2. Cancer Genome Atlas Research Network, Weinstein JN, Collisson EA et al (2013) The Cancer Genome Atlas Pan-Cancer analysis project. Nat Genet 45:1113–1120. http://dx.doi.org/10.1038/ng.2764, http://10.0.4.14/ng.2764
3. Durbin RM, Altshuler DL, Durbin RM et al (2010) A map of human genome variation from population-scale sequencing. Nature 467:1061–1073. http://www.nature.com/doifinder/10.1038/nature09534
4. Baran Y, Subramaniam M, Biton A et al (2015) The landscape of genomic imprinting across diverse adult human tissues. Genome Res 25:927–936. http://genome.cshlp.org/lookup/doi/10.1101/gr.192278.115
5. Pirinen M, Lappalainen T, Zaitlen NA et al (2015) Assessing allele-specific expression across multiple tissues from RNA-seq read data. Bioinformatics 31:2497–2504. http://bioinformatics.oxfordjournals.org/lookup/doi/10.1093/bioinformatics/btv074
6. Lappalainen T, Sammeth M, Friedländer MR et al (2013) Transcriptome and genome sequencing uncovers functional variation in humans. Nature 501:506–511. http://www.nature.com/doifinder/10.1038/nature12531
7. Mele M, Ferreira PG, Reverter F et al (2015) The human transcriptome across tissues and individuals. Science 348:660–665. http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=4547472&tool=pmcentrez&rendertype=abstract, http://www.sciencemag.org/cgi/doi/10.1126/science.aaa0355
8. Leiserson MDM, Vandin F, Wu H-T et al (2014) Pan-cancer network analysis identifies combinations of rare somatic mutations across pathways and protein complexes. Nat Genet 47:106–114. http://www.nature.com/doifinder/10.1038/ng.3168
9. Nawy T (2013) Single-cell sequencing. Nat Methods 11:18. http://www.nature.com/doifinder/10.1038/nmeth.2801, http://www.nature.com/doifinder/10.1038/nmeth.2771
10. Ramsköld D, Luo S, Wang Y-C et al (2012) Full-length mRNA-Seq from single-cell levels of RNA and individual circulating tumor cells. Nat Biotechnol 30:777–782. http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3467340&tool=pmcentrez&rendertype=abstract, http://www.nature.com/doifinder/10.1038/nbt.2282
11. Shalek AK, Satija R, Adiconis X et al (2013) Single-cell transcriptomics reveals bimodality

in expression and splicing in immune cells. Nature 498:236–240. https://doi.org/10.1038/nature12172. http://www.ncbi.nlm.nih.gov/pubmed/23685454, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3683364&tool=pmcentrez&rendertype=abstract

12. Jaitin DA, Kenigsberg E, Keren-Shaul H et al (2014) Massively parallel single-cell RNA-Seq for marker free decomposition of tissues into cell types. Science 343:776–779. http://www.sciencemag.org/cgi/doi/10.1126/science.1247651
13. Macosko EZ, Basu A, Satija R et al (2015) Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. Cell 161:1202–1214. https://doi.org/10.1016/j.cell.2015.05.002. http://linkinghub.elsevier.com/retrieve/pii/S0092867415005498
14. Stephens ZD, Lee SY, Faghri F et al (2015) Big data: astronomical or genomical? PLoS Biol. 13:e1002195. http://dx.plos.org/10.1371/journal.pbio.1002195
15. Marx V (2013) Biology: the big challenges of big data. Nature 498:255–260. http://www.nature.com/doifinder/10.1038/498255a
16. Jiang D, Tang C, Zhang A (2004) Cluster analysis for gene expression data: a survey. IEEE Trans Knowl Data Eng 16:1370–1386. https://doi.org/10.1109/TKDE.2004.68. http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1339264
17. Sørlie T, Perou CM, Tibshirani R et al (2001) Gene expression patterns of breast carcinomas distinguish tumor subclasses with clinical implications. Proc Natl Acad Sci USA 98:10869–10874. https://doi.org/10.1073/pnas.191367098, http://www.ncbi.nlm.nih.gov/pubmed/11553815, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC58566
18. Kapp AV, Jeffrey SS, Langerød A et al (2006) Discovery and validation of breast cancer subtypes. BMC Genomics 7:231. https://doi.org/10.1186/147121647231, http://www.ncbi.nlm.nih.gov/pubmed/16965636, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC1574316
19. Rothenberg ME, Nusse Y, Kalisky T et al (2012) Identification of a cKit(+) colonic crypt base secretory cell that supports Lgr5(+) stem cells in mice. Gastroenterology 142:1195–1205.e6. https://doi.org/10.1053/j.gastro.2012.02.006, http://www.ncbi.nlm.nih.gov/pubmed/22333952
20. Pollen AA, Nowakowski TJ, Shuga J et al (2014) Low-coverage single-cell mRNA sequencing reveals cellular heterogeneity and activated signaling pathways in developing cerebral cortex. Nat Biotechnol 32:1053–1058. http://www.nature.com/doifinder/10.1038/nbt.2967
21. Treutlein B, Lee QY, Camp JG et al (2016) Dissecting direct reprogramming from fibroblast to neuron using single-cell RNA-seq. Nature 1–15. http://www.nature.com/doifinder/10.1038/nature18323
22. Kolodziejczyk AA, Kim JK, Tsang JCH et al (2015) Single cell RNA-sequencing of pluripotent states unlocks modular transcriptional variation. Cell stem cell 17:471–85. https://doi.org/10.1016/j.stem.2015.09.011. http://www.ncbi.nlm.nih.gov/pubmed/26431182, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4595712
23. Wang J, Xia S, Arand B et al (2016) Single-cell co-expression analysis reveals distinct functional modules, co-regulation mechanisms and clinical outcomes. PLoS Comput Biol 12:e1004892. https://doi.org/10.1371/journal.pcbi.1004892. http://www.ncbi.nlm.nih.gov/pubmed/27100869, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4839722
24. Wills QF, Livak KJ, Tipping AJ et al (2013) Single-cell gene expression analysis reveals genetic associations masked in whole-tissue experiments. Nat Biotechnol 31:748–752. https://doi.org/10.1038/nbt.2642, http://www.ncbi.nlm.nih.gov/pubmed/23873083
25. Hung J-H, Yang T-H, Hu Z et al (2012) Gene set enrichment analysis: performance evaluation and usage guidelines. Brief Bioinform 13:281–291. http://bib.oxfordjournals.org/cgi/doi/10.1093/bib/bbr049
26. Ashburner M, Ball CA, Blake JA et al (2000) Gene Ontology: tool for the unification of biology. Nat Genet 25:25–29. http://www.nature.com/doifinder/10.1038/75556
27. Kanehisa M (2000) KEGG: Kyoto Encyclopedia of Genes and Genomes. Nucleic Acids Res 28:27–30. http://nar.oxfordjournals.org/lookup/doi/10.1093/nar/28.1.27
28. Hamosh A (2004) Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders. Nucleic Acids Res 33:D514–D517. http://nar.oxfordjournals.org/lookup/doi/10.1093/nar/gki033
29. Huang DW, Sherman BT, Lempicki RA (2009) Bioinformatics enrichment tools: paths toward the comprehensive functional analysis of large gene lists. Nucleic Acids Res 37:1–13. http://naroxfordjournals.org/lookup/doi/10.1093/nar/gkn923

30. Jain AK, Murty MN, Flynn PJ (1999) Data clustering: a review. https://doi.org/10.1145/331499.331504
31. Kohonen T (1990) The self-organizing map. Proc IEEE 78:1464–1480. https://doi.org/10.1109/5.58325, http://ieeexplore.ieee.org/document/58325/
32. Von Luxburg U (2007) A tutorial on spectral clustering. Stat Comput 17:395–416. https://doi.org/10.1007/s11222007903 3z. arXiv:0711.0189v1
33. Martin E, Hans-Peter K, Jörg S et al (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD-96 Proceedings, pp 226–231. CiteSeerX:10.1.1.121.9220
34. Xu R, Wunsch DC (2010) Clustering algorithms in biomedical research: a review. IEEE Rev Biomed Eng 3:120–154. https://doi.org/10.1109/RBME.2010.2083647, http://www.ncbi.nlm.nih.gov/pubmed/22275205
35. Berkhin P (2006) A survey of clustering data mining techniques. In: Kogan J, Nicholas C, Teboulle M (eds) Grouping multidimensional data: recent advances in clustering. Springer, Berlin, pp 25–71. https://doi.org/10.1007/3540283498_2
36. Lewis K, Kaufman J, Gonzalez M et al (2008) Tastes, ties, and time: a new social network dataset using Facebook.com. Soc Netw 30:330–342. https://doi.org/10.1016/j.socnet.2008.07.002, http://linkinghub.elsevier.com/retrieve/pii/S0378873308000385
37. Ediger D, Jiang K, Riedy J et al (2010) Massive social network analysis: mining twitter for social good. In: 2010 39th International conference on parallel processing. IEEE, pp 583–593. https://doi.org/10.1109/ICPP.2010.66
38. Jeong H, Mason SP, Barabási A-L et al (2001) Lethality and centrality in protein networks. Nature 411:41–42. http://www.nature.com/doifinder/10.1038/35075138
39. Shen-Orr SS, Milo R, Mangan S et al (2002) Network motifs in the transcriptional regulation network of Escherichia coli. Nat Genet 31:64–68. http://www.nature.com/doifinder/10.1038/ng881
40. Papadopoulos S, Kompatsiaris Y, Vakali A et al (2012) Community detection in Social Media. Data Min Knowl Discov 24:515–554. http://link.springer.com/10.1007/s106180110224z
41. Chen J, Yuan B (2006) Detecting functional modules in the yeast protein-protein interaction network. Bioinformatics 22:2283–2290. http://bioinformatics.oxfordjournals.org/cgi/doi/10.1093/bioinformatics/btl370
42. Dourisboure Y, Geraci F, Pellegrini M (2007) Extraction and classification of dense communities in the web. In: Proceedings of the 16th international conference on world wide web WWW '07. ACM, New York, pp 461–470. https://doi.org/10.1145/1242572.1242635
43. Fortunato S (2010) Community detection in graphs. Phys Rep 486:75–174. https://doi.org/10.1016/j.physrep.2009.11.002, http://linkinghub.elsevier.com/retrieve/pii/S0370157309002841
44. Newman MEJ (2006) Modularity and community structure in networks. Proc Natl Acad Sci 103:8577–8582. http://www.pnas.org/cgi/doi/10.1073/pnas.0601602103
45. Newman MEJ, Girvan M (2004) Finding and evaluating community structure in networks. Phys Rev E 69:026113. http://www.ncbi.nlm.nih.gov/pubmed/14995526, https://link.aps.org/doi/10.1103/PhysRevE.69.026113
46. Newman MEJ (2004) Analysis of weighted networks. Phys Rev E 70:056131. https://link.aps.org/doi/10.1103/PhysRevE.70.056131
47. Clauset A, Newman MEJ, Moore C (2004) Finding community structure in very large networks. Phys Rev E 70:066111. http://link.aps.org/doi/10.1103/PhysRevE.70.066111
48. Reichardt J, Bornholdt S (2006) Statistical mechanics of community detection. Phys Rev E 74:016110. http://link.aps.org/doi/10.1103/PhysRevE.74.016110
49. Rosvall M, Bergstrom CT (2008) Maps of random walks on complex networks reveal community structure. Proc Natl Acad Sci 105:1118–1123. http://www.pnas.org/cgi/doi/10.1073/pnas.0706851105
50. Yucel M, Muchnik L, Hershberg U (2016) Detection of network communities with memory-biased random walk algorithms. J Complex Netw 5:48–69. http://comnet.oxfordjournals.org/content/early/2016/04/22/comnet.cnw007.abstract%5Cnpapers2://publication/doi/10.1093/comnet/cnw007
51. Jiang P, Singh M (2010) SPICi: a fast clustering algorithm for large biological networks. Bioinformatics (Oxford, England) 26:1105–1111. https://doi.org/10.1093/bioinformatics/btq078, http://www.ncbi.nlm.nih.gov/pubmed/20185405, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC2853685
52. Blondel VD, Guillaume J-L, Lambiotte R et al (2008) Fast unfolding of communities in large networks. J Stat Mech Theory Exp 2008:P10008.

https://doi.org/10.1088/17425468/2008/10/P10008, http://stacks.iop.org/17425468/2008/i=10/a=P10008?key=crossref.46968f6ec61eb8f907a760be1c5ace52

53. Waltman L, van Eck NJ (2013) A smart local moving algorithm for large-scale modularity-based community detection. Eur Phys J B 86:471. http://link.springer.com/10.1140/epjb/e201340829-0
54. Levine J, Simonds E, Bendall S et al (2015) Data-driven phenotypic dissection of AML reveals progenitor-like cells that correlate with prognosis. Cell 162:184–197. https://doi.org/10.1016/j.cell.2015.05.047, http://linkinghub.elsevier.com/retrieve/pii/S0092867415006376
55. Xu C, Su Z (2015) Identification of cell types from single-cell transcriptomes using a novel clustering method. Bioinformatics 31:1974–1980. http://dx.doi.org/10.1093/bioinformatics/btv088
56. PhenoGraph repository. https://github.com/jacoblevine/PhenoGraph. Accessed 3 May 2018
57. SNN-Cliq repository. http://bioinfo.uncc.edu/SNNCliq/. Accessed 3 May 2018
58. Butler A, Hoffman P, Smibert P et al (2018) Integrating single-cell transcriptomic data across different conditions, technologies, and species. Nat Biotechnol 36:411–420. https://doi.org/10.1038/nbt.4096, http://www.ncbi.nlm.nih.gov/pubmed/29608179
59. Seurat repository. http://satijalab.org/seurat/. Accessed 3 May 2018
60. Patel AP, Tirosh I, Trombetta JJ et al (2014) Single-cell RNA-seq highlights intratumoral heterogeneity in primary glioblastoma. Science (New York, N.Y.) 344:1–9. http://www.ncbi.nlm.nih.gov/pubmed/24925914, http://www.sciencemag.org/cgi/doi/10.1126/science.1254257
61. Series GSE57872. ftp://ftp.ncbi.nlm.nih.gov/geo/series/GSE57nnn/GSE57872/suppl/GSE57872_GBM_data_matrix.txt.gz. Accessed 7 Sept 2017
62. Klein AM, Mazutis L, Akartuna I et al (2015) Droplet barcoding for single-cell transcriptomics applied to embryonic stem cells. Cell 161:1187–1201. https://doi.org/10.1016/j.cell.2015.04.044, http://linkinghub.elsevier.com/retrieve/pii/S0092867415005000
63. Series GSE65525. http://www.ncbi.nlm.nih.gov/geo/download/?acc=GSE65525&format=file. Accessed 7 Sept 2017
64. GTEx Portal. http://www.gtexportal.org/home/datasets. Accessed 7 Sept 2017
65. Durinck S, Moreau Y, Kasprzyk A et al (2005) BioMart and bioconductor: a powerful link between biological databases and microarray data analysis. Bioinformatics (Oxford, England) 21:3439–3440. https://doi.org/10.1093/bioinformatics/bti525, http://www.ncbi.nlm.nih.gov/pubmed/16082012
66. Series GSE63472. ftp://ftp.ncbi.nlm.nih.gov/geo/series/GSE63nnn/GSE63472/suppl/GSE63472_P14Retina_merged_digital_expression.txt.gz. Accessed 7 Sept 2017
67. Pedregosa F, Varoquaux G, Gramfort A et al (2011) Scikit-learn: machine learning in python. J Mach Learn Res 12:2825–2830. http://dl.acm.org/citation.cfm?id=1953048.2078195
68. Omohundro SM (1989) Five balltree construction algorithms. International Computer Science Institute, Berkeley
69. Fruchterman TMJ, Reingold EM (1991) Graph drawing by force-directed placement. Softw Pract Exp 21:1129–1164. http://dx.doi.org/10.1002/spe.4380211102
70. Csardi G, Nepusz T (2006) The igraph software package for complex network research. Int J Complex Syst 1695(5):1–9
71. Desgraupes B (2018) clusterCrit: clustering indices
72. R Core Team (2016) R: a language and environment for statistical computing
73. Karatzoglou A, Smola A, Hornik K et al (2004) kernlab an S4 package for Kernel methods in R. J Stat Softw 11. https://doi.org/10.18637/jss.v011.i09, http://www.jstatsoft.org/v11/i09/
74. Uhlen M, Fagerberg L, Hallstrom BM et al (2015) Tissue-based map of the human proteome. Science 347:1260419. http://www.sciencemag.org/cgi/doi/10.1126/science.1260419
75. Human Protein Atlas Version 14. http://v14.proteinatlas.org. Accessed 6 Aug 2018
76. Bullard JH, Purdom E, Hansen KD et al (2010) Evaluation of statistical methods for normalization and differential expression in mRNA-Seq experiments. BMC Bioinformatics 11:94. http://bmcbioinformatics.biomedcentral.com/articles/10.1186/147121051194
77. Diaz A, Liu SJ, Sandoval C et al (2016) SCell: integrated analysis of single-cell RNA-seq data. Bioinformatics 32:2219–2220. http://bioinformatics.oxfordjournals.org/lookup/doi/10.1093/bioinformatics/btw201
78. Guo M, Wang H, Potter SS et al (2015) SINCERA: a pipeline for single-cell RNA-Seq profiling analysis. PLoS Comput Biol 11:e1004575. http://dx.plos.org/10.1371/journal.pcbi.1004575
79. Li P, Piao Y, Shon HS et al (2015) Comparing the normalization methods for the differential

analysis of Illumina high-throughput RNA-Seq data. BMC Bioinformatics 16:347. http://bmcbioinformatics.biomedcentral.com/articles/10.1186/s1285901507787

80. Vallejos CA, Risso D, Scialdone A et al (2017) Normalizing single-cell RNA sequencing data: challenges and opportunities. Nat Methods 14:565–571. http://www.nature.com/doifinder/10.1038/nmeth.4292
81. van Dongen S, Enright AJ (2012) Metric distances derived from cosine similarity and Pearson and Spearman correlations. Preprint, arXiv:1208.3145. http://arxiv.org/abs/1208.31451208.3145
82. Jaskowiak PA, Campello RJGB, Costa IG (2014) On the selection of appropriate distances for gene expression data clustering. BMC Bioinformatics 15(Suppl 2):S2. https://doi.org/10.1186/1471210515S2S2, http://www.ncbi.nlm.nih.gov/pubmed/24564555, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4072854
83. Heng TSP, Painter MW, Immunological Genome Project Consortium (2008) The Immunological Genome Project: networks of gene expression in immune cells. Nat Immunol 9:1091–1094. https://doi.org/10.1038/ni10081091, http://www.ncbi.nlm.nih.gov/pubmed/18800157
84. Harding SD, Armit C, Armstrong J et al (2011) The GUDMAP database–an online resource for genitourinary research. Development (Cambridge, England) 138:2845–2853. https://doi.org/10.1242/dev.063594, http://www.ncbi.nlm.nih.gov/pubmed/21652655, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3188593
85. Ritchie ME, Phipson B, Wu D et al (2015) limma powers differential expression analyses for RNA-sequencing and microarray studies. Nucleic Acids Res 43:e47. http://nar.oxfordjournals.org/lookup/doi/10.1093/nar/gkv007, https://academic.oup.com/nar/articlelookup/doi/10.1093/nar/gkv007, http://www.ncbi.nlm.nih.gov/pubmed/25605792, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4402510
86. Subramanian A, Tamayo P, Mootha VK et al (2005) Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. Proc Natl Acad Sci 102:15545–15550. http://www.pnas.org/cgi/doi/10.1073/pnas.0506580102
87. Yaari G, Bolen CR, Thakar J et al (2013) Quantitative set analysis for gene expression: a method to quantify gene set differential expression including gene-gene correlations. Nucleic Acids Res 41:e170. https://doi.org/10.1093/nar/gkt660, http://www.ncbi.nlm.nih.gov/pubmed/23921631, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3794608
88. Dalerba P, Kalisky T, Sahoo D et al (2011) Single-cell dissection of transcriptional heterogeneity in human colon tumors. Nat Biotechnol 29:1120–1127. https://doi.org/10.1038/nbt.2038, http://wwwpubmedcentral.nih.gov/articlerender.fcgi?artid=3237928%7B&%7Dtool=pmcentrez%7B&%7Drendertype=abstract, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=3237928&tool=pmcentrez&rendertype=abstract
89. Huang DW, Sherman BT, Lempicki RA (2009) Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. Nat Protoc 4:44–57. https://doi.org/10.1038/nprot.2008.211, http://www.ncbi.nlm.nih.gov/pubmed/19131956
90. Fan HC, Fu GK, Fodor SPA (2015) Combinatorial labeling of single cells for gene expression cytometry. Science 347:1258367. http://www.sciencemag.org/cgi/doi/10.1126/science.1258367, http://www.sciencemag.org/content/347/6222/1258367.abstract
91. Andoni A, Indyk P (2008) Near-optimal Hashing algorithms for approximate nearest neighbor in high dimensions. Commun ACM 51:117–122. http://doi.acm.org/10.1145/1327452.1327494
92. Bawa M, Condie T, Ganesan P (2005) LSH forest: self-tuning indexes for similarity search. In: Proceedings of the 14th international conference on world wide web WWW '05. ACM, New York, pp 651–660. https://doi.org/10.1145/1060745.1060840
93. Wang M, Zhang W, Ding W et al (2014) Parallel clustering algorithm for large-scale biological data sets. PLoS ONE 9:e91315. http://dx.plos.org/10.1371/journal.pone.0091315
94. Hastie T, Tibshirani R (2004) Efficient quadratic regularization for expression arrays. Biostatistics 5:329–340. https://doi.org/10.1093/biostatistics/kxh010, http://biostatistics.oxfordjournals.org/content/5/3/329.abstract
95. Halko N, Martinsson PG, Tropp JA (2011) Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions. SIAM Rev 53:217–288. http://epubs.siam.org/doi/abs/10.1137/090771806
96. van der Maaten L, Hinton GE (2008) Visualizing high-dimensional data using t-SNE. J Mach Learn Res 9:2579–2605

97. Tirosh I, Izar B, Prakadan SM et al (2016) Dissecting the multicellular ecosystem of metastatic melanoma by single-cell RNA-seq. Science 352:189–196. http://www.sciencemag.org/cgi/doi/10.1126/science.aad0501arXiv:1011.1669v3

98. Tenenbaum JB, de Silva V, Langford JC (2000) A global geometric framework for nonlinear dimensionality reduction. Science (New York, N.Y.) 290:2319–2323. https://doi.org/10.1126/science.290.5500.2319, http://www.ncbi.nlm.nih.gov/pubmed/11125149

Chapter 4

Processing and Analysis of RNA-seq Data from Public Resources

Yazeed Zoabi and Noam Shomron

Abstract

Advances in next generation sequencing (NGS) technologies resulted in a broad array of large-scale gene expression studies and an unprecedented volume of whole messenger RNA (mRNA) sequencing data, or the transcriptome (also known as RNA sequencing, or RNA-seq). These include the Genotype Tissue Expression project (GTEx) and The Cancer Genome Atlas (TCGA), among others. Here we cover some of the commonly used datasets, provide an overview on how to begin the analysis pipeline, and how to explore and interpret the data provided by these publicly available resources.

Key words RNA-seq, RNA sequencing, Genomics, Databases, Differential expression analysis, GTEx, TCGA, NGS

1 Introduction

1.1 The Rise of RNA-seq

The past 20 years have seen the development of various technologies to quantify global messenger RNA (mRNA) expression, or the "transcriptome". The need to view multiple gene levels in one experiment led to the development of hybridization-based approaches such as gene arrays and then to microarrays [1]. At first these were large membranes with printed spots, or "mountains" of complementary DNA (cDNA) to the query sequences. Once a spot "lights up" (e.g., via fluorescence) its intensity would indicate the amount of gene present in that experiment. These gene arrays were eventually miniaturized to the size of a few centimeters of glass slides containing millions of "query" spots, namely, microarrays [2–5]. In addition, increasingly specialized microarrays have been designed to study different variations of expression. For example, splicing junction microarrays with probes spanning known exon junctions are used to detect and quantify distinct spliced mRNA isoforms [6, 7]. Additionally, genomic tiling microarrays representing the genome at very high density allow for the mapping of the transcribed regions at a high resolution [4, 5, 8, 9].

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_4,

Despite their innovative approach and their ability to advance experimental work from a single to thousands of queries at once, these methods had several limitations [10–15]. They relied upon existing knowledge on the expressed gene's sequences and hence did not detect novel mRNAs; they exhibited high background levels owing to cross-hybridization; and presented a limited dynamic range of detection due to background for low expression transcripts and saturation of signals for high expression transcripts. Moreover, comparing expression levels across different experiments was often very difficult and required challenging normalization methods.

The paradigm shift arrived with the advent of next generation sequencing (NGS) [16, 17] and the development of RNA sequencing (RNA-seq) [18–21]. RNA-seq has become an omnipresent procedure in molecular biology with a novel idea of reading each RNA sequence present in the sample and then interpreting this data to expression levels (Fig. 1) [22]. RNA-seq is a key tool for understanding the transcriptome and hence to interpret the mechanisms underlying genomic function and human diseases (*see* **Notes 1** and **2**). RNA-seq is primarily used for differential gene expression (DGE) analysis, which is the basic method that allows users to determine the quantitative changes in expression levels of genes and/or transcripts between experimental groups [23].

The RNA-seq workflow begins in the laboratory, with RNA extraction and isolation, followed by RNA selection by mRNA enrichment or ribosomal RNA depletion, cDNA synthesis and preparation of an adaptor-ligated sequencing library. The library

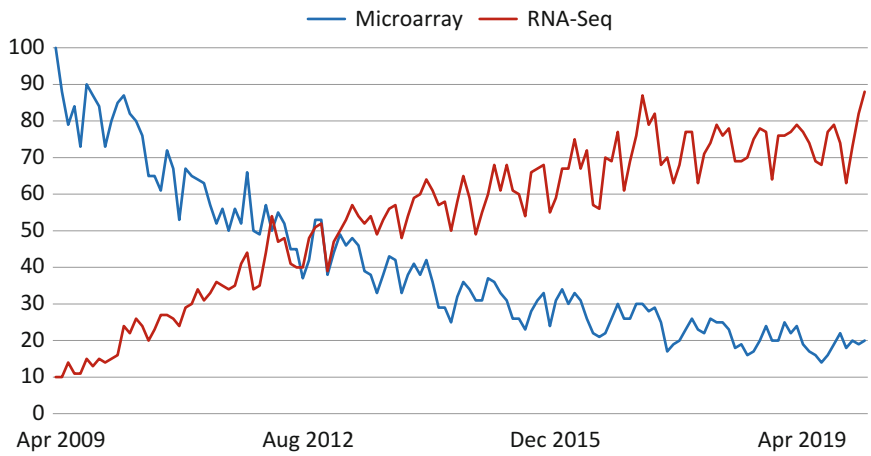


Fig. 1 Interest in the topics "RNA-seq" and "Microarray" during the time period ranging from April 1st, 2009 to April 1st, 2020. Data source: Google Trends (https:/www.google.com/trends)

Table 1
Overview of public resources containing mRNA expression data (for other useful resources, *see* Notes 3 and 4)

Resource name	Number of RNA-seq samples and type	Affiliation
TCGA [24]	9895 samples from 31 cancer types	National Cancer Institute (USA)
GTEx [25]	17,383 samples across 54 normal tissues from 980 healthy individuals	Broad Institute (USA)
CCLE [36]	1019 cancer cell lines	Broad Institute (USA)
HipSci [37]	693 iPSC lines	Sanger Institute (UK)
Expression Atlas [38]	>3000 experiments from 40 different organisms. ~1100 human experiments	EMBL-EBI (UK)

is then sequenced on a high-throughput platform. This should be followed by computational analysis: aligning the sequencing reads to a transcriptome, quantifying reads that overlap transcripts, and DGE analysis.

1.2 Data Availability and Impact

Over the past decade, a multitude of projects have generated raw RNA-seq data. Two large-scale sequencing studies alone, The Cancer Genome Atlas (TCGA) [24] and the Genotype Tissue Expression (GTEx) [25, 26], have catalogued RNA-seq and gene expression in ~27,000 samples (*see* Table 1). Most recently, data from single-cell RNA-seq (scRNA-seq) experiments is also becoming publicly available through consortium projects such as the Human Cell Atlas [27] and others [28–30].

The accumulation of RNA-seq data in public repositories from samples from a wide variety of biological conditions (as we show here) offers a unique and unprecedented resource to gain deeper insight into complex human diseases and biological mechanisms. Along with the increasing interest in machine learning and deep learning for computational biology applications [31], the availability of these resources has made possible many recent advances in the research community [32–35].

2 Public Resources Overview

2.1 The Cancer Genome Atlas: TCGA

2.1.1 Introduction

The TCGA consortium, which is led by the National Institute of Health (NIH), makes publicly available molecular and clinical information for more than 30 types of human cancers including information about exomes (variant analysis), single nucleotide polymorphisms (SNP), DNA methylation, the transcriptome (mRNA), microRNAs (miRNA) and the proteome [24].

Sample types available at TCGA are primary solid tumors, recurrent solid tumors, blood derived normal and tumor, metastatic, and solid tissue normal. TCGA has quantified gene expression from ~10,000 samples.

Website: https://portal.gdc.cancer.gov/

2.1.2 Data Access

TCGA data is accessible via the National Cancer Institute (NCI) Genomic Data Commons (GDC) data portal (https://portal.gdc.cancer.gov/), GDC Legacy Archive (https://portal.gdc.cancer.gov/legacy-archive) and the Broad Institute's (*see* below) GDAC Firehose (https://gdac.broadinstitute.org/). Raw paired-end reads of the RNA-seq samples for the TCGA project can be retrieved from the Cancer Genomics Hub (CGHub) [39].

For some entries (e.g., stomach adenocarcinoma), only aligned sequence reads are provided. However, there are methods for extracting raw reads in these cases (*see* **Note 5**).

2.2 Genotype-Tissue Expression: GTEx

2.2.1 Introduction

The Genotype-Tissue Expression (GTEx) is an effort of The Eli and Edythe L. Broad Institute of MIT and Harvard that aims to make tissue-specific gene expression resources available for the research community [25, 26]. Samples were collected across 54 nondiseased tissue sites from 980 individuals (correct as of V8), primarily for molecular assays including WGS, WES, and RNA-seq. Samples were collected across 54 nondiseased tissue sites from 980 individuals (correct as of V8, Fig. 2).

Website: https://www.gtexportal.org/

2.2.2 Data Access

Data can be accessed from the GTEx Portal, which provides open access to data including gene expression, quantitative trait locus (QTLs), and histology images. Raw paired-end reads of the RNA-seq samples can be downloaded from the Database of Genotypes and Phenotypes (dbGaP, http://www.ncbi.nlm.nih.gov/gap), which hosts >17,000 RNA-seq samples for the GTEx study. Information about the up-to-date dbGaP accession number can be found in the documentation section in the GTEx website.

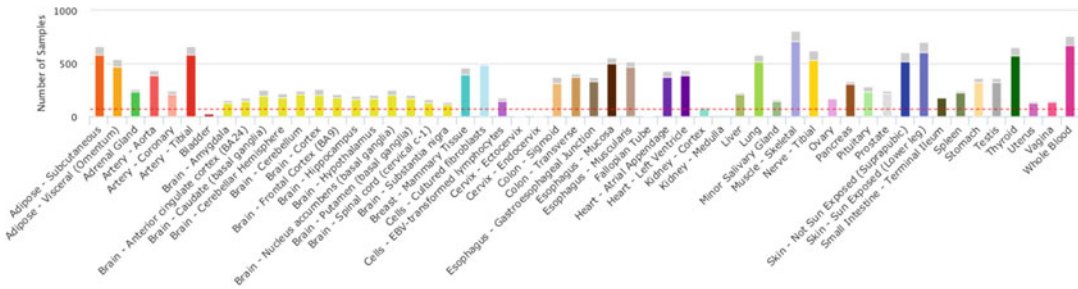


Fig. 2 GTEx sample counts by tissues, GTEx V8

2.3 Cancer Cell Line Encyclopedia: CCLE

2.3.1 Introduction

The CCLE is another effort of the Broad Institute of MIT and Harvard, which contains a collection of over 1,400 publicly available human cancer cell line data. It aims to construct a detailed genetic and pharmacologic characterization of a large panel of human cancer models. Through this data, researchers can link distinct pharmacologic vulnerabilities to genomic patterns and to translate cell line integrative genomics into cancer patient stratification. The CCLE includes data from RNA sequencing (RNA-seq), whole exome sequencing (WES), whole genome sequencing (WGS), reverse phase protein array (RPPA), reduced representation bisulfite sequencing (RRBS), microRNA expression profiling, and global histone modification profiling for cell lines from various lineages and ethnicities [36, 40].

Website: https://portals.broadinstitute.org/ccle

2.3.2 Data Access

The CCLE provides public access to genomic data, analysis and visualization for their cell lines through their website. The raw counts data can be downloaded through the CCLE website (https://portals.broadinstitute.org/ccle/data).

There are raw RNA-seq reads for ~1,000 cell lines currently available, which can be accessed through the European Nucleotide Archive (ENA), under the study number: PRJNA523380 (https://www.ebi.ac.uk/ena/data/view/PRJNA523380).

2.4 Human Induced Pluripotent Stem Cell Initiative: HipSci

2.4.1 Introduction

HipSci, which is maintained by the Wellcome Trust Sanger Institute in the UK, generates a high-quality reference panel of Induced Human Pluripotent Stem Cell (iPSC) lines from hundreds of healthy volunteers and patients from specific rare disease communities [37]. Lines generated by HipSci are extensively characterized with RNA-seq, whole exome sequencing, and other assays. HipSci cell lines are also available from cell banks such as The European Collection of Authenticated Cell Cultures (ECACC), and European Bank for Induced Pluripotent Stem Cells (EBiSC).

Website: http://www.hipsci.org/

2.4.2 Data Access

Data from HipSci can be accessed through the European Genome-phenome Archive (EGA) via a successful managed access application as described in the HipSci website (http://www.hipsci.org/).

2.5 Expression Atlas

2.5.1 Introduction

Expression Atlas is a database maintained by The European Molecular Biology Laboratory—European Bioinformatics Institute (EMBL-EBI) that can be used to query gene and protein expression data across species and biological conditions. Expression Atlas provides gene expression results on more than 3,000 experiments from 40 different organisms, all manually curated. Human experiments represent one third of Expression Atlas, with 1,098 experiments currently available, with thousands of microarrays and

RNA-seq datasets that are manually curated by biologists (*see* **Note 6**). Curation in Expression Atlas provides a critical review of each dataset to provide a comprehensive representation of gene expression data [38].

RNA-seq data are reanalyzed by Expression Atlas using the Integrated RNA-seq Analysis Pipeline (iRAP, https://github.com/nunofonseca/irap). RNA-seq experiments in Expression Atlas include data from large landmark studies mentioned above such as GTEx, CCLE, ENCODE, or HipSci.

Website: http://www.ebi.ac.uk/gxa

2.5.2 Data Access

Downstream analysis results are available via Expression Atlas. Raw reads can be downloaded if available for the specific experiment by accessing ArrayExpress on the Expression Atlas website.

3 Analysis

3.1 General Pipeline

After getting the raw reads in FASTQ format for any RNA-seq experiment (*see* **Note 7**), the pipeline for eventually performing a DGE analysis should start with quality control (QC) of the data. NGS data should be filtered from adapter contamination, base content biases, overrepresented sequences, and inaccurate representations of original nucleic acid sequences caused by library preparation and sequencing errors. In this first step, one should detect the fraction of low-quality reads, detect reads with uncalled bases, and detect contamination of bacterial origin in the samples. The most common tools for this steps are the FastQC software package (https://github.com/s-andrews/FastQC) [41] which was mainly designed for Illumina, and PRINSEQ [42] for Roche 454 technology. It is also recommended to perform adapter trimming, read filtering and base correction for FASTQ data before continuing with the analysis. Trimmomatic is a popular tool designed for RNA-seq data obtained from Illumina sequencers [43]. Another suggested tool is fastp, which is getting popular recently because of its fast performance and its ability to automatically detect adapter sequences [44].

Successful evaluation of sample quality and filtering should be followed by mapping sequence reads to a known transcriptome (or annotated genome), preferably to the latest version of the genome reference for more accurate profiling [45]. This can be done by converting each sequence read to one or more genomic coordinates alignment of the raw reads to the reference genome. This should normally be followed by quantification of gene abundance and read counting. For these two steps, there are several tools available, the most popular being STAR [46], HISAT2 [47], featureCounts [48], and the python framework HTseq [49].

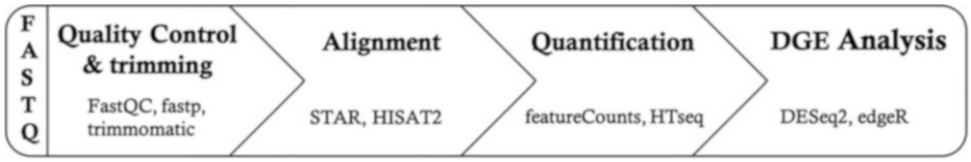


Fig. 3 A scheme for a recommended RNA-seq pipeline

Finally, for differential expression analysis, genes expressed differentially between the test and the reference groups of each pairwise contrast can be identified and modelled using tools such as DESeq2 [50] or edgeR [51], which have been shown to perform best when evaluated with different normalization methods [52]. Figure 3 shows a schematic flow chart of essential processing steps for RNA-seq analysis. Useful external resources on RNA-seq data analysis are described in **Note 8.**

3.2 Combining RNA-seq Data from Different Sources

Combining samples from two (or more) RNA-seq datasets might yield better insight into complex human diseases. However, it is not straightforward due to differences in sample and data processing, such as sequencing platform and chemistry, personnel and technical knowhow, and details in the analysis pipeline. For example, the RNA-seq expression levels of the majority of genes quantified in TCGA are in the range of 4–10 (log2 of normalized_count), and 0–4 (log2 of RPKM) for GTEx, and that is because different computational analysis pipelines were used, thus making it impossible to compare gene expression levels from the two projects directly.

A pipeline that processes and unifies RNA-seq data from different studies should include uniform realignment and gene expression quantification for all samples. Moreover, the removal of other batch effects is fundamental to facilitate data comparison when analyzing RNA-seq data from different sources.

3.2.1 General Approach

Recent studies have shown and validated that combining and integrating RNA-seq data from different studies can be made possible through a pipeline that removes batch effects for each study, and carefully reanalyzes the data by uniformly reprocessing the RNA-seq data. For example, researchers in [53] have combined GTEx and TCGA data through harmonizing them using quantile normalization and surrogate variable analysis (SVA)-based [54] batch effect removal.

Acquiring raw sequencing reads of the RNA-seq samples is the first step in this process. If FASTQ files are not available from the resources, aligned sequences can be used to extract raw reads (this was discussed previously under TCGA). Subsequently, one should realign the raw reads, requantify gene expression, and then remove

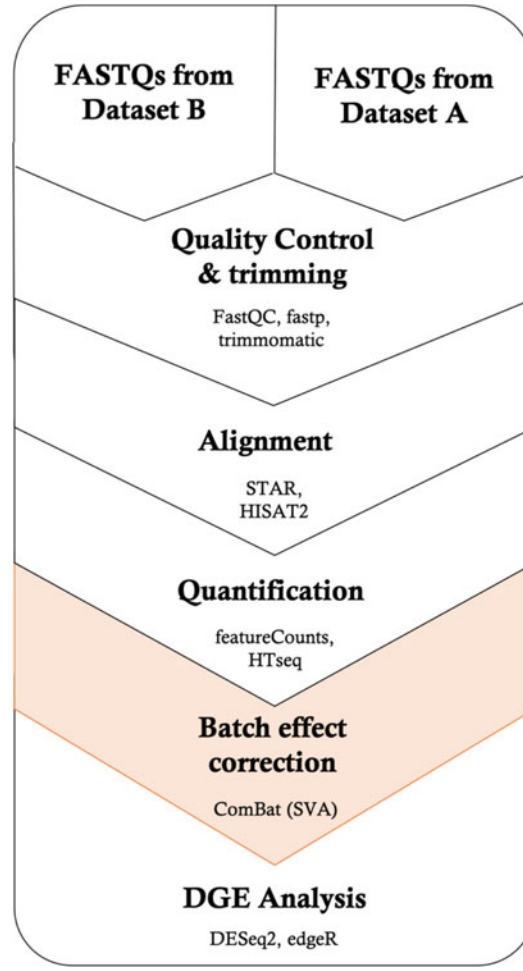


Fig. 4 An example of a pipeline for combining RNA-seq data from various studies

biases specific to each study. Figure 4 shows a schematic diagram for this process.

3.2.2 Batch Correction

Heterogenous gene expression data can present noise generated by a single or multiple sources of batch effects. This data can be adjusted by several powerful approaches for batch correction [54–59]. The SVA method [56] and its SVAseq [54] extension for RNA-seq data can be applied through the SVA package in R [54]. This package provides a framework for removing artifacts either by (1) estimating surrogate variables that introduce unwanted variability into high-throughput, high-dimensional datasets or (2) for removal of known covariates by using the ComBat function [60] which applies an empirical Bayes approach to remove those batch effects. We should also note that trying to remove batch effects may result in overcorrection, which results in the

loss of biologically valuable components of the data. Thus, several combined methods were developed recently to assess data over-correction and to evaluate the adjustments made to the data [59, 61–64].

4 Notes

1. As the transition to RNA-seq continues, microarray data remains a wealthy resource of gene expression data within public repositories. For example, ArrayExpress (maintained by the EBI) [65] contains more than 60,000 microarray experiments. Data from microarray experiments remains robust and trustworthy and can thus be exploited through integrating it with RNA-seq data [66, 67].
2. In the sequencing step in an RNA-seq experiment, the cDNA library is sequenced on a high-throughput platform to a read depth that varies depending on the goals of the study. For example, targeted RNA expression requires as few as three million reads per sample, meanwhile gene expression studies that aim to detect highly expressed genes only need 5–25 million reads per sample. Studies looking to build a more global view of gene expression, usually require 30–60 million reads per sample, and can even require 100–200 million reads if a more in-depth view of the transcriptome and proper detection of alternative splicing events is required [68–70].
3. File formats:
   (a) FASTQ is a text-based format for storing nucleotide sequences (reads) and their quality scores [71].
   (b) SAM: The Sequence Alignment/Map (SAM) format is a generic alignment format for storing read alignments against reference sequences, supporting short and long reads produced by different sequencing platforms [72].
   (c) BAM: is the binary representation of SAM and keeps exactly the same information as SAM [72].
4. Other useful RNA-seq data resources:
   (a) Human Protein Atlas (HPA) [73]—https://www.proteinatlas.org/
   (b) ENCODE [74]—https://www.encodeproject.org/
   (c) BodyMap 2.0 [75]—available in Expression Atlas.
   (d) Mammalian Transcriptomic Database (MTD) [76] http://mtd.cbi.ac.cn/
   (e) CellFinder [77]—http://cellfinder.org/

(f) BioXpress [78]—https://hive.biochemistry.gwu.edu/bioxpress

(g) FANTOM [79]—http://fantom.gsc.riken.jp/

(h) Gene Expression Omnibus (GEO) [80]—https://www.ncbi.nlm.nih.gov/geo/

5. The resources mentioned above include repositories for external data (Expression Atlas, GEO) and internally generated RNA-seq data (TCGA, GTEx, and others).
6. When FASTQ files are not available, for example, for stomach adenocarcinoma in TCGA, one can download aligned sequence reads (in BAM format) and extract reads from BAM files using UBU v1.0 (https://github.com/mozack/ubu) and SAMtools [72] before processing samples using the pipeline recommended above.
7. The sequencing platform for most RNA-seq studies is usually Illumina. However, there are other platforms available, such as the Thermo Fisher Scientific's Ion Proton System and the more recent direct RNA-seq technology developed by Oxford Nanopore Technologies (ONT) based on their MinION (or other) Nanopore sequencing device [81].
8. Useful external resources on RNA-seq data analysis:
   (a) A Guide for Designing and Analyzing RNA-Seq Data [82].
   (b) The DESeq2 R package authors in R provide a step by step RNA-seq data analysis workflow. They also provide an example on how to use SVA for batch correction in DGE analysis using DESeq2 [83].
   (c) An RNA-seq workflow using limma, Glimma, and edgeR [84].
   (d) Differential Expression Analysis of Complex RNA-seq Experiments Using edgeR [85].

References

1. Schena M, Shalon D, Davis RW, Brown PO (1995) Quantitative monitoring of gene expression patterns with a complementary DNA microarray. Science 270:467. https://doi.org/10.1126/science.270.5235.467
2. Clark TA, Sugnet CW, Ares M (2002) Genomewide analysis of mRNA processing in yeast using splicing-specific microarrays. Science 296:907. https://doi.org/10.1126/science.1069415
3. Yamada K, Lim J, Dale JM et al (2003) Empirical analysis of transcriptional activity in the arabidopsis genome. Science 302:842. https://doi.org/10.1126/science.1088305
4. Cheng J, Kapranov P, Drenkow J et al (2005) Transcriptional maps of 10 human chromosomes at 5-nucleotide resolution. Science 308:1149. https://doi.org/10.1126/science.1108625
5. David L, Huber W, Granovskaia M et al (2006) A high-resolution map of transcription in the yeast genome. Proc Natl Acad Sci U S A 103:5320. https://doi.org/10.1073/pnas.0601091103

6. Clark TA, Schweitzer AC, Chen TX et al (2007) Discovery of tissue-specific exons using comprehensive human exon microarrays. Genome Biol 8:R64. https://doi.org/10.1186/gb-2007-8-4-r64
7. Liu S, Lin L, Jiang P et al (2011) A comparison of RNA-Seq and high-density exon array for detecting differential gene expression between closely related species. Nucleic Acids Res 39:578–588. https://doi.org/10.1093/nar/gkq817
8. Bertone P, Stolc V, Royce TE et al (2004) Global identification of human transcribed sequences with genome tiling arrays. Science 306:2242. https://doi.org/10.1126/science.1103388
9. Mockler TC, Ecker JR (2005) Applications of DNA tiling arrays for whole-genome analysis. Genomics 85:1–15. https://doi.org/10.1016/j.ygeno.2004.10.005
10. Edwards HD, Nagappayya SK, Pohl NLB (2011) Probing the limitations of the fluorous content for tag-mediated microarray formation. Chem Commun 48:510–512. https://doi.org/10.1039/C1CC16022B
11. Khouja MH, Baekelandt M, Sarab A et al (2010) Limitations of tissue microarrays compared with whole tissue sections in survival analysis. Oncol Lett 1:827–831. https://doi.org/10.3892/ol_00000145
12. Tanase CP, Albulescu R, Neagu M (2011) Application of 3D hydrogel microarrays in molecular diagnostics: advantages and limitations. Expert Rev Mol Diagn 11:461–464. https://doi.org/10.1586/erm.11.30
13. Weisenberg JLZ (2008) Diagnostic yield and limitations of chromosomal microarray: a retrospective chart review. Ann Neurol 64:S101
14. Okoniewski MJ, Miller CJ (2006) Hybridization interactions between probesets in short oligo microarrays lead to spurious correlations. BMC Bioinformatics 7:276. https://doi.org/10.1186/1471-2105-7-276
15. Royce TE, Rozowsky JS, Gerstein MB (2007) Toward a universal microarray: prediction of gene expression through nearest-neighbor probe sequence identification. Nucleic Acids Res 35:e99–e99. https://doi.org/10.1093/nar/gkm549
16. Mardis ER (2008) The impact of next-generation sequencing technology on genetics. Trends Genet 24:133–141. https://doi.org/10.1016/j.tig.2007.12.007
17. Goodwin S, McPherson JD, McCombie WR (2016) Coming of age: ten years of next-generation sequencing technologies. Nat Rev Genet 17:333–351. https://doi.org/10.1038/nrg.2016.49
18. Wang Z, Gerstein M, Snyder M (2009) RNA-Seq: a revolutionary tool for transcriptomics. Nat Rev Genet 10:57–63. https://doi.org/10.1038/nrg2484
19. Marioni JC, Mason CE, Mane SM et al (2008) RNA-seq: An assessment of technical reproducibility and comparison with gene expression arrays. Genome Res 18:1509–1517. https://doi.org/10.1101/gr.079558.108
20. Mortazavi A, Williams BA, McCue K et al (2008) Mapping and quantifying mammalian transcriptomes by RNA-Seq. Nat Methods 5:621–628. https://doi.org/10.1038/nmeth.1226
21. Cloonan N, Forrest ARR, Kolle G et al (2008) Stem cell transcriptome profiling via massive-scale mRNA sequencing. Nat Methods 5:613–619. https://doi.org/10.1038/nmeth.1223
22. Stark R, Grzelak M, Hadfield J (2019) RNA sequencing: the teenage years. Nat Rev Genet 20:631–656. https://doi.org/10.1038/s41576-019-0150-2
23. Costa-Silva J, Domingues D, Lopes FM (2017) RNA-Seq differential expression analysis: an extended review and a software tool. PLoS One 12:e0190152. https://doi.org/10.1371/journal.pone.0190152
24. Chang K, Creighton CJ, Davis C et al (2013) The Cancer Genome Atlas pan-cancer analysis project. Nat Genet 45:1113–1120. https://doi.org/10.1038/ng.2764
25. Lonsdale J, Thomas J, Salvatore M et al (2013) The Genotype-Tissue Expression (GTEx) project. Nat Genet 45:580–585. https://doi.org/10.1038/ng.2653
26. The GTEx Consortium (2015) The Genotype-Tissue Expression (GTEx) pilot analysis: multitissue gene regulation in humans. Science 348:648. https://doi.org/10.1126/science.1262110
27. Rozenblatt-Rosen O, Stubbington MJT, Regev A, Teichmann SA (2017) The Human Cell Atlas: from vision to reality. Nature 550:451–453. https://doi.org/10.1038/550451a
28. Mereu E, Lafzi A, Moutinho C et al (2020) Benchmarking single-cell RNA-sequencing protocols for cell atlas projects. Nat Biotechnol 38(6):1–9. https://doi.org/10.1038/s41587-020-0469-4
29. Papatheodorou I, Moreno P, Manning J et al (2020) Expression Atlas update: from tissues to

single cells. Nucleic Acids Res 48:D77–D83. https://doi.org/10.1093/nar/gkz947

30. Franzén O, Gan L-M, Björkegren JLM (2019) PanglaoDB: a web server for exploration of mouse and human single-cell RNA sequencing data. Database 2019:baz046. https://doi.org/10.1093/database/baz046
31. Angermueller C, Pärnamaa T, Parts L, Stegle O (2016) Deep learning for computational biology. Mol Sys Biol 12:878. https://doi.org/10.15252/msb.20156651
32. Chiu Y-C, Chen H-IH, Zhang T et al (2019) Predicting drug response of tumors from integrated genomic profiles by deep neural networks. BMC Med Genet 12:18. https://doi.org/10.1186/s12920-018-0460-9
33. Sun Y, Zhu S, Ma K et al (2019) Identification of 12 cancer types through genome deep learning. Sci Rep 9:17256. https://doi.org/10.1038/s41598-019-53989-3
34. Zhang Z, Pan Z, Ying Y et al (2019) Deep-learning augmented RNA-seq analysis of transcript splicing. Nat Methods 16:307–310. https://doi.org/10.1038/s41592-019-0351-9
35. Xiong HY, Alipanahi B, Lee LJ et al (2015) The human splicing code reveals new insights into the genetic determinants of disease. Science 347:1254806. https://doi.org/10.1126/science.1254806
36. Ghandi M, Huang FW, Jané-Valbuena J et al (2019) Next-generation characterization of the Cancer Cell Line Encyclopedia. Nature 569:503–508. https://doi.org/10.1038/s41586-019-1186-3
37. Streeter I, Harrison PW, Faulconbridge A et al (2017) The human-induced pluripotent stem cell initiative-data resources for cellular genetics. Nucleic Acids Res 45:D691–D697. https://doi.org/10.1093/nar/gkw928
38. Papatheodorou I, Fonseca NA, Keays M et al (2017) Expression Atlas: gene and protein expression across multiple studies and organisms. Nucleic Acids Res 46:D246–D251. https://doi.org/10.1093/nar/gkx1158
39. Wilks C, Cline MS, Weiler E et al (2014) The Cancer Genomics Hub (CGHub): overcoming cancer through the power of torrential data. Database 2014:bau093. https://doi.org/10.1093/database/bau093
40. Barretina J, Caponigro G, Stransky N et al (2012) The Cancer Cell Line Encyclopedia enables predictive modelling of anticancer drug sensitivity. Nature 483:603–607. https://doi.org/10.1038/nature11003
41. Andrews S, Krueger F, Segonds-Pichon A et al (2012) FastQC. Babraham, UK
42. Schmieder R, Edwards R (2011) Quality control and preprocessing of metagenomic datasets. Bioinformatics 27:863–864. https://doi.org/10.1093/bioinformatics/btr026
43. Bolger AM, Lohse M, Usadel B (2014) Trimmomatic: a flexible trimmer for Illumina sequence data. Bioinformatics 30:2114–2120. https://doi.org/10.1093/bioinformatics/btu170
44. Chen S, Zhou Y, Chen Y, Gu J (2018) fastp: an ultra-fast all-in-one FASTQ preprocessor. Bioinformatics 34:i884–i890. https://doi.org/10.1093/bioinformatics/bty560
45. Guo Y, Dai Y, Yu H et al (2017) Improvements and impacts of GRCh38 human reference on high throughput sequencing data analysis. Genomics 109:83–90. https://doi.org/10.1016/j.ygeno.2017.01.005
46. Dobin A, Davis CA, Schlesinger F et al (2012) STAR: ultrafast universal RNA-seq aligner. Bioinformatics 29:15–21. https://doi.org/10.1093/bioinformatics/bts635
47. Kim D, Langmead B, Salzberg SL (2015) HISAT: a fast spliced aligner with low memory requirements. Nat Methods 12:357–360. https://doi.org/10.1038/nmeth.3317
48. Liao Y, Smyth GK, Shi W (2013) featureCounts: an efficient general purpose program for assigning sequence reads to genomic features. Bioinformatics 30:923–930. https://doi.org/10.1093/bioinformatics/btt656
49. Anders S, Pyl PT, Huber W (2014) HTSeq—a Python framework to work with high-throughput sequencing data. Bioinformatics 31:166–169. https://doi.org/10.1093/bioinformatics/btu638
50. Love MI, Huber W, Anders S (2014) Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biol 15:550. https://doi.org/10.1186/s13059-014-0550-8
51. Robinson MD, McCarthy DJ, Smyth GK (2009) edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26:139–140. https://doi.org/10.1093/bioinformatics/btp616
52. Dillies M-A, Rau A, Aubert J et al (2012) A comprehensive evaluation of normalization methods for Illumina high-throughput RNA sequencing data analysis. Brief Bioinform 14:671–683. https://doi.org/10.1093/bib/bbs046

53. Wang Q, Armenia J, Zhang C et al (2018) Unifying cancer and normal RNA sequencing data from different sources. Sci Data 5:180061. https://doi.org/10.1038/sdata.2018.61
54. Leek JT (2014) svaseq: removing batch effects and other unwanted noise from sequencing data. Nucleic Acids Res 42:e161. https://doi.org/10.1093/nar/gku864
55. Leek JT, Johnson WE, Parker HS et al (2012) The sva package for removing batch effects and other unwanted variation in high-throughput experiments. Bioinformatics 28:882–883. https://doi.org/10.1093/bioinformatics/bts034
56. Leek JT, Storey JD (2007) Capturing heterogeneity in gene expression studies by surrogate variable analysis. PLoS Genet 3:e161. https://doi.org/10.1371/journal.pgen.0030161
57. Chakraborty S (2019) Use of Partial Least Squares improves the efficacy of removing unwanted variability in differential expression analyses based on RNA-Seq data. Genomics 111:893–898. https://doi.org/10.1016/j.ygeno.2018.05.018
58. Gagnon-Bartsch JA, Speed TP (2012) Using control genes to correct for unwanted variation in microarray data. Biostatistics 13:539–552. https://doi.org/10.1093/biostatistics/kxr034
59. Somekh J, Shen-Orr SS, Kohane IS (2019) Batch correction evaluation framework using a-priori gene-gene associations: applied to the GTEx dataset. BMC Bioinformatics 20:268. https://doi.org/10.1186/s12859-019-2855-9
60. Johnson WE, Li C, Rabinovic A (2006) Adjusting batch effects in microarray expression data using empirical Bayes methods. Biostatistics 8:118–127. https://doi.org/10.1093/biostatistics/kxj037
61. Oytam Y, Sobhanmanesh F, Duesing K et al (2016) Risk-conscious correction of batch effects: maximising information extraction from high-throughput genomic datasets. BMC Bioinformatics 17:332. https://doi.org/10.1186/s12859-016-1212-5
62. Mostafavi S, Battle A, Zhu X et al (2013) Normalizing RNA-sequencing data by modeling hidden covariates with prior knowledge. PLoS One 8:e68141. https://doi.org/10.1371/journal.pone.0068141
63. Long Q, Argmann C, Houten SM et al (2016) Inter-tissue coexpression network analysis reveals DPP4 as an important gene in heart to blood communication. Genome Med 8:15. https://doi.org/10.1186/s13073-016-0268-1
64. Chen C, Grennan K, Badner J et al (2011) Removing batch effects in analysis of expression microarray data: an evaluation of six batch adjustment methods. PLoS One 6:e17238. https://doi.org/10.1371/journal.pone.0017238
65. Rustici G, Kolesnikov N, Brandizi M et al (2013) ArrayExpress update—trends in database growth and links to data analysis tools. Nucleic Acids Res 41:D987–D990. https://doi.org/10.1093/nar/gks1174
66. Castillo D, Gálvez JM, Herrera LJ et al (2017) Integration of RNA-Seq data with heterogeneous microarray data for breast cancer profiling. BMC Bioinformatics 18:506. https://doi.org/10.1186/s12859-017-1925-0
67. Thompson JA, Tan J, Greene CS (2016) Cross-platform normalization of microarray and RNA-seq data for machine learning applications. PeerJ 4:e1621. https://doi.org/10.7717/peerj.1621
68. Considerations for RNA-Seq read length and coverage. https://support.illumina.com/bulletins/2017/04/considerations-for-rna-seq-read-length-and-coverage-.html?langsel=/us/. Accessed 6 Apr 2020
69. Conesa A, Madrigal P, Tarazona S et al (2016) A survey of best practices for RNA-seq data analysis. Genome Biol 17:13. https://doi.org/10.1186/s13059-016-0881-8
70. Liu Y, Ferguson JF, Xue C et al (2013) Evaluating the impact of sequencing depth on transcriptome profiling in human adipose. PLoS One 8:e66883. https://doi.org/10.1371/journal.pone.0066883
71. Cock PJA, Fields CJ, Goto N et al (2009) The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. Nucleic Acids Res 38:1767–1771. https://doi.org/10.1093/nar/gkp1137
72. Li H, Handsaker B, Wysoker A et al (2009) The Sequence Alignment/Map format and SAMtools. Bioinformatics 25:2078–2079. https://doi.org/10.1093/bioinformatics/btp352
73. Uhlén M, Fagerberg L, Hallström BM et al (2015) Tissue-based map of the human proteome. Science 347:1260419. https://doi.org/10.1126/science.1260419
74. Dunham I, Kundaje A, Aldred SF et al (2012) An integrated encyclopedia of DNA elements in the human genome. Nature 489:57–74. https://doi.org/10.1038/nature11247

75. Bradley RK, Merkin J, Lambert NJ, Burge CB (2012) Alternative splicing of RNA triplets is often regulated and accelerates proteome evolution. PLoS Biol 10:e1001229. https://doi.org/10.1371/journal.pbio.1001229
76. Sheng X, Wu J, Sun Q et al (2016) MTD: a mammalian transcriptomic database to explore gene expression and regulation. Brief Bioinform 18:28–36. https://doi.org/10.1093/bib/bbv117
77. Stachelscheid H, Seltmann S, Lekschas F et al (2013) CellFinder: a cell data repository. Nucleic Acids Res 42:D950–D958. https://doi.org/10.1093/nar/gkt1264
78. Wan Q, Dingerdissen H, Fan Y et al (2015) BioXpress: an integrated RNA-seq-derived gene expression database for pan-cancer analysis. Database 2015:bav019. https://doi.org/10.1093/database/bav019
79. Yu NY-L, Hallström BM, Fagerberg L et al (2015) Complementing tissue characterization by integrating transcriptome profiling from the Human Protein Atlas and from the FANTOM5 consortium. Nucleic Acids Res 43:6787–6798. https://doi.org/10.1093/nar/gkv608
80. Barrett T, Wilhite SE, Ledoux P et al (2013) NCBI GEO: archive for functional genomics data sets—update. Nucleic Acids Res 41:D991–D995. https://doi.org/10.1093/nar/gks1193
81. Garalde DR, Snell EA, Jachimowicz D et al (2018) Highly parallel direct RNA sequencing on an array of nanopores. Nat Methods 15:201–206. https://doi.org/10.1038/nmeth.4577
82. Chatterjee A, Ahn A, Rodger EJ et al (2018) A guide for designing and analyzing RNA-Seq data. Methods Mol Biol 1783:35–80. https://doi.org/10.1007/978-1-4939-7834-2_3
83. Love MI, Anders S, Kim V, Huber W (2015) RNA-Seq workflow: gene-level exploratory analysis and differential expression. F1000Res 4:1070. https://doi.org/10.12688/f1000research.7035.1
84. Law CW, Alhamdoosh M, Su S et al (2018) RNA-seq analysis is easy as 1-2-3 with limma, Glimma and edgeR. F1000Res 5:ISCB Comm J-1408. https://doi.org/10.12688/f1000research.9005.3
85. Chen Y, Lun ATL, Smyth GK (2014) Differential expression analysis of complex RNA-seq experiments using edgeR. In: Datta S, Nettleton D (eds) Statistical analysis of next generation sequencing data. Springer, Cham, pp 51–74

Chapter 5

Improved Analysis of High-Throughput Sequencing Data Using Small Universal *k*-Mer Hitting Sets

Yaron Orenstein

Abstract

High-throughput sequencing machines can read millions of DNA molecules in parallel in a short time and at a relatively low cost. As a consequence, researchers have access to databases with millions of genomic samples. Searching and analyzing these large amounts of data require efficient algorithms.

Universal hitting sets are sets of words that must be present in any long enough string. Using small universal hitting sets, it is possible to increase the efficiency of many high-throughput sequencing data analyses. But, generating minimum-size universal hitting sets is a hard problem. In this chapter, we cover our algorithmic developments to produce compact universal hitting sets and some of their potential applications.

Key words Universal hitting sets, Minimizers, de Bruijn graph

1 Introduction

Large amounts of DNA sequencing data are being produced in almost any biological or clinical study. Due to the low cost of sequencing millions of DNA reads in parallel, it has become standard to probe and measure molecular interactions and biomarkers using DNA read quantities [21]. Technologies based on high-throughput DNA sequencing have been developed for all of the major genomic tasks: genetic and structural variation detection, gene expression quantification, epigenomic signal quantification, protein binding measurements, and many more [8].

The common computational challenges shared among these bioinformatic tasks lie in the analysis of high-throughput DNA sequencing data. The challenges include read mapping to a reference genome, read compression, storing reads in a data structure for fast querying, finding read overlaps, and more [24]. Thus, it is not surprising that a large number of computational methods were developed to analyze high-throughput DNA sequencing reads [2].

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_5,

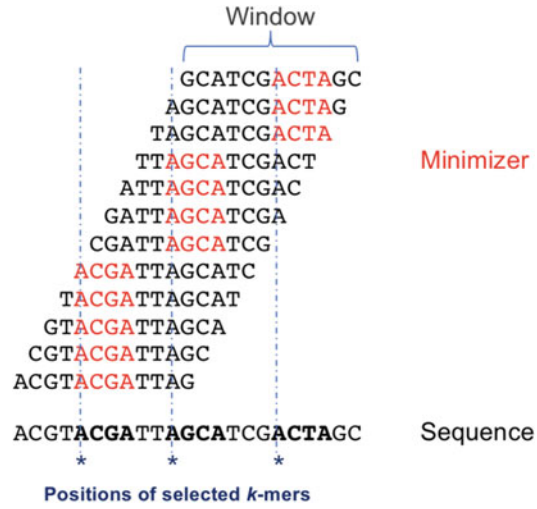


Fig. 1 A minimizers scheme. Given a sequence, in each window of length $L = 12$, a minimizer of length $k = 4$ is selected (red). Consecutive windows are likely to select the same minimizer, and the selected positions constitute a sampling of the original sequence (blue asterisks)

Many methods for analyzing high-throughput DNA sequencing are based on minimizers [10, 22, 23]. The minimizers scheme is a method to sample k-mers (words of length k) from a sequence. Given a sequence of length L and integer k, its *minimizer* is the smallest k-mer in it (smallest according to a predefined order, e.g., lexicographic order) among the $w = L - k + 1$ overlapping k-mers in it. When applying the scheme to a longer sequence, all L-long windows are scanned and the minimizer is selected in each one (Fig. 1).

Using minimizers to represent the windows in a sequence has three key advantages: (a) the sampling interval is small; (b) the same k-mers are often sampled from overlapping windows; and (c) identical windows have the same representative. Minimizers help design algorithms that are more efficient both in runtime and memory usage by reducing the amount of information to process while losing little or no information.

In bioinformatics of high-throughput sequencing, minimizers are being used in many different settings, such as binning input sequences [6, 22, 23], generating sparse data structures [7, 26], and sequence classification [25]. All of these applications share the need for a small signature, or fingerprint, in order to recognize longer exact matches between sequences.

The main measure of the performance of a minimizers scheme is its *density*, defined as the number of selected positions over the length of the sequence [13]. In most applications, a lower density is beneficial as it results in more savings in runtime and memory

usage. The density of a scheme depends on the choice of the order on the *k*-mers [1].

Recently, we introduced the concept of universal hitting sets [18] as a way to design minimizers schemes with low density. For integers k and L, a set U_{kL} is called a *universal hitting set* (UHS) of k-mers if every possible sequence of length L contains at least one k-mer from U_{kL} as a substring. In other words, it is a set of k-mers that is unavoidable by sequences of length at least L. The ability to generate UHSs of small size is a necessary step to design minimizers schemes with low density [13].

To that end, we developed several algorithms, DOCKS and its variants DOCKSany and DOCKSanyX [17, 18] to generate a small UHS. In this chapter, we will cover DOCKS and its variants. We will showcase its usage and runtime and memory performance. We will conclude with potential applications of UHSs in high-throughput sequencing analysis.

2 Materials

DOCKS (short for Design Of Compact *K*-mer Sets) is a software for finding a compact set of k-mers that hits any L-long sequence. DOCKS takes as input a list of parameters and outputs a list of k-mers. It is available via acgt.cs.tau.ac.il/docks/ and github.com/Shamir-Lab/DOCKS.

First, one needs to generate a decycling set, a set that hits any infinite-long sequence. This is done by running:

```
java -jar decycling.jar <output file> <k> <alphabet>
```

For example, the following command will generate a decycling set for $k=5$ over $\{A, C, G, T\}$. The output will be saved to decycling_5_ACGT.txt.

```
java -jar decycling.jar decycling_5_ACGT.txt 5 ACGT
```

Following the generation of a decycling set, additional k-mers to hit all remaining L-long sequences need to be found using the following command:

```
java -jar DOCKS.jar <output file> <input decycling set file> <k> <L - sequence length> <alphabet> 0 <1 / 2 - DOCKS / DOCKSAny> <X - optional input for DOCKSAny>.
```

An example of running DOCKS:

```
java -jar DOCKS.jar res_5_20_ACGT_0_1.txt decycling_5_ACGT.txt 5 20 ACGT 0 1
```

An example of running DOCKSany:

```
java -jar DOCKS.jar res_5_20_ACGT_0_2.txt decycling_5_ACGT.txt 5 20 ACGT 0 2
```

An example of running DOCKSanyX ($X=125$):

```
java -jar DOCKS.jar res_5_20_ACGT_0_2_125.txt
decycling_5_ACGT.txt 5 20 ACGT 0 2 125
```

In all of the runs, the user needs to provide a decycling set as input (can be computed by `decycling.jar`). Additional memory for higher values of k, i.e., $k \geq 10$, may be needed. This can be done, for example, by adding `-Xmx4096m` Java option.

The output of the decycling procedure and DOCKS is a list of k-mers, e.g.:

```
AAAACAAA
AAAAGAAA
AAAATAAA
AAACCAAA
...
ATAACGAA
TCACCGAA
GCCTACTA
TCCTCCTA
```

Each line is a k-mer in the set. The sets generated by the decycling process and the DOCKS method should be concatenated to form a complete UHS.

3 Methods

3.1 Definitions

For $k \geq 1$ and a finite alphabet Σ of size $\sigma = |\Sigma|$, a directed graph $B_k = (V, E)$ is a *de Bruijn graph* of order k if V is the set of all k-long strings over Σ, and an edge may exist from vertex u to vertex v if the $(k-1)$-suffix of u is the $(k-1)$-prefix of v (Fig. 2). Thus, for any edge $(u, v) \in E$ with label $\mathcal{L}$, labels of vertices u and v are the prefix and suffix of length k of $\mathcal{L}$, respectively. A path of w vertices in the graph where $w = L - k + 1$ represents an L-long string over Σ. A *minimum-size decycling set* (MDS) is a minimum subset of edges such that their removal turns the graph acyclic.

For any string s over Σ, we say that a set of k-mers M *hits* s if there exists a k-mer in M that is a contiguous substring in s. Consequently, a *universal hitting set* (UHS) U_{kL} is a set of k-mers that hits any L-long string over Σ. A trivial UHS always exists by taking all σ^k k-mers, but our interest is in one as small as possible, in order to reduce the computational expense for practical applications. One natural application of a small UHS is in minimizers schemes, as any w-long window of k-mers is hit by a U_{kL} if $w \geq L - k + 1$. Note that for any given k and L, a UHS does not depend on any particular dataset.

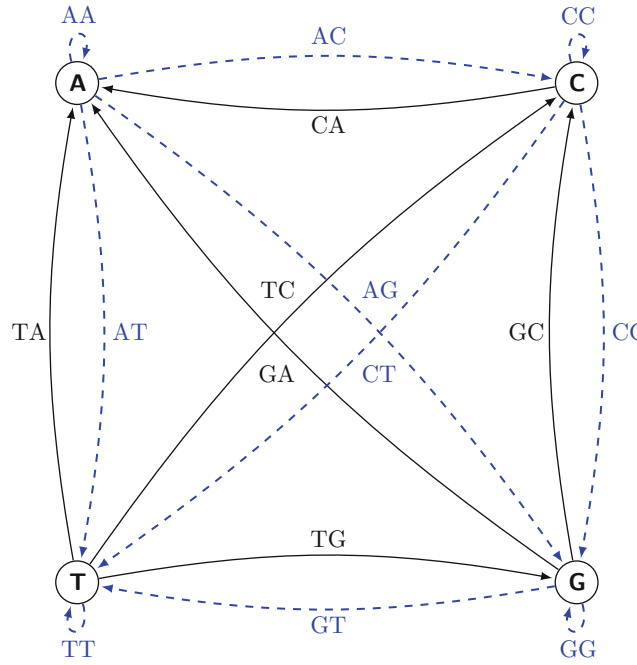


Fig. 2 A de Bruijn graph of order 1 over the DNA alphabet. The blue dashed edges form a minimum-size decycling set. This set of edges comprise a UHS for $k=2$ and $w=4$ (equivalently, $k=2$ and $L=5$), as the longest remaining path $T\rightarrow G\rightarrow C\rightarrow A$ (representing *TGCA*) is of length 4

A minimizers scheme is defined by parameters k, w and order o of the k-mers (e.g., lexicographic order or defined by a permutation of all k-mers). The *particular density* of order o on sequence S is the proportion of selected positions over the total number of k-mers in S:

$$d_{k,w,o}(S) = \frac{|\mathcal{S}_{k,w,o}(S)|}{|S| - k + 1}, \tag{1}$$

where $\mathcal{S}_{k,w,o}(S)$ is the set of selected positions. The *expected density* is the density expected for a random sequence.

Currently, the problem of computing a minimum-size UHS has no known hardness results, but there are several NP-hard problems related to it. In particular, the problem of computing a minimum-size UHS is very similar to the $(k,\mathcal{S})$-*hitting set* problem, which is the problem of finding a minimum-size k-mer set that hits a set $\mathcal{S}$ of input sequences (of any lengths), is NP-hard [18].

3.2 DOCKS

We previously developed DOCKS to generate a UHS for any given k and L [17, 18]. DOCKS first removes from a complete de Bruijn graph of order $k-1$ G=(V,E) an MDS constructed by Mykkeltveit's algorithm [15], which finds an MDS in $O(\sigma^k)$ time. The remaining graph G'=(V,E') is a directed acyclic graph (DAG).

Even after all cycles are removed from the graph, there may still be paths representing sequences of length L, which also need to be hit by the UHS. DOCKS removes an additional set of k-mers (represented as edges in the graph) that hits all remaining sequences of length L, so that no path representing an L-long sequence remains in the graph.

However, finding a minimum-size set of edges to cover all paths of length w in a DAG is NP-hard [19]. In order to find a small, but not necessarily minimum-size, set of edges to cover all w-long paths, we introduced the notion of a *hitting number*, the number of w-long paths containing edge $e=(u, v)$, denoted by $T(u, v, w)$ [18]. DOCKS uses the hitting number to prioritize the removal of edges that are likely to cover a large number of paths in the graph.

DOCKS computes the hitting numbers for all edges by dynamic programming: For any vertex v and $0 \leq i \leq w$, DOCKS calculates the number of i-long paths starting at v, $D(v, i)$, and the number of i-long paths ending at v, $F(v, i)$. Then, the hitting number in $G'=(v', E')$ is directly computable by

$$T(u, v, w) = \sum_{i=0}^{w-1} F(u, i) \cdot D(v, w-i-1)$$

and the dynamic programming calculation is given by

$$\forall v \in V,\ D(v, 0) = F(v, 0) = 1,$$

$$D(v, i) = \sum_{(v, u) \in E'} D(u, i-1),$$

$$F(v, i) = \sum_{(u, v) \in E'} F(u, i-1).$$

3.3 DOCKSany and DOCKSanyX

As an additional heuristic, we developed DOCKSany with a similar structure as DOCKS, but instead of removing the edge that hits the most w-long paths, it removes an edge that hits the most paths, in each iteration, after the removal of a decycling set [18]. This reduces the runtime by a factor of L, as calculating the hitting number $T(u, v)$ for each edge can be done in linear time in the size of the graph in DAGs.

DOCKSany computes the hitting numbers for all edges by dynamic programming: For any vertex v, DOCKSany calculates the number of paths starting at v, $D(v)$, and the number of paths ending at v, $F(v)$. Then, the hitting number in $G'=(v', E')$ is directly computable by

$$T(u, v) = F(u) \cdot D(v)$$

and the dynamic programming calculation, which is calculated in topological order for F and reverse topological order for D, is given by

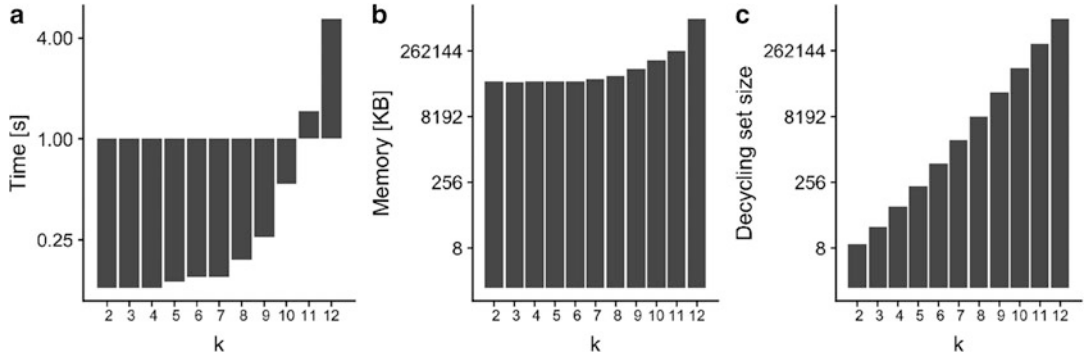


Fig. 3 Minimum decycling set generation. Measurements of our implementation of Mykkeltveit's MDS generation procedure for $2 \leq k \leq 12$ in (**a**) runtime, (**b**) maximum memory usage, and (**c**) resulting set size. Note that the *y*-axes are in log scale

$$D(v) = 1 + \sum_{(v,u)\in E'} D(u),$$

$$F(v) = 1 + \sum_{(u,v)\in E'} F(u).$$

The DOCKSanyX heuristic extends DOCKSany by removing X edges with the largest hitting numbers in each iteration, where X is a user-defined parameter.

3.4 Results

We benchmarked our implementation of Mykkeltveit's MDS generation process in both runtime and memory usage (Fig. 3). As this step runs in time $O(\sigma^k)$ [15], it is expected to terminate in seconds. Indeed, our results show that the sets are generated very fast and with small memory usage. The size of the sets converges to the theoretical asymptotic value of σ^k/k [15]. All runs were conducted on a server with two multi-core Intel Xeon Gold 6130 CPUs and 256 GB RAM.

We benchmarked DOCKS, DOCKSany, and DOCKSanyX on various combinations of k and L (Fig. 4a–c). Runtimes for finding additional k-mers, that together with the decycling set hit all L-long sequences span a wide range of values, as the theoretical complexity of DOCKS depends on the number of k-mers, i.e., $O(P_{\mathrm{DOCKS}}\sigma^k L)$, where P_{DOCKS} is the number of additional edges (representing k-mers) removed by DOCKS [18]. DOCKSany is faster and uses less memory, as each iteration runs in time linear in the size of the graph, i.e., $O(P_{\mathrm{DOCKSany}}\sigma^k)$. But, its produced sets are larger. DOCKSanyX is faster than DOCKSany, as it removes X edges in each iteration, but its set sizes are larger. Its theoretical runtime is $O(P_{\mathrm{DOCKSanyX}}\sigma^k/X)$.

To benchmark the quality of each of the computed UHSs, we calculated the expected density for the UHS computed by each method and compared it to a minimizers scheme with lexicographic ordering (Fig. 4d). Results show that smaller UHSs give rise to

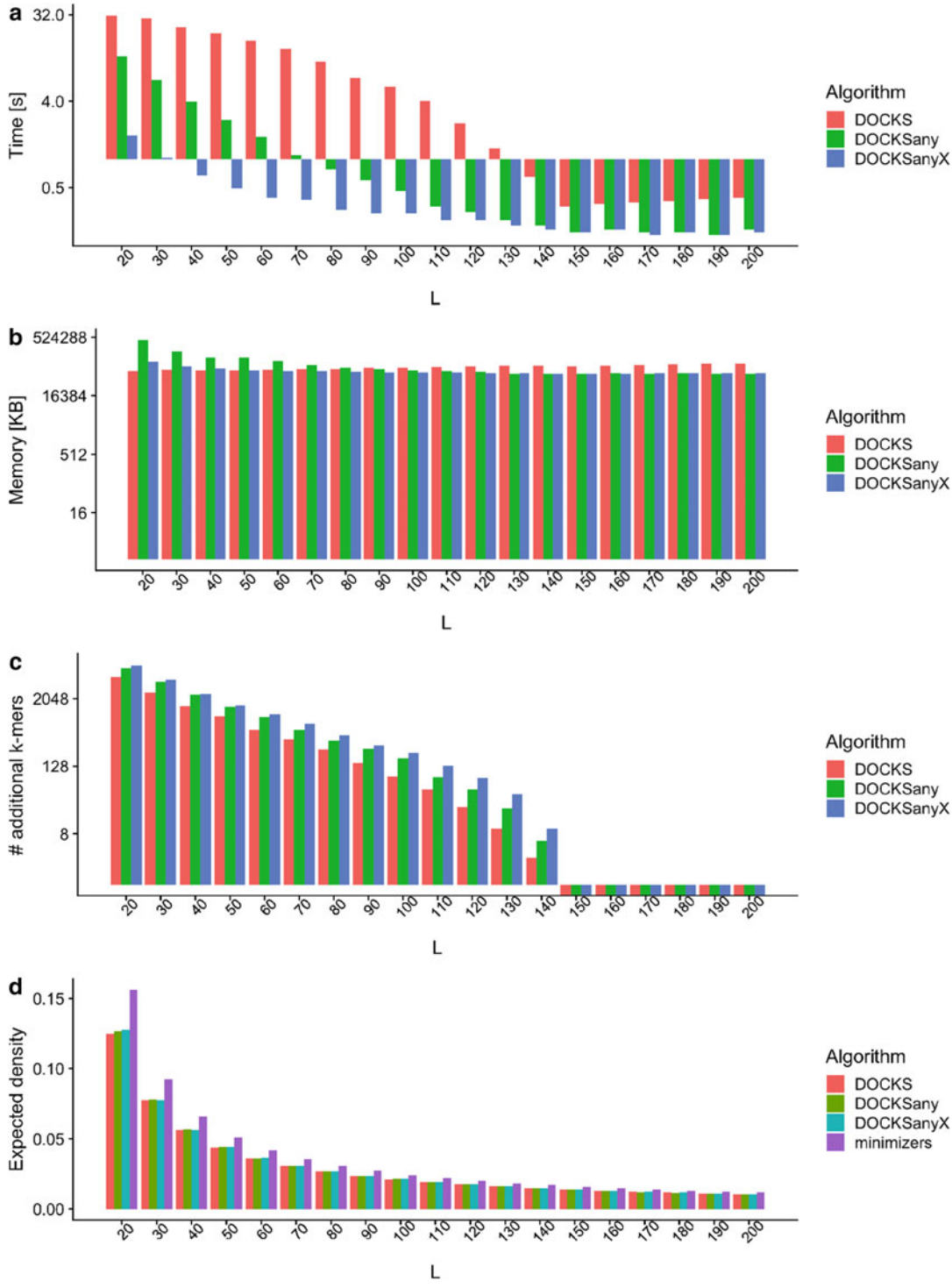


Fig. 4 Comparison of methods to find additional *k*-mers. Measurements of DOCKS, DOCKSany, and DOCKSanyX for $k = 8$ and $20 \leq L \leq 200$ in (**a**) runtime, (**b**) maximum memory usage, and (**c**) the number of additional *k*-mers. (**d**) Expected density for UHSs computed by the different methods, including a minimizer scheme based on a lexicographic order. Note that the y-axes in panels (a,b,c) are in log scale

lower densities, which is expected to improve high-throughput sequence analysis tasks. Moreover, all UHSs, whether generated by DOCKS, DOCKSany, or DOCKSanyX, had lower density compared to a minimizers scheme based on a lexicographic order.

3.5 Potential Applications

An improvement in many high-throughput sequence analysis tasks can be immediately achieved by updating the ordering of minimizers schemes with more efficient orders based on small UHSs. By using sets with lower expected density, speed-ups in runtime and reduced memory usage in genomic analyses may be obtained.

Improved Sequence Binning by UHSs: Many high-throughput sequence analyses perform binning of sequences as a preprocessing step to allow for efficient parallel processing and storage. For example, the MSP algorithm performs *de novo* read assembly by first binning genomic sequences using their minimizers as the hash key (with $k=12$ and $L=59$), assembling the sequences in each bin into a separate de Bruijn graph and then merging these graphs [12]. The handling of bins separately (in sequence or in parallel) reduces storage requirements by a factor of 10–20 for vertebrate genomes and runtime by a factor of 2–10. The metagenomic read classifier Kraken stores a database of long words ($L=31$ by default) grouped by their minimizers ($k=15$), so that adjacent L-mers will be found consecutively in memory and will be together in CPU cache [25]. BCALM2 hashes k-mers into bins based on their minimizers to parallelize de Bruijn graph compaction ($L=25$–71, $k=4$–10) [5]. Fast and memory efficient L-mer counters bin long L-mers using different hashing methods, including minimizers [6, 11].

An improvement in runtime and memory usage may be achieved by introducing UHS-based minimizers schemes into these and other binning-based applications. For example, in the MSP algorithm the number of bins the sequences are distributed to is fixed [12]. Hence, the main advantage in using UHS-based minimizers schemes may be in obtaining a flatter distribution of bin sizes and in particular in reducing the maximum bin size. This is expected following our previous results showing that k-mers from a small UHS are more evenly distributed along the genome than lexicographic minimizers [18].

The same approach may be successful in other binning applications. By changing the values of k and L, different numbers of bins and bin sizes may be created. This may allow each application to achieve its target number of bins or bin size, while controlling bin occupancy better, and making memory usage as efficient as possible while fully utilizing parallelism. A UHS-based minimizers scheme in a binning method is expected to provide improvements in runtime, memory usage, number of bins, and distribution of bin size compared to lexicographic minimizer-based binning methods on sequencing data for different target values of bin sizes and numbers.

Improved Sequence Similarity Estimation by UHSs: Another important application to potentially benefit from UHS is sequence similarity estimation. For different genomic data, researchers need to estimate sequence similarity for many sequencing reads [20]. The similarity between two sequences is commonly measured by the Jaccard index of the k-mer set comprising the two sequences [16]. But, iterating over all k-mers comprising two sequences is time-consuming. Thus, the Jaccard index value can be approximated by the overlap of randomly selected k-mers, using different random k-mer orderings, a process known as sketching [14].

Several methods were developed for this task in past years, with MinHash making a breakthrough in sequence similarity estimations using minimizers [16]. MinHash was shown to dramatically speed up sequence comparison in high-throughput sequencing applications including metagenomics, single-molecule sequencing, and long reads [4, 9]. MinHash uses $k = 16$ and $L \approx 100$. Though the result of applying the MinHash sketch on minimizers is no longer a random sample of k-mers in the read, it was shown empirically that the quality of the estimation of the Jaccard distance is not reduced [3].

Methods for sequence similarity estimation may be improved by using minimizers that are based on UHSs. As we previously showed [18], fewer k-mers are used by a winnowing scheme with an ordering defined by a U_{kL} than that defined by a lexicographic order scheme. Moreover, these k-mers are more evenly distributed. Hence, basing the minimizers order by a UHS is likely to substantially reduce the memory requirements. Moreover, it can improve the accuracy of the Jaccard similarity score estimation between sequences, since the UHS-based k-mers are more evenly covered than the lexicographically smallest k-mers.

4 Notes

Pre-computed UHSs for many combinations of k and L are available, without the need to run any software to generate them, at acgt.cs.tau.ac.il/docks/.

References

1. Almutairy M, Torng E (2018) Comparing fixed sampling with minimizer sampling when using k-mer indexes to find maximal exact matches. PLoS One 13(2):e0189960
2. Anders S, Pyl PT, Huber W (2015) HTSeq—a Python framework to work with high-throughput sequencing data. Bioinformatics 31(2):166–169
3. Baker DN, Langmead B (2019) Dashing: fast and accurate genomic distances with HyperLogLog. Genome Biol 20(1):265
4. Berlin K, Koren S, Chin CS, Drake JP, Landolin JM, Phillippy AM (2015) Assembling large genomes with single-molecule sequencing and locality-sensitive hashing. Nat Biotechnol 33(6):623

5. Chikhi R, Limasset A, Medvedev P (2016) Compacting de Bruijn graphs from sequencing data quickly and in low memory. Bioinformatics 32(12):i201–i208
6. Deorowicz S, Kokot M, Grabowski S, Debudaj-Grabysz A (2015) KMC 2: fast and resource-frugal k-mer counting. Bioinformatics 31(10):1569–1576
7. Grabowski S, Raniszewski M (2015) Sampling the suffix array with minimizers. In: String processing and information retrieval. Springer, Berlin, pp 287–298
8. Huber W, Carey VJ, Gentleman R, Anders S, Carlson M, Carvalho BS, Bravo HC, Davis S, Gatto L, Girke T, et al (2015) Orchestrating high-throughput genomic analysis with Bioconductor. Nat Methods 12(2):115
9. Jain C, Dilthey A, Koren S, Aluru S, Phillippy A (2017) A fast approximate algorithm for mapping long reads to large reference databases. In: Sahinalp S (ed) Research in computational molecular biology. RECOMB 2017. Lecture notes in computer science, vol 10229. Springer, Berlin, pp 66–81
10. Kucherov G (2019) Evolution of biosequence search algorithms: a brief survey. Bioinformatics 35(19):3547–3552
11. Li Y, Yan X (2015) MSPKmerCounter: a fast and memory efficient approach for k-mer counting. Preprint. arXiv:1505.06550
12. Li Y, Kamousi P, Han F, Yang S, Yan X, Suri S (2013) Memory efficient minimum substring partitioning. In: Proceedings of the VLDB endowment, vol 6, pp 169–180. VLDB Endowment
13. Marçais G, Pellow D, Bork D, Orenstein Y, Shamir R, Kingsford C (2017) Improving the performance of minimizers and winnowing schemes. Bioinformatics 33(14):i110–i117
14. Marçais G, Solomon B, Patro R, Kingsford C (2019) Sketching and sublinear data structures in genomics. Annu Rev Biomed Data Sci 2:93–118
15. Mykkeltveit J (1972) A proof of Golomb's conjecture for the de Bruijn graph. J Comb Theory Ser B 13(1):40–45
16. Ondov BD, Treangen TJ, Melsted P, Mallonee AB, Bergman NH, Koren S, Phillippy AM (2016) Mash: fast genome and metagenome distance estimation using MinHash. Genome Biol 17(1):132
17. Orenstein Y, Pellow D, Marçais G, Shamir R, Kingsford C (2016) Compact universal k-mer hitting sets. In: International workshop on algorithms in bioinformatics. Springer, Berlin, pp 257–268
18. Orenstein Y, Pellow D, Marçais G, Shamir R, Kingsford C (2017) Designing small universal k-mer hitting sets for improved analysis of high-throughput sequencing. PLoS Comput Biol 13(10):e1005777
19. Paindavoine M, Vialla B (2015) Minimizing the number of bootstrappings in fully homomorphic encryption. In: International conference on selected areas in cryptography. Springer, Berlin, pp 25–43
20. Pearson WR (2013) An introduction to sequence similarity ("homology") searching. Curr Protoc Bioinf 42(1):1–3
21. Reuter JA, Spacek DV, Snyder MP (2015) High-throughput sequencing technologies. Mol Cell 58(4):586–597
22. Roberts M, Hayes W, Hunt BR, Mount SM, Yorke JA (2004) Reducing storage requirements for biological sequence comparison. Bioinformatics 20(18):3363–3369
23. Roberts M, Hunt BR, Yorke JA, Bolanos RA, Delcher AL (2004) A preprocessor for shotgun assembly of large genomes. J Comput Biol 11 (4):734–752
24. Rodríguez-Ezpeleta N, Hackenberg M, Aransay AM (2011) Bioinformatics for high throughput sequencing. Springer Science & Business Media, Berlin
25. Wood DE, Salzberg SL (2014) Kraken: ultrafast metagenomic sequence classification using exact alignments. Genome Biol 15(3):R46
26. Ye C, Ma ZS, Cannon CH, Pop M, Douglas WY (2012) Exploiting sparseness in de novo genome assembly. BMC Bioinf 13(6):S1

Chapter 6

An Introduction to Whole-Metagenome Shotgun Sequencing Studies

Tyler A. Joseph and Itsik Pe'er

Abstract

Microbial communities are found across diverse environments, including within and across the human body. As many microbes are unculturable in the lab, much of what is known about a microbiome—a collection of bacteria, fungi, archaea, and viruses inhabiting an environment—-is from the sequencing of DNA from within the constituent community. Here, we provide an introduction to whole-metagenome shotgun sequencing studies, a ubiquitous approach for characterizing microbial communities, by reviewing three major research areas in metagenomics: assembly, community profiling, and functional profiling. Though not exhaustive, these areas encompass a large component of the metagenomics literature. We discuss each area in depth, the challenges posed by whole-metagenome shotgun sequencing, and approaches fundamental to the solutions of each. We conclude by discussing promising areas for future research. Though our emphasis is on the human microbiome, the methods discussed are broadly applicable across study systems.

Key words Whole metagenome, Shotgun sequencing, Microbial communities, Microbiome

1 Introduction

The inception of the Human Microbiome Project (HMP) in 2007 [1] marked a new era in genomics. At the time, it had long been recognized that microbial cells in the human body outnumber human ones [2–4], health and that microbial communities play an important role in human health and disease [1]. Yet, further characterization of the genomic content of microbial communities had remained difficult, because most microbes cannot be cultivated in laboratory settings required for traditional approaches to whole-genome sequencing [5]. Two landmark studies in 2004 provided a way around this limitation: Venter et al. [6] and Tyson et al. [7] applied shotgun sequencing techniques—typically used to reconstruct the genomes of single organisms—to environmental samples in order to generate random fragments from the genomes of *all* microbes in a community. This new technique, termed whole-

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_6,

metagenome shotgun sequencing, created an opportunity to gain a detailed, mechanistic view into microbial communities. The first metagenomic study of the human gut microbiome followed shortly afterward [8], along with a realization that the genes encoded by human-associated microbial communities far outnumbered those in the human genome itself [9], and the viewpoint of a "human genetic landscape" [1] composed of the human genome with genomes of all associated species. Indeed, Bäckhed et al. [10] argued that cataloging the gene content of human-associated microbial communities was a natural extension to the then recently completed Human Genome Project. Thus, the beginning of HMP marked a synthesis of these ideas, and initiated the first large-scale effort to catalog constituents of the human microbiome and their genetic landscape.

Since then, microbiome-focused sequencing studies have expanded our knowledge of the structure and function of microbial taxa and their communities. The first [11] and second [12] phases of HMP collectively generated more than 2000 reference genomes from human body sites, and the complementary MetaHIT project [13] identified over 9.75 million nonredundant microbial genes in the human gut [14]. From these and related efforts, we now know that human-associated communities display substantial variation across body sites, but less variability across individuals at a body site [15], suggesting that community assembly is nonrandom. Further, imbalances in microbial taxa in gut communities have been associated with several diseases [16]. Still, the current catalog of the genetic landscape of microbial communities is far from complete. For instance, a more recent study from 2019 [17] generated metagenomic assemblies for more than 150,000 genomes representing approximately 5000 species—77% of which were novel.

Each of these examples highlights the fundamental aims of metagenomics: community profiling, functional profiling, and metagenomic assembly (genome reconstruction). Both community profiling and functional profiling are limited by the availability of reference genomes, which for many microbes must be reconstructed from metagenomic samples themselves. Thus, metagenomic assembly, community profiling, and functional profiling form the core of many metagenomic studies. Nonetheless, analysis of metagenomic samples is challenging because metagenomic datasets consist of random fragments of all the DNA in a sample. A useful mental model is to conceptualize the sampling process as selecting random segments of the genomes of each microbe in a sample uniformly at random: the number of sequencing reads from the genome of one species is proportional to the length of its genome and the fraction of that species in the community [6]. It is the random nature of sequencing reads not only along the

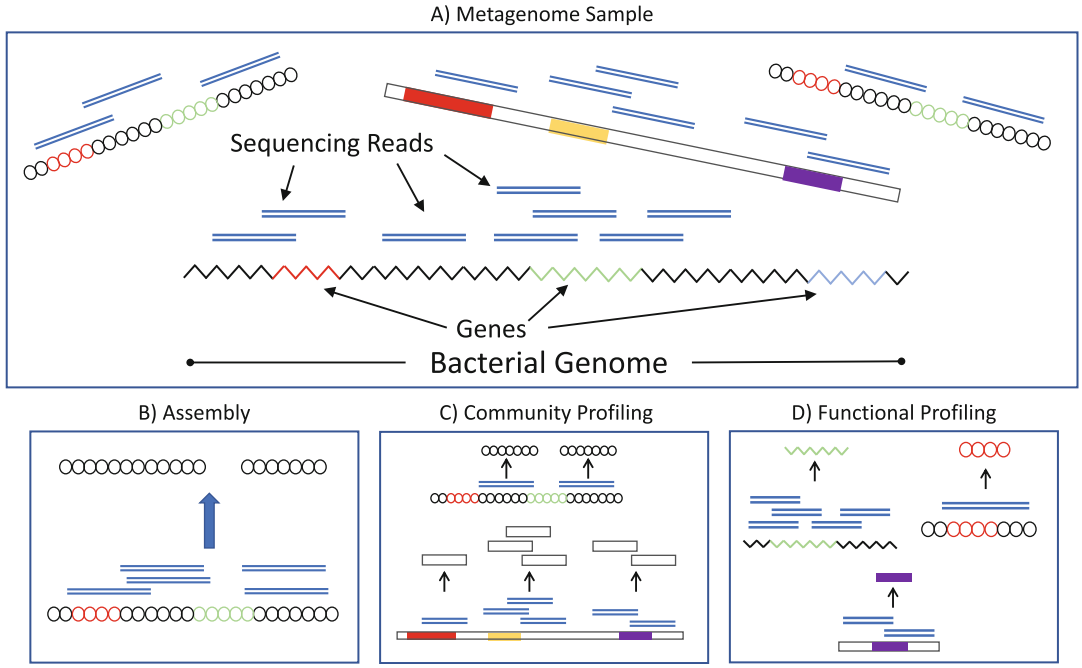


Fig. 1 Illustration of the major components of whole-metagenome shotgun sequencing studies. (**a**) The sampling process for four (linearized) bacterial genomes across three taxa. (**b**) Illustration of genome assembly from sequencing reads. (**c**) Illustration of community profiling, the determination of taxa in a community. (**d**) Illustration of functional profiling, the determination of the functional content in a community

genome, but among taxa, that presents challenges to assembly, community profiling, and functional profiling.

In this review, we provide an introduction to whole-metagenome shotgun sequencing through the problems of metagenomic assembly, community profiling, and functional profiling (Fig. 1). While these are not the only areas of research—metagenomics is a large field spanning multiple areas of genomics—many studies incorporate some or all of these elements. Our goal is to provide new entrants to the field a broad overview, a solid foundation for deeper study, and the ability to quickly place current research in context. We discuss each problem in depth, the challenges presented by metagenomic sampling, and elements fundamental to the solutions of each from the current state of the art. We conclude by discussing some promising new developments in metagenomics.

2 Metagenomic Assembly

Characterizing the complete genetic landscape of microbial communities requires the genomes of its constituents. Thus genome assembly—genome reconstruction from DNA sequencing data—is

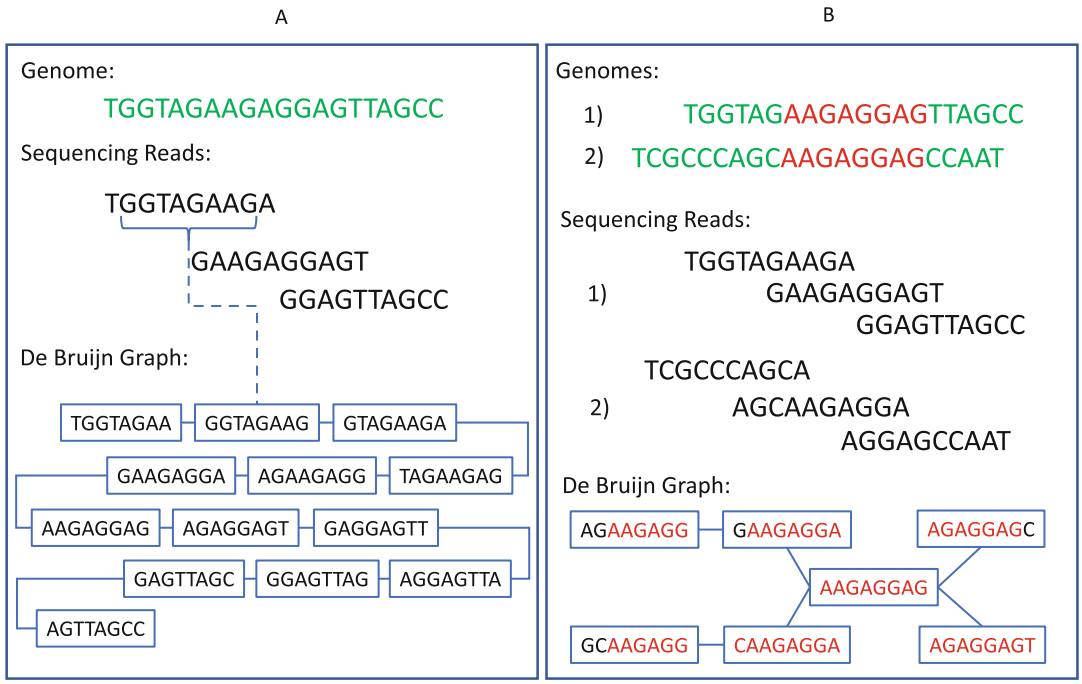


Fig. 2 Single-organism and metagenomic assembly. (**a**) Construction of a de Bruijn graph for single-organism assembly. Sequencing reads are broken down into subsequences of *k*-mers (8-mers) which form the nodes of the graph. Two *k*-mers are connected by an edge of they share a subsequence of 7 nucleotides. Traversing the graph reconstructs the genome. (**b**) Illustration of one of the challenges in metagenome assembly. Two or more genomes share a subsequence (red) that is longer than the *k*-mer length (perhaps by horizontal gene transfer) that leads to an ambiguous de Bruijn graph, and possibly a chimeric assembly

a major area of research in metagenomics. Databases of reference genomes are the crux of many downstream studies: they are a valuable resource for community and functional profiling. For instance, reference genomes are required to accurately assign short reads from DNA sequencers to species and genes [1]. Yet, assembling genomes from metagenomic samples is challenging. In single-organism assemblies, sequencing reads originate from the same genome; the assembly problem is a matter of placing sequencing reads in the right order (Fig. 2). In metagenomic samples, however, sequencing reads originate from many different genomes. Metagenomic assemblers need to disambiguate reads from multiple species or related strains.

Assembly has been intensively studied, originally motivated by single genomes sequenced deliberately to create a reference. The dominant paradigm for assembly revolves around the use of short-read sequencing technologies and de Bruijn graphs [18–20]. In de Bruin graph approaches, sequencing reads are broken down into overlapping subsequences of k nucleotides called k-mers. Then, overlapping k-mers are stitched together to reconstruct a genome.

Formally, the *k*-mers serve as the vertices in a graph. An edge connects one *k*-mer to another if the $k-1$-suffix of the former is the prefix of the latter. The connected nodes thus correspond to a sequence of $k+1$ bases in the genome. If each *k*-mer corresponds to a unique position in the fully covered genome with no errors, a path through the graph that traverses each edge once corresponds exactly to the genome of interest. In practice, assembly is more complicated due to repeated regions, structural variants, polymorphisms, and sequencing errors that create ambiguities along the path that need to be resolved. Furthermore, incomplete sequence coverage results in a disconnected de Bruijn graph, with respective paths through each connected component, that when traversed produce short contiguous sequences of bases called *contigs*. Contigs can further be grouped and ordered into larger regions called *scaffolds*, for instance, by using additional information provided by paired reads in single-organism assemblies [21]. An assembly can either consist of the scaffolds themselves, or the scaffolds after an additionally processing step that attempts to order scaffolds and fill in the gaps between them.

Genome assembly from metagenomic sequencing adds an additional level of complexity because sequencing reads not only have to be placed in the correct order but assigned to the correct species, sub-species, or strain. Moreover, single-organism assemblers use heuristics to eliminate errors and improve contig quality by removing erroneous *k*-mers or graph paths based on the assumption that sequencing depth is approximately uniform along the genome [22]. However, this assumption is false for metagenomic samples, because the species in a community are not uniformly represented and thus coverage across *k*-mers is nonuniform. Furthermore, problems with repeated regions are exacerbated because repeats can both occur within a genome and across genomes [19]. For instance, the presence of closely related species, evolutionary conserved genes, or genes from horizontal gene transfer are challenging to detect during assembly, and can lead to chimeric assemblies consisting of multiple species. To mitigate this issue, several assemblers use an additional step of binning contigs into putative species based related sequencing coverage, sequence composition, or using information from paired reads, then scaffolding the contigs within a bin [20, 23]. More recent work [24] strings *k*-mers into paths whenever possible, then aligns them all-against-all to inform a *Repeat Graph* structure that disambiguates orthologous versus paralogous repeating segments. Still, none of these heuristics are perfect, and disambiguating closely related species or strains—that may only differ in a handful of genes—is incredibly challenging.

Further complicating matters is a lack of a gold standard for evaluating de novo assemblies where pre-existing reference genomes do not exist. One classical metric is N50, the contig length of

the median assembled bases [20]. Intuitively, a larger N50 means longer contigs and is indicative of reference quality. Still, N50 does not indicate whether the order of bases in a contig is correct, and, thus, a larger N50 does not always mean a better assembly (imagine arbitrarily concatenating all contigs together). Other useful statistics for evaluating assemblies are the proportion of mapped reads, reasoning that a higher fraction of mapped reads means the assembly explains more of the data, or the number of predicted genes across an assembly.

To address these issues, the Critical Assessment of Metagenome Interpretation [25] (CAMI) challenge sought to benchmark metagenomic assembly methods on high quality reference data and using the same standards of evaluation. The challenge highlighted many of the difficulties in metagenomic assembly. While the benchmarked assemblers all performed well on low complexity samples where the genomes among species were mostly unique, all had difficulty in assembling high complexity datasets with closely related species and substrains. Moreover, differences in the parameters provided to each assembler lead to substantially different quality of assembly. Thus, metagenomic assembly is still very much an open area of research. Nonetheless, assemblies from metagenomic samples are valuable resources for downstream studies.

3 Community Profiling

The goal of community profiling is to identify the taxa in a community and their abundances. Community profiling serves as a large component of many microbiome studies. Indeed, a major aim of the first phase of the HMP was to understand the range of diversity and distribution of microbial taxa in healthy individuals, and several notable discoveries have been based on examining the profiles of microbial communities. For example, microbial communities across body sites are somewhat stable when looking at bacterial phyla [26], and vary more among individuals that within an individual over time [15]. In addition, many study designs look for differences in community abundance profiles between healthy and disease states—among the most commonly cited are a shift from Bacteroidetes to Firmicutes in the gut microbiome compositions of people with obesity [27], and a decrease in community diversity in gut microbiomes of people with Crohn's disease [28].

The predominant technique for community profiling uses targeted amplicon sequencing of the 16S rRNA gene, not whole-metagenome shotgun sequencing (Fig. 3). This is sometimes also referred to as metagenomics, especially in earlier literature. However, the methodology surrounding 16S rDNA is distinct, and deserves separate classification. 16S rDNA sequencing is popular because the 16S rRNA gene is present among all prokaryotes, and it

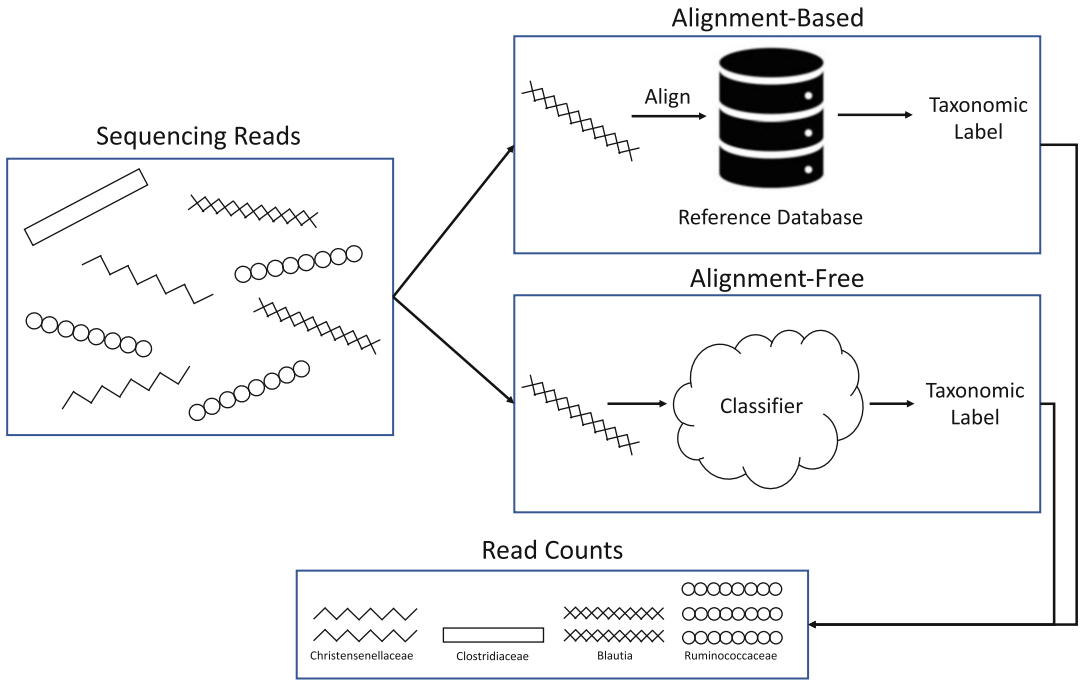


Fig. 3 Two approaches to community profiling. Alignment-based approaches query a reference database of genomes or marker genes to produce taxonomic labels for each sequence. Classifier-based approaches train a machine learning classifier on a reference database of genome by extracting features (e.g. GC-content, *k*-mer frequency) to label sequencing reads. Trained classifiers are applied on novel sequencing datasets to produce taxonomic labels for each sequence

contains conserved regions to serve as targets for PCR primers that flank variable regions useful for phylogenetic classification [29]. This facilitates cost-effective, coarse-grained profiling. Nonetheless, the resolution of 16S sequencing is often limited to genera, it will miss the small numbers of eukaryotes present in a community, and can lead to biased estimates of taxa abundances if PCR primers are not carefully chosen. Metagenomic sequencing can potentially mitigate these issues.

Community profiling from shotgun metagenomics has two related goals: detection of the taxa in a community and quantification of their abundances. Early approaches to community profiling focused on the first goal, and relied on directly aligning reads to a database of reference genomes such as NCBI RefSeq [30]. Reads are assigned a taxonomic classification based on the lowest common ancestor of the genomes matching a read. However, accurately assigning reads to species relies on the breadth of genomes in the reference. Novel species without a reference genome can only be resolved at a higher phylogenetic classification, such as genus or family, and the resolution of this assignment depends on the phylogenetic similarity to species in the reference database. A 2012 study

comparing alignment methods for taxonomic classification found that only 67% of reads mapped to the correct strain, and 21% of the remaining reads mapped to the correct genus [31]. Obviously, completeness of databases is a moving target. Read length also significantly impacts taxonomic assignment [30], with shorter read lengths (< 200 bp) resulting fewer matches against a reference database with more ambiguity [32]. This is particularly problematic because shorter reads are more cost effective. Finally, alignment methods are computationally expensive and will only become more cumbersome as the throughput of metagenomic sequencing increases (*see* Wood et el. [33] for an exception).

An alternate approach forgoes alignment and instead tries to classify reads based on sequence features such as frequencies of sequence patterns [34], *k*-mer profiles [35], or signatures from interpolated Markov models [36]. In each case, a classifier is trained on a training set of reference genomes from a database, which is then applied to assign taxonomic classifications on new sequencing datasets. Hence, alignment-free approaches are more computationally attractive, and potentially mitigate issues with incomplete reference sets. However, they all suffer from loss of fidelity with shorter sequence lengths, and do not (accurately) classify reads below the genus level. A potentially more severe problem, shared by alignment methods, is the transformation from taxonomically classified reads to taxa relative abundances (the percentage of each taxon in a sample). This requires normalization by genome size—which is unknown for novel species—as well as mappability scores reflecting conservation of some sequencing regions among species [37].

A third class of methods mitigates these problems by building curated reference sets of phylogenetically informative marker genes, synthesizing alignment-based approaches and approaches from 16S rDNA sequencing. Genes are selected based on their ability to discriminate between species. Removing uninformative regions of the genome drastically reduces the search space for aligning sequencing reads at the expense of throwing away useful data. These approaches have been shown to accurately classify reads from short-read technologies. Furthermore, the normalizing constant needed to transform read counts to relative abundance depends only on the database itself, not unknown genomes. For example, Mende et al. [38] used 3496 prokaryotic reference genomes to identify a set of 40 universal single-copy phylogenetic marker genes. They demonstrated that a 97% sequence similarity threshold could accurately delineate species with high precision ($> 98\%$) and high recall ($> 99\%$). A major benefit of universal marker genes is the ability to identify novel species because single-copy genes among the reference genomes are likely conserved. A method based on universal marker genes [39] found that 58% of species in a dataset of gut communities from HMP and MetaHIT did not have available reference genomes. These species

represented an average of ~43% observed species abundances in a sample. Yet, by limiting the search space to a set of 40 marker genes, the majority of sequencing reads are thrown away. In practice, this means that rare species are less likely to be detected, and estimates of relative abundances for rare species are more error prone.

Another approach based on marker genes builds a reference set of clade-specific markers—genes that uniquely identify a clade in a taxonomy—enabling a larger reference database retaining more sequencing reads. Segata et al. [37] built a catalog of 400,141 clade-specific marker genes resulting in an average of 231 markers across 1221 species, and more than 115,000 markers identifying higher taxonomic levels. Novel species without references are assigned to a higher taxonomic level based on their proximity to the reference. The authors demonstrated that clade-specific marker genes led to faster and more accurate estimates of species abundances than classification-based approaches that compare reads against reference genomes. Moreover, clade-specific marker genes provide a larger reference set than universal marker genes, meaning they provide more information given the same sequencing depth.

Read mapping, universal marker genes, and clade-specific marker genes each provides read counts against a reference database, but statistical analysis using read counts is deceptively challenging. Importantly, the total number of reads is a function of sequencing time. This means that read counts do not provide any information about the total size of the community, and only the relative abundances (the percentages) of each taxon can be recovered. Even after transforming read counts to relative abundances, caution must be taken when applying standard statistical techniques. This is because relative abundances are in a constrained space—the components must sum to one—which induces a negative correlation between observed abundances. Applying standard statistical tools (for instance, computing the correlation between two species) to relative abundance data can lead to spurious associations [40–42]. Increasingly, the field is moving towards techniques from compositional data analysis [43–45], which specifically addresses challenges posed by relative abundance data. However, compositional data analysis techniques have yet to achieve widespread adoption, likely because compositional data analysis falls outside standard curricula for statistical training.

4 Functional Profiling

Community profiling tells us which taxa are in a community, but gives little information about what they are doing. Functional profiling (Fig. 4) of the genes in community provides knowledge about its biological role—the potential proteins encoded and their molecular function. In general, functional profiling studies fall into

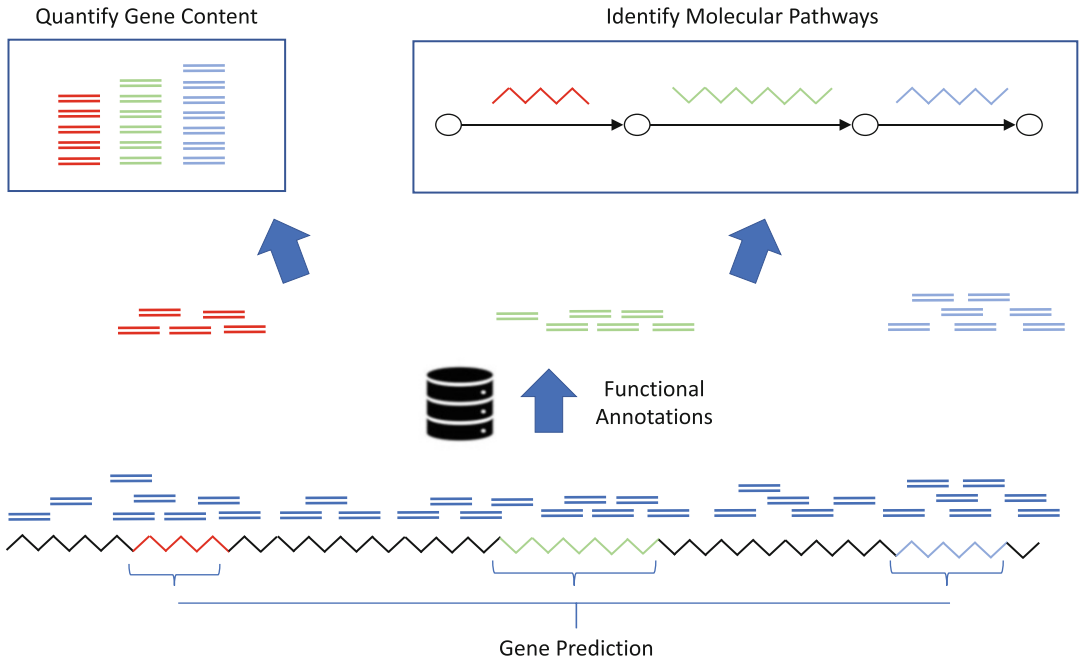


Fig. 4 Illustration of four aspects of functional profiling. Bottom: Gene predictions algorithms classify open reading frames (colored segments) in a assembly (zig-zag line). Middle: Querying sequencing reads against a knowledge based of functional annotations recovers the gene content in a sample. Top: Functional annotations can be used to quantify gene content (left), or reconstruct metabolic pathways (right)

two categories: upstream studies focused on resource generation for the broader community, and downstream studies which uses these resources on particular systems of interest. Upstream studies focus on annotating genes in a genome by identifying open reading frames, which are combined with a knowledge base of functional annotations that characterize biological processes. Downstream studies use such resources to obtain counts of functional units (e.g. genes). For example, downstream studies can look for molecular targets for drug design, or characterize differential gene content between healthy and disease states. Genome annotation and functional annotation are not new problems, and earlier metagenomic studies relied on tooling developed with single-organism samples in mind. However, new developments in these areas have been driven by the challenges of metagenomic data.

For example, the MetaHIT Consortium's goal was to develop a catalog of nonredundant genes across all human gut-associated microbiota [13, 14]. To achieve this, they generated metagenomic sequences across a large cohort, assembled them into contigs, and used standard algorithms for gene prediction to identify open reading frames. These open reading frames were further clustered by sequence similarity to create a nonredundant gene set. The most recent iteration of the MetaHIT project combined metagenomic

sequences from three globally distributed cohorts with previous characterized genomes of gut microbes, and covered an estimated 94.5% of the gene content (>9.5 million genes) in the gut microbiome [14]. Nonetheless, accurately predicting open reading frames in a metagenomic sample is challenging because single sequencing reads likely do not span an entire gene, and generating contigs required for more accurate gene prediction is difficult for problems discussed in the section on Assembly.

Assigning genes to biological functions relies on carefully curated reference databases that draw from the scientific literature. For instance, the UniProt [46] database contains specific protein sequences and functional annotations from the published literature. However, such annotation requires specific functional characterization of a protein from a published study. For the vast majority of the >9.5 million genes in the gut microbiome, such detailed annotation is unfeasible.

Instead, function for uncharacterized proteins is inferred from homology to characterized ones. An early and general example of that involves the Pfam [47] database focuses on groups of closely related protein families. These databases provide search functionality either based on a variation of BLAST or specialized hidden-Markov model. More specifically, orthologous genes [48], homologs that descended from the same gene through a speciation event, are likely to retain their function; hence, knowledge of the function of one ortholog informs the others [49]. The COG [49], eggNOG [48], and KEGG Orthology [50] databases are commonly used orthology databases for metagenomic studies. Each uses a heuristic to cluster groups of genes into putative orthologs, and assigns functions to each cluster. The MetaHIT project used both the KEGG Orthology and eggNOG databases to annotate genes.

The COG (Clusters of Orthologous Groups) database is a hand-curated database of functionally annotated orthologous gene clusters. The most recent update [51] was specifically tailored to microbial sequencing studies. It contains 4631 functionally annotated COGs from 46 bacterial, 13 archaeal, and 3 eukaryotic genomes. One benefit of the COG database is that COGs are organized by 26 broader function categories providing a high level overview of the major biological processes within a community. Similarly, the eggNOG (evolutionary genealogy of genes: Non-supervised Orthologous Groups) database uses the same principles as COG, but relies on automated annotation instead of manual curation. It, therefore, contains a larger repertoire of orthologous groups.

The KEGG (Kyoto Encyclopedia of Genes and Genomes) Orthology database also provides automatically generated clusters of orthologous genes [50]. Orthologous genes are given unique identifiers called KO numbers, collections of which can be used to

identify hand-curated molecular pathways (KEGG Pathways) representing networks of functional significance (e.g. protein–protein interactions). Moreover, records in KEGG are organized hierarchically, allowing different granularity of annotations. KEGG provides specific tools to automatically annotate metagenomic sequences [52].

Nonetheless, use of functional annotation databases has several caveats. For instance, factors such as read length can affect the quality and number of annotations in a sample [53]. For this reason, there has been interest in developing specialized pipelines for functional annotation from metagenomic samples [54, 55]. These pipelines leverage additional, metagenomic specific, such as the species [54] present and their annotated pan genomes [55], and can impute gaps in functional pathways. Still, annotation databases are limited by the breadth of functional annotations, and data from less well-studied systems are likely to have fewer functional predictions [53]. In practice, this means that common housekeeping functions, functions shared by many taxa, and functions of human-associated communities, are more likely to be annotated.

In principle many of these databases present the same information in different ways, and many rely on each other for their annotation schemes. The output of all these analyses are all matrices of samples by a specific, database dependent, functional unit (e.g. genes, proteins, protein families, COG categories, KO numbers, KEGG pathways). This means that some of the caveats about compositional data from community profiling apply here. However, in many cases standard statistical tools are applied for downstream analysis [56].

5 Future Perspectives

As metagenomic sequencing matures, research has moved beyond these three fundamental domains. Increasingly, researchers are taking a holistic view toward microbiome studies by integrating multi-omics with metagenomics [57, 58]. Alternate technologies, such as metabolomics, metaproteomics, and metatranscriptomics, measure the products of active biological processes, that when combined with metagenomics provide a view into the interactions between microbiomes and their environments. For instance, Olson et al. [59] combined 16S sequencing with metabolomics to show that the gut microbiome mediates the protective effect of the ketogenic diet on refractory epilepsy by changing the balance of neurotransmitters in the brain. Lloyd-Price et al. [60] took longitudinal multi-omic measurements from a cohort with Crohn's disease, and demonstrated that shifts in community composition during active

periods of the disease are associated with disruptions in normal molecular activities.

There has also been an interest using metagenomics to investigate microbial temporal dynamics. Using time-series models and longitudinal sampling, Buffie et al. [61] showed that *C. scindens* provided resistance to *C. difficile* infection, and identified the resistance mechanism by inspecting the gene repertoire of *C. scindens*. Time-series modeling also has promising applications in precision medicine, where temporal models could be used to predict health outcomes and design bacteriotherapies. Furthermore, single metagenomic samples have been shown to encode information about bacterial growth. Because bacterial DNA replication (usually) proceeds bidirectionally from a single replication origin, in rapidly dividing populations the more sequencing reads are expected near the replication origin than the terminus. Korem et al. [62] showed that the ratio of sequencing reads between the replication origin and terminus—termed the peak-to-trough (PTR) ratio—is correlated with growth rates, and that PTRs are associated with several metabolic disorders.

Lastly, as techniques for read mapping improve, along with the quality of reference genomes, so does the potential to identify changes in the gene content within their genetic context. Zeevi et al. [63] showed that structural variations, deletions and copy number variations encompassing multiple nearby genes in a genome, play an important role in factors of disease risk. For example, they found a complete metabolic pathway in the genome of *Anaerostipes hadrus* that when present reduces risk for metabolic disease.

Each of these areas holds promising opportunities to better understand microbial communities.

6 Conclusion

Metagenomic studies over the last decade broadly involved three major research paradigms: metagenomic assembly, the construction of reference genomes; community profiling, which determines the taxa present and their proportions; and functional profiling, which determines the genes encoded in the community and their biochemical potential. Each has motivated methodological advances that provide valuable resources to investigate microbial communities in depth, and the relationship between the microbiome and its environment. In humans, metagenomics holds promise to elucidate the mechanisms of complex diseases and their treatment. As these three paradigms matured, they have opened the door to more sophisticated methodologies. High quality reference genomes can pinpoint genes responsible for drug resistance; accurate community profiling facilitates the use of more advanced statistical procedures;

and, improved functional annotation can highlight drug targets for pharmaceuticals. Each increases our understanding of the human genetic landscape, microbial organisms and all.

References

1. Turnbaugh PJ, Ley RE, Hamady M, Fraser-Liggett CM, Knight R, Gordon, JI (2007) The human microbiome project. Nature 449 (7164):804
2. Luckey TD (1972) Introduction to intestinal microecology. Am J Clin Nutr 25:1292–1294
3. Berg RD (1996) The indigenous gastrointestinal microflora. Trends Microbiol 4 (11):430–435
4. Sender R, Fuchs S, Milo R (2016) Revised estimates for the number of human and bacteria cells in the body. PLoS Biol 14(8): e1002533
5. Wooley JC, Godzik A, Friedberg I (2010) A primer on metagenomics. PLoS Comput Biol 6 (2):e1000667
6. Venter JC, Remington K, Heidelberg JF, Halpern AL, Rusch D, Eisen JA, Wu D, Paulsen I, Nelson KE, Nelson W et al (2004) Environmental genome shotgun sequencing of the Sargasso Sea. Science 304(5667):66–74
7. Tyson GW, Chapman J, Hugenholtz P, Allen EE, Ram RJ, Richardson PM, Solovyev VV, Rubin EM, Rokhsar DS, Banfield JF (2004) Community structure and metabolism through reconstruction of microbial genomes from the environment. Nature 428(6978):37
8. Eckburg PB, Bik EM, Bernstein CN, Purdom E, Dethlefsen L, Sargent M, Gill SR, Nelson KE, Relman DA (2005) Diversity of the human intestinal microbial flora. Science 308(5728):1635–1638
9. Ley RE, Peterson DA, Gordon JI (2006) Ecological and evolutionary forces shaping microbial diversity in the human intestine. Cell 124 (4):837–848
10. Bäckhed F, Ley RE, Sonnenburg JL, Peterson DA, Gordon JI (2005) Host-bacterial mutualism in the human intestine. Science 307 (5717):1915–1920
11. Huttenhower C, Gevers D, Knight R, Abubucker S, Badger JH, Chinwalla AT, Creasy HH, Earl AM, FitzGerald MG, Fulton RS et al (2012) Structure, function and diversity of the healthy human microbiome. Nature 486(7402):207
12. Lloyd-Price J, Mahurkar A, Rahnavard G, Crabtree J, Orvis J, Hall AB, Brady A, Creasy HH, McCracken C, Giglio MG et al (2017) Strains, functions and dynamics in the expanded human microbiome project. Nature 550(7674):61
13. Qin J, Li R, Raes J, Arumugam M, Burgdorf KS, Manichanh C, Nielsen T, Pons N, Levenez F, Yamada T et al (2010) A human gut microbial gene catalogue established by metagenomic sequencing. Nature 464 (7285):59
14. Li J, Jia H, Cai X, Zhong H, Feng Q, Sunagawa S, Arumugam M, Kultima JR, Prifti E, Nielsen T et al (2014) An integrated catalog of reference genes in the human gut microbiome. Nat Biotechnol 32(8):834
15. Costello EK, Lauber CL, Hamady M, Fierer N, Gordon JI, Knight R (2009) Bacterial community variation in human body habitats across space and time. Science 326 (5960):1694–1697
16. Clemente JC, Ursell LK, Parfrey LW, Knight R (2012) The impact of the gut microbiota on human health: an integrative view. Cell 148 (6):1258–1270
17. Pasolli E, Asnicar F, Manara S, Zolfo M, Karcher N, Armanini F, Beghini F, Manghi P, Tett A, Ghensi P et al (2019) Extensive unexplored human microbiome diversity revealed by over 150,000 genomes from metagenomes spanning age, geography, and lifestyle. Cell 176(3):649–662
18. Pevzner PA, Tang H, Waterman MS (2001) An Eulerian path approach to DNA fragment assembly. Proc Natl Acad Sci 98 (17):9748–9753
19. Olson ND, Treangen TJ, Hill CM, Cepeda-Espinoza V, Ghurye J, Koren S, Pop M (2017) Metagenomic assembly through the lens of validation: recent advances in assessing and improving the quality of genomes assembled from metagenomes. Brief Bioinform 20 (4):1140–1150
20. Ayling M, Clark MD, Leggett RM (2019) New approaches for metagenome assembly with short reads. Brief Bioinform 21:584–594
21. Li R, Zhu H, Ruan J, Qian W, Fang X, Shi Z, Li Y, Li S, Shan G, Kristiansen K et al (2010) De novo assembly of human genomes with massively parallel short read sequencing. Genome Res 20(2):265–272
22. Peng Y, Leung HCM, Yiu S-M, Chin FYL (2012) IDBA-UD: a de novo assembler for

single-cell and metagenomic sequencing data with highly uneven depth. Bioinformatics 28 (11):1420–1428

23. Nurk S, Meleshko D, Korobeynikov A, Pevzner PA (2017) metaspades: a new versatile metagenomic assembler. Genome Res 27 (5):824–834
24. Kolmogorov M, Rayko M, Yuan J, Polevikov E, Pevzner P (2019) metaFlye: scalable long-read metagenome assembly using repeat graphs. bioRxiv, p 637637
25. Sczyrba A, Hofmann P, Belmann P, Koslicki D, Janssen S, Dröge J, Gregor I, Majda S, Fiedler J, Dahms E et al (2017) Critical assessment of metagenome interpretation-a benchmark of metagenomics software. Nat Methods 14(11):1063
26. Cho I, Blaser MJ (2012) The human microbiome: at the interface of health and disease. Nat Rev Genet 13(4):260
27. Turnbaugh PJ, Ley RE, Mahowald MA, Magrini V, Mardis ER, Gordon JI (2006) An obesity-associated gut microbiome with increased capacity for energy harvest. Nature 444(7122):1027
28. Manichanh C, Rigottier-Gois L, Bonnaud E, Gloux K, Pelletier E, Frangeul L, Nalin R, Jarrin C, Chardon P, Marteau P et al (2006) Reduced diversity of faecal microbiota in Crohn's disease revealed by a metagenomic approach. Gut 55(2):205–211
29. Kuczynski J, Lauber CL, Walters WA, Parfrey LW, Clemente JC, Gevers D, Knight R (2012) Experimental and analytical tools for studying the human microbiome. Nat Rev Genet 13 (1):47
30. Huson DH, Auch AF, Qi J, Schuster SC (2007) Megan analysis of metagenomic data. Genome Res 17(3):377–386
31. Martin J, Sykes S, Young S, Kota K, Sanka R, Sheth N, Orvis J, Sodergren E, Wang Z, Weinstock GM et al (2012) Optimizing read mapping to reference genomes to determine composition and species prevalence in microbial communities. PLoS One 7(6):e36427
32. Wommack KE, Bhavsar J, Ravel J (2008) Metagenomics: read length matters. Appl Environ Microbiol 74(5):1453–1463
33. Wood DE, Salzberg SL (2014) Kraken: ultrafast metagenomic sequence classification using exact alignments. Genome Biol 15(3):R46
34. McHardy AC, Martín HG, Tsirigos A, Hugenholtz P, Rigoutsos I (2007) Accurate phylogenetic classification of variable-length DNA fragments. Nat Methods 4(1):63
35. Rosen G, Garbarine E, Caseiro D, Polikar R, Sokhansanj B (2008) Metagenome fragment classification using n-mer frequency profiles. Adv Bioinform 2008:205969
36. Brady A, Salzberg SL (2009) Phymm and phymmbl: metagenomic phylogenetic classification with interpolated Markov models. Nat Methods 6(9):673
37. Segata N, Waldron L, Ballarini A, Narasimhan V, Jousson O, Huttenhower C (2012) Metagenomic microbial community profiling using unique clade-specific marker genes. Nat Methods 9(8):811
38. Mende DR, Sunagawa S, Zeller G, Bork P (2013) Accurate and universal delineation of prokaryotic species. Nat Methods 10(9):881
39. Sunagawa S, Mende DR, Zeller G, Izquierdo-Carrasco F, Berger SA, Kultima JR, Coelho LP, Arumugam M, Tap J, Nielsen HB et al (2013) Metagenomic species profiling using universal phylogenetic marker genes. Nat Methods 10 (12):1196
40. Gloor GB, Wu JR, Pawlowsky-Glahn V, Egozcue JJ (2016) It's all relative: analyzing microbiome data as compositions. Ann Epidemiol 26 (5):322–329
41. Tsilimigras MCB, Fodor AA (2016) Compositional data analysis of the microbiome: fundamentals, tools, and challenges. Ann Epidemiol 26(5):330–335
42. Gloor GB, Macklaim JM, Pawlowsky-Glahn V, Egozcue JJ (2017) Microbiome datasets are compositional: and this is not optional. Front Microbiol 8:2224
43. Fernandes AD, Reid JNS, Macklaim JM, McMurrough TA, Edgell DR, Gloor GB (2014) Unifying the analysis of high-throughput sequencing datasets: characterizing RNA-seq, 16s rRNA gene sequencing and selective growth experiments by compositional data analysis. Microbiome 2(1):15
44. Kurtz ZD, Müller CL, Miraldi ER, Littman DR, Blaser MJ, Bonneau RA (2015) Sparse and compositionally robust inference of microbial ecological networks. PLoS Comput Biol 11(5):e1004226
45. Silverman JD, Washburne AD, Mukherjee S, David LA (2017) A phylogenetic transform enhances analysis of compositional microbiota data. Elife 6:e21887
46. Apweiler R, Bairoch A, Wu CH, Barker WC, Boeckmann B, Ferro S, Gasteiger E, Huang H, Lopez R, Magrane M et al (2004) UniProt: the universal protein knowledgebase. Nucleic Acids Res 32(suppl 1):D115–D119
47. Bateman A, Birney E, Cerruti L, Durbin R, Etwiller L, Eddy SR, Griffiths-Jones S, Howe KL, Marshall M, Sonnhammer ELL (2002)

The Pfam protein families database. Nucleic Acids Res 30(1):276–280

48. Jensen LJ, Julien P, Kuhn M, von Mering C, Muller J, Doerks T, Bork P (2007) eggnog: automated construction and annotation of orthologous groups of genes. Nucleic Acids Res 36(suppl 1):D250–D254
49. Tatusov RL, Koonin EV, Lipman DJ (1997) A genomic perspective on protein families. Science 278(5338):631–637
50. Kanehisa M, Sato Y, Kawashima M, Furumichi M, Tanabe M (2015) KEGG as a reference resource for gene and protein annotation. Nucleic Acids Res 44(D1):D457–D462
51. Galperin MY, Makarova KS, Wolf YI, Koonin EV (2014) Expanded microbial genome coverage and improved protein family annotation in the cog database. Nucleic Acids Res 43(D1): D261–D269
52. Kanehisa M, Sato Y, Morishima K (2016) BlastKOALA and GhostKOALA: KEGG tools for functional characterization of genome and metagenome sequences. J Mol Biol 428 (4):726–731
53. Prakash T, Taylor TD (2012) Functional assignment of metagenomic data: challenges and applications. Brief Bioinform 13 (6):711–727
54. Abubucker S, Segata N, Goll J, Schubert AM, Izard J, Cantarel BL, Rodriguez-Mueller B, Zucker J, Thiagarajan M, Henrissat B et al (2012) Metabolic reconstruction for metagenomic data and its application to the human microbiome. PLoS Comput Biol 8(6): e1002358
55. Franzosa EA, McIver LJ, Rahnavard G, Thompson LR, Schirmer M, Weingart G, Lipson KS, Knight R, Caporaso JG, Segata N et al (2018) Species-level functional profiling of metagenomes and metatranscriptomes. Nat Methods 15(11):962
56. Quince C, Walker AW, Simpson JT, Loman NJ, Segata N (2017) Shotgun metagenomics, from sampling to analysis. Nat Biotechnol 35 (9):833
57. Integrative HMP (2014) The integrative human microbiome project: dynamic analysis of microbiome-host omics profiles during periods of human health and disease. Cell Host Microbe 16(3):276
58. Integrative HMP (2019) The integrative human microbiome project. Nature 569 (7758):641
59. Olson CA, Vuong HE, Yano JM, Liang QY, Nusbaum DJ, Hsiao EY (2018) The gut microbiota mediates the anti-seizure effects of the ketogenic diet. Cell 173(7):1728–1741
60. Lloyd-Price J, Arze C, Ananthakrishnan AN, Schirmer M, Avila-Pacheco J, Poon TW, Andrews E, Ajami NJ, Bonham KS, Brislawn CJ et al (2019) Multi-omics of the gut microbial ecosystem in inflammatory bowel diseases. Nature 569(7758):655
61. Buffie CG, Bucci V, Stein RR, McKenney PT, Ling L, Gobourne A, No D, Liu H, Kinnebrew M, Viale A et al (2015) Precision microbiome reconstitution restores bile acid mediated resistance to clostridium difficile. Nature 517(7533):205
62. Korem T, Zeevi D, Suez J, Weinberger A, Avnit-Sagi T, Pompan-Lotan M, Matot E, Jona G, Harmelin A, Cohen N et al (2015) Growth dynamics of gut microbiota in health and disease inferred from single metagenomic samples. Science 349(6252):1101–1106
63. Zeevi D, Korem T, Godneva A, Bar N, Kurilshikov A, Lotan-Pompan M, Weinberger A, Fu J, Wijmenga C, Zhernakova A et al (2019) Structural variation in the gut microbiome associates with host health. Nature 568(7750):43

Chapter 7

Microbiome Analysis Using 16S Amplicon Sequencing: From Samples to ASVs

Amnon Amir

Abstract

In this chapter, we will present an outline of a typical experimental and bioinformatic workflow for identification of bacterial amplicon sequence variants (ASVs) present in a set of samples. This chapter is written from a bioinformatic point of view; therefore, the specific experimental protocols are not detailed, but rather the impact of various experimental decisions on the downstream analysis is described. Emphasis is made on the transition from reads to ASVs, describing the Deblur algorithm.

Key words Bioinformatics, Microbiome, Amplicon sequencing

1 Introduction

A common goal in microbiome studies is identifying the bacteria present in a set of samples. High-throughput microbial amplicon sequencing has proven as a widely used and relatively inexpensive method for achieving this goal. The bacteria identified can then be used for comparing the bacterial composition across different conditions (i.e., sick vs. healthy people, etc.). In order to identify the bacteria, a fragment of DNA is amplified from all bacteria in the sample, and then sequenced using deep sequencing. The 16S rRNA gene has been the gene of choice for such experiments [1], as it is a universal gene present in all bacteria, containing both conserved regions (that can be used for almost universal amplification) and variable regions (enabling discrimination between different bacteria). Figure 1 shows a typical amplicon experiment pipeline, starting with samples collected from various sources, that are processed experimentally (a–c), sequenced (d), and finally processed computationally (e, f) resulting in a Sample X ASV Table (g), which can then be used for various downstream statistical analysis methods (h). This chapter discusses various experimental and bioinformatic details that can affect the ability of such experiments to correctly

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_7,

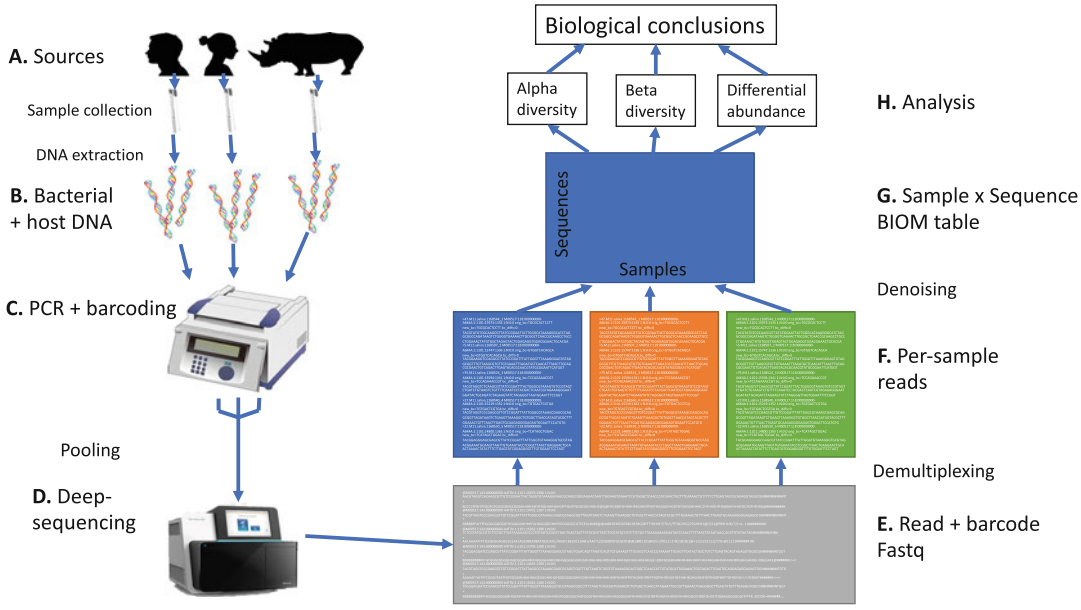


Fig. 1 Microbiome analysis using 16S Amplicon Sequencing. Samples are collected from different sources (**a**), and DNA is extracted. The resulting DNA (**b**) is amplified using primers amplifying part of the 16S rRNA gene, combined with a unique DNA barcode for each sample (**c**). Amplicons are pooled and sequenced using deep sequencing (**d**). The resulting reads (**e**) are split based on the barcodes to per-sample reads (**f**). The reads are then denoised to remove PCR and sequencing errors, and the resulting cleaned per-sample ASVs are joined to a single biom Table (**g**). This table can then be analyzed using various methods (**h**) in order to obtain biological insights

identify the bacteria in each sample, ideally resulting in novel insights into the underlying biology.

In this chapter we will use a simple scenario (which is a common scenario to many real-life microbiome experiments): we have a disease where we suspect a microbial involvement, and would like to validate the connection between the bacterial population and the disease, as well as identify specific bacteria involved. We will denote by H and S the healthy individuals and sick individuals respectively. However, when implementing such an experiment, various factors can affect the ability of our experiment to answer the questions. These include experimental design details such as the number of H and S samples to collect, the PCR primers and number of amplification cycles, the number of samples per run, etc., as well as bioinformatic details such as the denoising pipeline to use and effect of the various denoising parameters. This chapter will detail various experimental design and bioinformatic decisions, and their potential impact on the downstream results.

The bioinformatic processing described here uses denoising algorithms to obtain a table of ASVs present in each sample. These ASVs are the exact amplicon sequences of bacteria present in each sample and as such provide several advantages [2] over the

traditional closed/open-reference and de novo clustering methods [3]. First of all, ASVs provide single-nucleotide resolution over the sequenced region and therefore, in most cases, can separate two bacteria that are different in a single nucleotide (as long as they are present in comparable frequencies—as described in this chapter), as opposed to the 97% similarity required for clustering-based methods. Second, denoising based ASVs are nonsubjective, as the resulting sequences do not depend on any external database (such as used in closed-reference). Finally, each ASVs does not depend on the other sequences in the experiment, and therefore enable direct comparison between different experiments (as opposed to open-reference/de novo clustering, that require all samples compared to be processed together to enable comparison between samples).

2 Materials

1. The basic experimental pipeline in this chapter follows the Earth Microbiome Project (EMP) workflow [4], which provides a well-tested experimental pipeline for bacterial amplicon sequencing (Fig. 1). The complete EMP protocol can be found at http://www.earthmicrobiome.org/protocols-and-standards/
2. Qiime2 [5]—A comprehensive microbiome bioinformatic toolbox that includes plugins for the Deblur and DADA2 denoising algorithms: https://qiime2.org/
3. Debulr [6]—A denoising algorithm for amplicon reads. Can be run via the qiime2 plugin or via the standalone script available from: https://github.com/biocore/deblur
4. DADA2 [7]—A denoising algorithm for amplicon reads. Can be run via the qiime2 plugin or via the standalone script available from: https://benjjneb.github.io/dada2/
5. Denoise [8]—Automatic removal of contaminant ASVs (based on blank samples). Available from: https://github.com/benjjneb/decontam
6. Calour [9]—Interactive heatmap visualization of amplicon experiments: https://github.com/biocore/calour

   A full point and click GUI version (EZCalour) is also available: https://github.com/amnona/EZCalour
7. SEPP [10]—Incorporating ASVs into a phylogenetic tree. Available as a qiime2 plugin: https://github.com/qiime2/q2-fragment-insertion

3 Methods

3.1 Experimental Processing

3.1.1 Experiment Design

Every microbiome study starts from a set of biological samples (Fig. 1a), from which the microbial population is detected (i.e., fecal samples from sick and healthy individuals in our toy scenario). The total number of samples required for detection of the biological signal (i.e., bacteria different between H and S) depends on the underlying biology and can range from a few samples in each group to thousands of samples. As a rule of thumb, a few tens of samples per group are usually required when examining a strong microbiome-associated signal in a cross-sectional study of human population. This large number stems from the large interindividual variability of microbial species [11] that in turn leads to a sparse microbial abundance matrix, combined with typical complex (i.e., zero-inflated) microbial distributions that necessitate nonparametric tests.

These samples are then processed and sequenced on one or more sequencing runs. The number of samples to combine in a single sequencing run depends on the desired per-sample reads coverage, and presents a balance between cost (i.e., more samples per run make the experiment cheaper) and resolution (less samples per run provides more reads per sample, thus enabling detection of lower frequency bacteria). While in theory normalizing each sample biomass after PCR (described in Subheading 3.1.5) should result in more or less equal number of reads per sample, this is typically not the case, and the number of reads per sample can vary over ten-fold between samples in the same run. Based on our experience, for a single Illumina MiSeq run (10–20 million reads), 200–400 samples result in most samples having over 10,000 reads. Also note that while a larger number of reads per sample is always desirable (to identify more of the rare bacteria), at some point these additional reads will contribute to detection of very low frequency bacteria, who's biological relevance is less important (i.e., how important biologically is a bacteria present in a frequency of 0.00001% in a sample), as well as more contaminants being identified (thus requiring special care during analysis before deciding if a low frequency bacteria is actually in the sample). Typically, 10,000 reads/sample (for medium to high biomass samples) is a good compromise between cost and rare-bacteria identification (enabling reliable detection of bacteria with frequency of at least 0.01% in the sample).

When collecting and processing samples from different groups in the experiment (i.e., S and H samples), it is important to randomize the layout of these samples during the experiment. This means not processing all H samples in one batch and the S samples in another batch, but rather mixing samples from both groups in

each processing batch. This is in order to prevent batch effects from appearing as true biological signals during the downstream analysis (*see* for example [12]).

If samples cannot be processed immediately, bacterial growth within the sample may lead to changes in the bacterial composition after collection. Storage at −80 °C is usually considered a "gold standard" treatment for preventing such growth. However, sometimes it is not possible to immediately freeze at −80 °C each sample, thus requiring alternative sample preservation techniques. This subject has been extensively studied in human fecal samples [13] as well as other animal feces [14], with various stabilization buffers as well as sample drying showing acceptable results for fecal storage at room temperature for a few days. Importantly, each preservation method leads to different biases in the microbiome, and therefore similar preservation of all samples in the experiment is required. In human fecal samples from the American Gut Project [15], such preservation is not possible due to mail and cost limitations. As an alternative approach, a small subset of bacteria has been shown to be most affected by storage conditions. By bioinformatically removing and ignoring reads from this subset (and assuming random distribution of storage times between different groups), comparison of such samples is possible [16].

3.1.2 Sample Processing

Besides the bacterial population we are interest in, our samples may contain additional nonbacterial DNA (such as human host DNA in saliva or biopsy samples, or fungal and plant DNA in leaf samples). Since the workflow contains a PCR step specific for bacterial 16S gene fragment, removal of the nonbacterial DNA is usually not required. However, in cases of very low bacterial biomass (and high nonbacterial DNA mass, such as biopsies), nonspecific amplification of nonbacterial DNA may occur and lead to a large number of reads being "wasted." This may be overcome by either specific nonbacterial DNA depletion or increasing the number of reads per sample. However, since DNA depletion kits may contain bacterial contaminant DNA, increasing the number of reads per sample is preferred.

During the experimental process of bacterial DNA extraction and amplification, contamination of the samples with nonsample originating bacterial DNA may occur [17]. Sources for this contaminating DNA include extraction and PCR reagents, people performing the experiment, lab air, and so on. The effect of this contamination on the samples depends on the samples' bacterial biomass. For example, for fecal samples (approx. 3×10^{11} bacteria/g [18]), the effect of these contaminants will usually be negligible, whereas for skin samples (approx. 10^3 to 10^4 bacteria/cm^2), the level of contaminants will constitute a nonnegligible part of the resulting reads. As a general rule of thumb, when going below 10^5 bacteria/sample well, special care needs to be taken to reduce

contaminant effects [19]. A detailed description of contamination sources and bioinformatic approaches for dealing with them is provided in Subheading 3.3.1.

Besides nonsample originating bacteria, another source for sample contamination is intersample leakage of bacterial DNA [20]. Since samples are handled together in parts of the experimental workflow, spillage or aerosols between samples can occur, leading to the leakage of bacterial DNA from one sample to another. While for similar-biomass samples this effect is small (<1%), a much more pronounced effect can happen when DNA from a high-biomass sample leaks to a low-biomass sample. Therefore, it is recommended not to handle samples with different biomass together (i.e., do not put skin and fecal samples in wells of the same plate for extraction or amplification). Note that while some contaminations will inevitably happen, and can be corrected bioinformatically (as discussed in Subheading 3.3.1), reducing the amount of contamination in the experimental procedure is always preferred.

3.1.3 DNA Extraction

It has been shown that the DNA extraction step can have a large impact on the subset of bacteria identified from the sample [21]. It is important in this context to remember that amplicon sequencing is not presumed to capture the complete bacterial composition of the sample or the exact frequency distribution. Both the extraction and the PCR are not able to obtain the DNA of all the bacteria in the sample. Therefore, in order to compare different samples, they need to all undergo the same extraction (and amplification) protocol. In 96-well automated workflow, extraction has been shown to be the major contributor to intersample contamination [20] and therefore different biomass samples (i.e., feces and saliva) should only be extracted in different plates.

3.1.4 PCR Amplification

The barcoded primers used for the amplification represent a compromise between different criteria. These include universality (i.e., the fraction of bacterial sequences in the sample amplified by the primers), resolution (i.e., the ability to detect differences between closely related bacteria on the amplified region) and amplified fragment length (the ability to join the forward and reverse reads), as well as primers used in prior studies in the field (in order to enable ASV level comparison/meta-analysis). Common primers include V12, V34, and V4 (*see* for example [22]). Note that the specific primer choice for a given experiment depends may depend on the bacteria of interest in the sample. For example, V12 is commonly used for oral samples as it has higher resolution on the streptococcus group, which is dominant in oral samples, whereas V4 is used in the EMP protocol, as it is more universal for diverse sample types.

The number of cycles used for the amplification (typically 25–35 cycles) can also affect the resulting bacterial frequencies, and needs to be kept constant between samples that will be compared to each other. While a lower number of cycles introduces less biases in the frequencies (compared to the actual frequencies in the sample) and introduces a lower number of chimeric sequences (bioinformatic removal of chimeric sequences is discussed in Subheading 3.2.2), a higher number of cycles can provide higher amplification and therefore may be needed for low biomass samples.

Until recently, the common practice for PCR amplification has been to perform the PCR reaction for each sample in triplicates (i.e., split each sample to three separate tubes, amplify each tube with the same primer, and then mix the three tubes together). However, a recent paper [23] has shown no advantage for using triplicates over using a single PCR reaction per sample, thus reducing both the reagents cost and the amount of labor required.

3.1.5 Sequencing

Prior to sequencing, PCR products from all samples (for the specific run) are pooled together. Ideally, we would like to obtain a constant number of reads per sample, and therefore a constant amount of DNA from each sample should be mixed (following a per-sample DNA concentration measurement). Note however that even under these conditions, a tenfold variation in the number of reads per sample often occurs.

Since all amplified sequences span the same region and start in a conserved region, a low per-position entropy (i.e., only one of the nucelotides is present in each position) is observed at some positions. This effect is the most pronounced on the first nucleotides, as insertions/deletions in some bacteria later in the region increase the entropy following them. Since Illumina sequencing requires all bases to appear in sufficient frequencies in each position, a "random" DNA template is combined with the pooled amplified samples prior to sequencing. Typically, randomly cut PhiX DNA is used as this template. While in theory this template DNA should not contain barcodes, in reality a small fraction of resulting reads contain part of the PhiX DNA attached to some sample barcodes. Therefore, usually a special step in the bioinformatic processing of the reads removes all PhiX reads from each sample by comparing each read to the known PhiX sequence (*see* Subheading 3.2.2).

3.2 Bioinformatic Analysis

The output of the sequencing run contains (in the case of paired-end sequencing) two fastq read files (forward and reverse reads), with a third fastq file containing the per-read barcode. In the demultiplexing step, each of the forward and reverse fastq files are assigned to a set of per-sample fastq files, using the barcode associated with the read. This is typically done using Golay error-

3.2.1 Demultiplexing

correction [24]. With a typical Illumina per-nucleotide error rate of 0.5% [25], and using 12 bp barcodes, $1 - (1 - 0.005)^{12} = 0.06$ of the barcodes are expected to contain at least one error. Using the Golay error correction with a maximal distance of 1 enables retaining reads associated with 1 mismatch, thus increasing the number of reads by approx. 6% (since most of the error-containing barcode reads will contain only 1 error). Note that increasing the maximal distance for the error correction (or alternatively using shorter barcodes) may lead to sample mislabeling for some reads with error containing barcodes and therefore not recommended.

3.2.2 Denoising

Ideally, after demultiplexing the reads, each per-sample fastq file should contain the amplified region sequences of all the bacteria present in the sample. However, both PCR and sequencing may introduce various errors. Therefore, each true bacterial sequence will be inflated to a "cloud" of close bacterial sequences (*see* Fig. 2a, b). Using a typical Illumina error rate of 0.5% per nucleotide, and a read length of 150 bases, we would expect only $(1 - 0.005)^{150} = 0.5$ of the resulting reads to be error free. The other half of the reads will contain various mismatches at different positions (compared to the bacterial sequence they originated from). While in single-organism sequencing this is typically not a problem (since the majority of reads in each position can be used as the predicted true value), a similar approach cannot be used in microbiome sequencing, since multiple bacterial sequences can be present in the same sample. Initial approaches to overcome this problem included closed, de novo, and open-reference OTU picking methods. In recent years, these methods have been superseded by denoising methods such as DADA2 [7], UNoise2 [26], and Deblur [6], that overcome several limitations of the OTU picking methods. In this section we will introduce the general idea of denoising methods, with specific details for the deblur denoising method.

The core concept of denoising methods is that PCR and sequencing errors behave in a statistically predictable manner. By using the large number of reads obtained from an amplicon sequencing experiment, these errors can be estimated and removed. While each denoising algorithm uses a different statistical model for estimating the PCR and read errors, they all share the same core concept of removal (or fixing) of error containing reads, and hence obtaining exact error free reads as the output.

1. PCR and sequencing error profile: When performing the amplicon experiment, not all resulting reads reflect actual sequences present in the original sample. These sequences are termed "error" sequences, with the two main sources being the PCR and the sequencing. The resulting errors can include missense errors (i.e., replacement of one nucleotide by

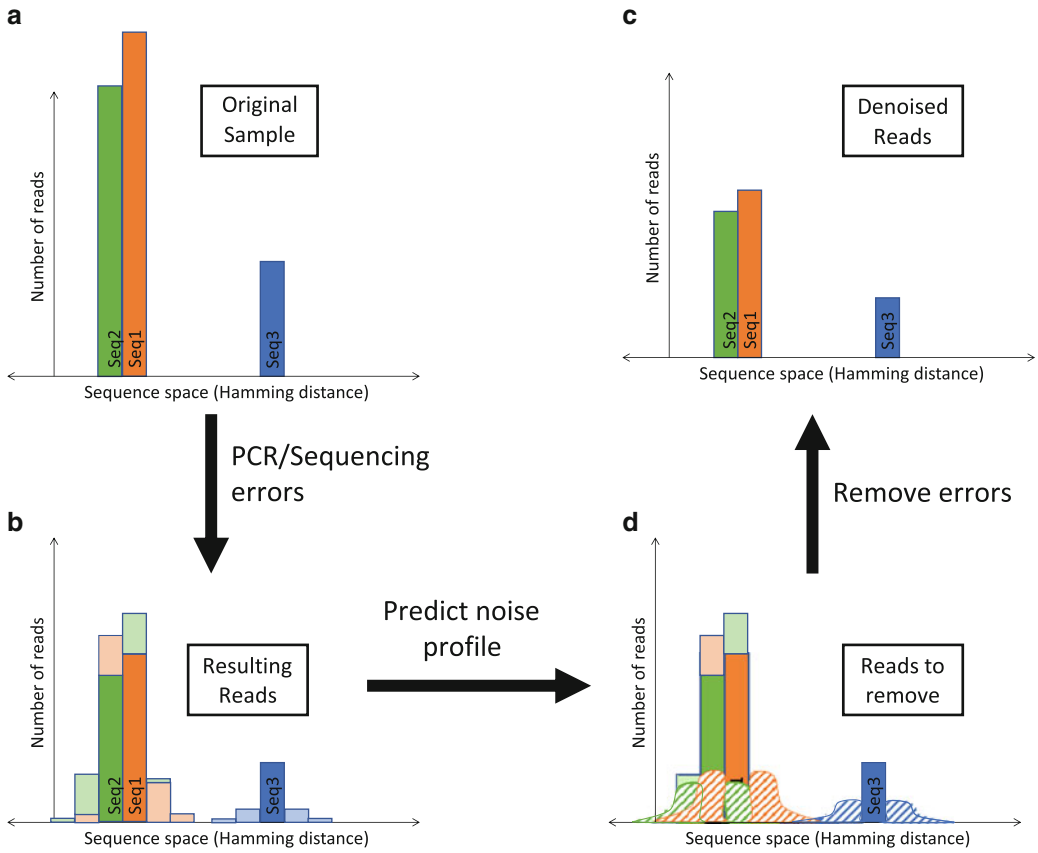


Fig. 2 Deblur algorithm based denoising. (**a**) Sample containing three bacterial sequences (two of which differ by one nucleotide). (**b**) Following PCR and sequencing, each sequence leaks to neighboring sequences due to PCR/sequencing errors (light colors). (**c**) Denoising algorithms predict the number of errors arising from each sequence in the result (dashed area of corresponding color) and subtract it. (**d**) The resulting denoised reads contain only reads from the original sample bacteria

another), insertions/deletions, as well as chimeric sequences (joining of two or more sequences due to polymerase dropping from one PCR amplicon and attaching to another). We will use the term "locus" for a given position in a given read. The error probability for such a locus depends on multiple factors, including the following:

(a) Read quality score (phred score): These scores are obtained from the sequencing machine, and reflect the sequencing quality of the locus, estimating the mean missense error probability. Note that this can only estimate the sequencing error, since for an error occurring during PCR, even perfect sequencing will keep the error.

(b) Local sequence context: The nucleotides before/after the locus can affect the probability of obtaining an error at the

locus. For example, homopolymer repeats can increase the indel error probability.

(c) Position within the amplicon: Illumina sequencing shows an increase in missense error probability as a function of the position (cycle) within the amplicon, and a sharp increase in error probability for the reverse reads.

(d) Number of PCR cycles: Higher number of PCR cycles can increase both chimeric sequence and missense/indel probabilities.

(e) PCR kit used: Fidelity level of the polymerase enzyme used can affect the PCR related read errors.

(f) Sequencing run parameters (fraction of spiked PhiX, specific sequencing machine used etc.)

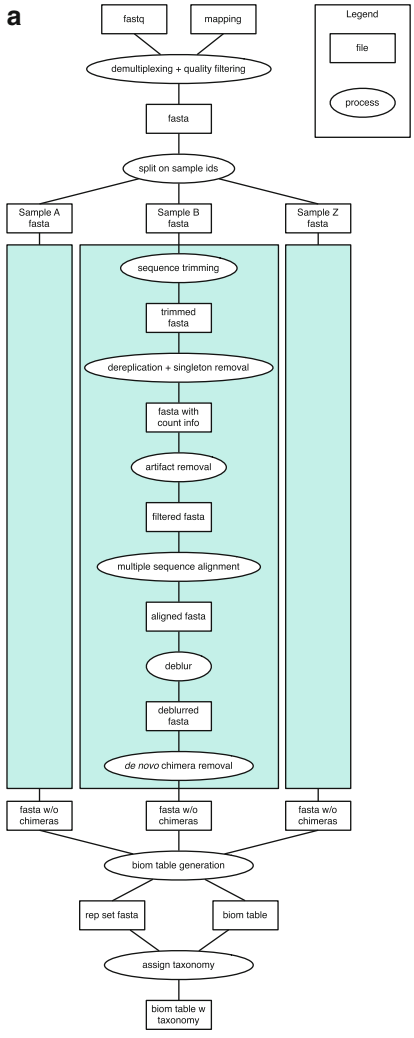
Since the exact error distribution is unknown, denoising algorithms use a partial error model incorporating some of the error sources. For example, Deblur uses an upper bound on the error probability incorporating dependence between errors in the same read, whereas DADA2 uses the per locus phred quality score.

2. Full deblur pipeline implementation: Given results of an amplicon experiment sequencing run, the algorithm treats each sample (i.e., all reads with a specific barcode) independently. The input to deblur is a collection of per-sample fasta (or fastq) files. The steps in the deblur pipeline are shown in Fig. 3a, and include:

(a) Optional quality filtering: Since deblur does not use the per-nucleotide phred scores, it assumes a minimal quality for the entire run. While in typical run conditions the deblur error model suffices, in extreme cases bad sequencing performance (as reflected in the phred scores) can cause an in increase in bad sequences. Therefore, an initial filtering of all reads based on a minimal phred quality value of 20 is recommended prior to running deblur.

(b) Trimming: Choosing the read length presents a balance between higher phylogenetic resolution (longer reads), and lower error rate (shorter reads, since illumine read error rate increases along the read). A typically used compromise is 150–200 bp. The current implementation of deblur does not support position-dependent read error profiles, and therefore using deblur also on reverse reads requires manual specification of the error profile (see below). Additionally, when processing forward (F) + reverse (R) reads, it is recommended to concatenate the F and R reads to a single long read, deblur, and then split and optionally join the F and R reads.

a



b

```
function deblur(L,p):
# L=(s,n)
# the list of (sequence, number of reads)
in the sample
#
# p(d) is the upper bound on the error
reads for hamming
# distance d
# sort by number of reads (highest first)
L←sort(L)
for (s,n) in L:
        for (r,m) in L:

        d=hamming_distance(s,r)
                m←m-p(d)*n
                if m<=0:

        remove (r,m) from L
return L
```

Fig. 3 Deblur pipeline. (**a**) The full deblur pipeline implementation. Most processing is done in parallel on all samples. (**b**) The main deblur algorithm pseudocode

(c) Dereplication and removal of singletons: The deblur algorithm counts the number of times each unique read appears in the sample. Singleton reads (appearing only once) are then removed from further downstream processing. This is done to prevent the effect of discreteness of the reads on the underlying statistical model.

(d) Removal of PhiX sequences: Some of the added PhiX DNA sequences obtain a barcode read, and are therefore observed in the sample reads. Using the known PhiX sequence, homologous sequences are discarded from the sample reads.

(e) Multiple sequence alignment: To facilitate fast pairwise alignment of reads while enabling insertion/deletion detection, all reads within the sample are aligned, thus enabling hamming distance calculation by counting mismatches.

(f) Main Deblur algorithm (Fig. 3b): Reads are sorted according to their frequency and then iterated (starting with the highest frequency read). For each read, all the close reads (i.e., hamming distance <9 and indel reads) are reduced in frequency according to the specified error profile. Reads with frequency < 0 are removed. Once all sequences are iterated, the remaining sequences and their frequencies represent the denoised sample.

(g) De novo chimera removal: Bimeric chimera sequences (i.e., sequences made from concatenation of two bacterial sequences) are identified using a de novo algorithm (meaning bacterial sequences are obtained from the sample itself rather than from an external database), requiring 100% identity to other common (deblurred) sequences (in the same sample) for both regions of the suspected Chimera read.

(h) Optional removal of nonbacterial sequences: Deblur outputs three biom tables: the first (all.biom) contains all deblurred reads. The second table (reference-hit.biom) contains only 16S reads. The latter is created by removing all deblurred sequences not homologous to bacterial 16S sequences (using a threshold of 65% homology to greengenes 13.8 88% representative set). The removed (nonbacterial) sequences are also output as the third table (reference-non-hit.biom).

(i) Joining of all samples, and optional removal of low abundance sequences: The out output biom tables are created by joining the deblurred reads from all samples. As an optional cleaning step (which is performed by default), sequences with <10 reads total over all samples are removed. Note that since deblur removes all error reads (rather than correcting them), the total number of reads in the output table is smaller (approximately 0.5 of the original reads for 150 bp reads with typical error rate).

3. Effect of over/underestimating the error probabilities: Since the read error behavior is very complex, deblur uses a partial error model for the error probabilities. Choosing this model represents a compromise between two competing options:

(a) A high model error rate will lead to removal of all erroneous sequences, but can also lead to removal of real

bacterial sequences in the sample. This can happen in cases where two close bacteria are present in the sample (say hamming distance 1 on the sequenced region), with one bacteria (A) present in high frequency and the other (B) at low frequency. If the frequency of bacteria B is lower than the model error rate * frequency of bacteria A, bacteria B reads will be identified as error reads and will be removed. Therefore, choosing a high model error rate increases the cases where such incorrect removal of bacteria happens.

(b) A low model error rate can lead to incomplete removal of read errors, therefore inflating the number of (incorrect) observed bacteria in the sample.

Since the effect of high model error rate is more subtle (resulting in removal of correct bacteria only if the sample contains closely related sequences at much higher frequency), the default deblur error model is a relatively "upper bound" on the error probability thus favoring removal of most errors. Specifying the error model for deblur can be performed using the "deblur workflow" command parameter: "--error-dist" Where the numbers specify the upper bound on probability for a read to contain 0–9 read errors. The mean error rate of the run is specified by the "--mean-error" parameter, and is used to calculate the original number of reads of a given sequence based on the observed number of reads and the read length.

3.3 Select Topics in Downstream Processing

3.3.1 Controlling for Contamination

1. Contamination sources: Following the denoising algorithm, we get a table containing the amount of reads for each ASV in each sample. Ideally this should reflect the relative abundance of each bacteria in the sample. However, besides the actual bacteria in the sample, results often include various contaminants arising from the reagents used, the experimental process and well-to-well contamination.

   We will briefly describe each source:

   (a) Reagent contamination: Amplicon sequencing requires the presence of bacterial DNA rather than intact bacteria. Therefore, even sterilized (i.e., autoclaved) reagents might contain bacterial DNA fragments, which are then amplified and sequenced. These contaminants may vary between kits or even batches of the same kit. The effect of these contaminants increases when processing low biomass samples.

   (b) Experimental procedure: the processing of samples will inevitably introduce some lab originating bacterial DNA into the samples. This bacterial DNA may include dust/air bacteria as well as human skin associated bacteria. These contaminants may vary between days and similarly

to reagent contaminants, are more pronounced in low biomass samples.

(c) Well to well: As described above, aerosols and sprays between samples being processed can lead to leakage of DNA from one sample to another. This effect is most pronounced when low and high biomass samples are handled together. Similar leakage can also occur in the stocks of barcoded primers used in the experiments, leading to similar sample-sample leakage (in a sample biomass-independent manner).

2. Effects of contaminants: Contaminants may cause multiple problems when analyzing amplicon sequencing results. These include:
   (a) increased alpha diversity (of bacteria not present in the real samples—an extreme example being when no bacteria are present in the sample and all the diversity arises from contaminants).
   (b) Enrichment of contaminant ASVs in samples with lower biomass. For example, Crohn's disease fecal samples are usually lower biomass compared to healthy controls, and therefore contaminants may present a higher relative abundance in these samples.
   (c) Identification of bacteria that are high in part of the samples as present in additional samples (due to well to well contamination).
   (d) Enrichment of contaminants in one group of samples vs. the other, due to difference in processing of the samples in the two groups (reagent batches, different days, etc.)
   (e) Masking of true biological signals in low biomass samples.
3. Bioinformatic control of contaminants: The best way to reduce the effect of contaminants is to prevent them during the experimental process. Some experimental practices should therefore be followed (as discussed in Subheading 3.1.2), including the randomization of samples from different groups prior to processing and separate processing of low and high biomass samples. Additionally, in order to enable bioinformatic mitigation of the contaminants, multiple negative and positive controls are required. Negative controls should include multiple blank samples exposed to the same environment and undergoing the same processing as the real samples. Ideally, positive controls should be of similar biomass to the real samples, and composed of one or more known bacteria, preferably not present in the real samples (to enable detection of well-to-well contamination), and again should undergo the same processing as real samples.

Due to well-to-well contamination, removal of all ASVs present in the blanks is not a good practice, as it may remove bacteria present in high frequency in the real samples (from where they leak to the blanks). Instead, contaminants can be identified by being in higher frequency in multiple blanks compared to the real samples. This can be done manually by examining the heatmaps of all ASVs (using tools such as Calour [9]), or automatically using programs such as decontam [8]. Another common feature of contaminant ASVs is highly correlated frequencies of reagent contaminants over the samples (due to similar dependence on the total sample biomass), as well as batch dependence, both which can be identified when examining the heatmap of ASVs in the samples. Additionally, prior biological knowledge about the ASVs can be used to mark suspicious ASVs (i.e., water bacteria appearing in skin samples, etc.). An interactive heatmap integrating prior information about ASVs is also available in the Calour analysis program [9].

3.3.2 Phylogenetic Tree Generation

Phylogenetic trees are used for both alpha diversity (i.e., Faiths phylogenetic diversity) and beta diversity (i.e., weighted/unweighted Unifrac) analysis. While for closed-reference OTUs, a pregenerated tree encompassing all representative set sequences was used, the introduction of ASVs requires generation of a new phylogenetic tree for each experiment, covering the specific ASVs observed in the experiment (as each experiment may contain different ASVs). De novo tree generation based on the ASVs from the experiment suffers from the limited scope (i.e., not all bacteria are represented) and short read length of the ASVs. As an alternative approach, SEPP [10] uses a tree insertion algorithm to place the observed ASVs into a preexisting scaffold tree (built based on a large full-length 16S dataset), resulting in more accurate phylogenetic trees for downstream analysis.

3.3.3 Rarefaction

The number of reads obtained from each sample in an amplicon experiment is not identical for all samples in the run. Prior to any downstream bioinformatic analysis, a common practice in many amplicon experiment bioinformatic is rarefaction—i.e., random subsampling of reads from all samples to obtain the same number of reads for all samples. This is done by selecting a threshold T for the number of reads per sample, dropping samples with number of reads $< T$, and randomly sampling (without repeats) T reads from each sample with $\geq T$ reads. Rarefaction is performed in order to prevent the effect of the discreteness of the data (i.e., reads are discrete events, leading to noncontinuous frequency counts). For example, if we look at the same sample twice, once with 100 reads and once with 10,000 reads, a bacteria present in frequency of 1/1000 will likely be observed in the 10,000 reads, but not in the 100 reads. In the case where the number of reads depends on

the sample type, this can lead to biases and incorrect rejection of the null hypothesis.

Rarefaction comes with a statistical cost—randomly dropping reads (and hence losing information). In some cases, where statistical tests that take into account the discrete nature of reads are used, it is possible to skip the rarefaction step (i.e., Deseq2 [27]). However, in other cases (mostly involving metrics and statistics based on presence/absence), rarefaction is necessary in order to prevent false positives. While it has also been argued that rarefaction itself can introduce false positives [28], this claim has been later refuted [29]. Therefore, as a general rule, we recommend rarefaction as a default unless it is known the specific test used is insensitive to the read depth.

3.4 Full Bioinformatic Processing Pipeline Using Qiime2

This pipeline follows the Deblur pipeline described in the excellent qiime2 "moving pictures" tutorial: https://docs.qiime2.org/2019.10/tutorials/moving-pictures/

The pipeline starts with the output of a deep-sequencing experiment, We assume we have the forward reads file (named sequences.fastq.gz) and the barcode reads (named barcodes.fastq.gz) in a single directory named READS_DIR. Additionally, we have a tab-separated mapping file associating each sample (first column, titled "sampleid") with the barcode used (second column, titled "BarcodeSequence") and any additional per-sample metadata needed.

3.4.1 Import the Deep-Sequencing Output Reads

```
qiime tools import \
--type EMPSingleEndSequences \
--input-path READS_DIR \
--output-path RUN1.qza
```

* This is done separately for each sequencing run. Other import options are available for other sequencing output formats. See the qiime2 importing data tutorial: https://docs.qiime2.org/2019.10/tutorials/importing/

3.4.2 Demultiplexing

```
qiime demux emp-single \
--i-seqs RUN1.qza \
--m-barcodes-file MAP.txt \
--p-rev-comp-barcodes \
--m-barcodes-column BarcodeSequence \
--o-per-sample-sequences DEMUX1.qza
```

3.4.3 Quality Filtering

```
qiime quality-filter q-score \
--i-demux DEMUX1.qza \
--o-filtered-sequences FILTERED1.qza \
--o-filter-stats filter-stats.qza \
```

```
--p-min-quality 19 \
--p-quality-window 3 \
--p-min-length-fraction 0.10
```

* This uses a relatively stringent quality criterion on minimal quality 19. This will result in a higher loss of reads but a cleaner output. To obtain more reads, "--p-min-quality 3" can be used instead (but may result in a slightly higher rate of error containing sequences).

3.4.4 Denoising Using deblur

```
qiime deblur denoise-16S \
--i-demultiplexed-seqs FILTERED1.qza \
--p-trim-length 150 \
--o-representative-sequences REPSEQS1.qza \
--o-table TABLE1.qza \
--p-sample-stats \
--o-stats deblur-stats.qza \
--p-no-hashed-feature-ids \
--p-jobs-to-start 4
```

* The trim length is usually the number of cycles in the sequencing run. Shorter trim lengths can be used if the sequencing run shows a strong drop in quality at some cycle.

* Since deblur is fully parallelizable, we get an n-fold increase in performance by using n threads. This is specified by the "--p-jobs-to-start" parameter.

* We use the "--p-no-hashed-feature-ids" flag to obtain the actual sequences embedded in the resulting table. Alternatively, if this flag is not specified, we will get a unique hash of each sequence instead of the actual sequence.

3.4.5 Creating a Phylogenetic Tree

1. Inserting ASVs into the tree.

```
qiime fragment-insertion sepp \
    --i-representative-sequences REPSEQS1.qza \
    --o-tree rooted-tree.qza \
    --o-placements insertion-placements.qza \
    --p-threads 4
```

* The phylogenetic tree is required for phylogenetic-aware metrics (such as unifrac). We recommend inserting the ASVs into an existing tree using SEPP (REF XX). While building a de novo tree using the ASVs is possible, it is less accurate, as it uses only a partial set of bacterial sequences (compared to a reference set), as well as shorter sequence lengths.

2. Removal of noninserted ASVs.

```
qiime fragment-insertion filter-features \
    --i-table TABLE1.qza \
```

```
--i-tree rooted-tree.qza \
--o-filtered-table TABLE1-FILTERED.qza \
--o-removed-table TABLE1-REMOVED.qza
```

* We need to remove the ASVs which could not be inserted into the bacterial tree. These may include host/other nonbacterial sequences. Phylogenetic-based analysis (i.e., unifrac etc.) should be performed on the filtered table.

3.5 Conclusions

Experimental and bioinformatic choices can have a large effect on the ability of the experiment to detect the true biological signal of interest. Since the experimental pipeline is not perfect, various artifacts may combine with the true microbial populations. It is therefore important to know the limitations of the methods used, in order to both minimize the effect of the artifacts, and to prevent incorrect conclusions. We described some common approaches for starting with a set of samples, and obtaining the microbial population in each of these samples. The limitations of the resulting ASV biom table (contaminants, variation in the number of reads per sample, incomplete extraction efficiency, etc.) should be taken into account when performing and interpreting downstream analysis results (alpha/beta diversity, differential abundance, correlation networks, etc.), as well as when designing novel analysis methods.

References

1. Lane DJ, Pace B, Olsen GJ et al (1985) Rapid determination of 16S ribosomal RNA sequences for phylogenetic analyses. Proc Natl Acad Sci U S A 82:6955–6959
2. Callahan BJ, Mcmurdie PJ, Holmes SP (2017) Exact sequence variants should replace operational taxonomic units in marker-gene data analysis. ISME J 11:2639–2643
3. Rideout JR, He Y, Navas-Molina JA et al (2014) Subsampled open-reference clustering creates consistent, comprehensive OTU definitions and scales to billions of sequences. PeerJ 2:e545
4. Thompson LR, Sanders JG, Mcdonald D et al (2017) A communal catalogue reveals Earth's multiscale microbial diversity. Nature 551 (7681):457–463
5. Bolyen E, Rideout JR, Dillon MR, Bokulich NA et al (2019) Reproducible , interactive , scalable and extensible microbiome data science using QIIME2. Nat Biotechnol 37:852–857
6. Amir A, McDonald D, Navas-Molina JA et al (2017) Deblur rapidly resolves single-nucleotide community sequence patterns. mSystems 2:1–7
7. Callahan BJ, McMurdie PJ, Rosen MJ et al (2016) DADA2: high-resolution sample inference from Illumina amplicon data. Nat Methods 13:581–583
8. Davis NM, Proctor DM, Holmes SP et al (2018) Simple statistical identification and removal of contaminant sequences in marker-gene and metagenomics data. Microbiome 6:1–14
9. Xu ZZ, Amir A, Sanders JG et al (2019) Calour: an interactive, microbe-centric analysis tool. mSystems 4:1–12
10. Janssen S, Mcdonald D, Gonzalez A et al (2018) Phylogenetic placement of exact amplicon sequences. mSystems 3:1–14
11. The Human Microbiome Project Consortium (2012) Structure, function and diversity of the healthy human microbiome. Nature 486:207–214
12. Salter SJ, Cox MJ, Turek EM et al (2014) Reagent and laboratory contamination can

critically impact sequence-based microbiome analyses. BMC Biol 12:1–12

13. Byrd DA, Chen J, Vogtmann E et al (2019) Reproducibility, stability, and accuracy of microbial profiles by fecal sample collection method in three distinct populations. PLoS One 14:1–19
14. Hale VL, Tan CL, Niu K et al (2016) Effects of fi eld conditions on fecal microbiota. J Microbiol Methods 130:180–188
15. Mcdonald D, Hyde E, Debelius JW et al (2018) American gut: an open platform for citizen science microbiome research. mSystems 3:1–28
16. Amir A, McDonald D, Navas-Molina JA et al (2017) Correcting for microbial blooms in fecal samples during room-temperature shipping amnon. mSystems 2:1–5
17. De Goffau MC, Lager S, Salter SJ et al (2018) Recognizing the reagent microbiome. Nat Microbiol 3(8):851–853
18. Milo R, Jorgensen P, Moran U et al (2010) BioNumbers—the database of key numbers in molecular and cell biology. Nucleic Acids Res 38:750–753
19. Minich JJ, Zhu Q, Janssen S et al (2018) KatharoSeq enables high-throughput microbiome analysis from low-biomass samples. mSystems 3:1–16
20. Minich JJ, Sanders JG, Amir A et al (2019) Quantifying and understanding well-to-well contamination in microbiome research. mSystems 4:1–13
21. Sinha R, Abu-ali G, Vogtmann E et al (2017) Assessment of variation in microbial community amplicon sequencing by the Microbiome Quality Control (MBQC) project consortium. Nat Publ Gr 35:1077–1086
22. Wear EK, Wilbanks EG, Nelson CE et al (2018) Primer selection impacts specific population abundances but not community dynamics in a monthly time-series 16S rRNA gene amplicon analysis of coastal marine bacterioplankton. Environ Microbiol 20 (8):2709–2726
23. Marotz C, Sharma A, Humphrey G et al (2019) Triplicate PCR reactions for 16S rRNA gene amplicon sequencing are unnecessary. BioTechniques 67:29–32
24. Hamady M, Walker JJ, Harris JK et al (2008) Error-correcting barcoded primers for pyrosequencing hundreds of samples in multiplex. Nat Methods 5:235–237
25. Loman NJ, Misra RV, Dallman TJ et al (2012) Performance comparison of benchtop high-throughput sequencing platforms. Nat Biotechnol 30:434–439
26. Edgar RC (2016) UNOISE2: improved error-correction for Illumina 16S and ITS amplicon sequencing. BioRxiv
27. Love MI, Huber W, Anders S (2014) Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2 1–21. Genome Biol 15(12):550
28. McMurdie PJ, Holmes S (2014) Waste not, want not: why rarefying microbiome data is inadmissible. PLoS Comput Biol 10:e1003531
29. Weiss S, Xu ZZ, Peddada S et al (2017) Normalization and microbial differential abundance strategies depend upon data characteristics. Microbiome 5:1–18

Chapter 8

RNA-Seq in Nonmodel Organisms

Vered Chalifa-Caspi

Abstract

RNA-Seq is nowadays an indispensable approach for comparative transcriptome profiling in model and nonmodel organisms. Analyzing RNA-Seq data from nonmodel organisms poses unique challenges, due to unavailability of a high-quality genome reference and to relative sparsity of tools for downstream functional analyses. In this chapter, we provide an overview of the analysis steps in RNA-Seq projects of nonmodel organisms, while elaborating on aspects that are unique to this analysis. These will include (1) strategic decisions that have to be made in advance, regarding sequencing technology and reference to use; (2) how to search for available draft genomes, and, if necessary, how to improve their gene prediction and annotation; (3) how to clean raw reads before de novo assembly; (4) how to separate the reads in RNA-Seq projects of symbiont organisms; (5) how to design and carry out a de novo transcriptome assembly that will be comprehensive and reliable; (6) how to assess transcriptome quality; (7) when and how to reduce redundancy in the transcriptome; (8) techniques and considerations in transcriptome functional annotation; (9) quantitating transcript abundance in the face of high transcriptome redundancy; and, most importantly, (10) how to achieve functional enrichment testing using available tools which either support a large range of species or enable a universal, non–species-specific analysis.

Throughout the chapter, we will refer to a variety of useful software tools. For the initial analysis steps involving high-volume data, these will include Linux-based programs. For the later steps, we will describe both Linux and R packages for advanced users, as well as many user-friendly tools for nonprogrammers. Finally, we will present a full workflow for RNA-Seq analysis of nonmodel organisms using the NeatSeq-Flow platform, which can be used locally through a user-friendly interface.

Key words RNA-Seq, Nonmodel organisms, Transcriptome, De novo assembly, Annotation, Expression profiling, Differential expression, Functional enrichment, Pathway analysis

1 Introduction

Human and model organisms, such as mouse, rat, and Arabidopsis, typically have a high-quality genome sequence which can serve as a reference for RNA-Seq data analysis, and a rich assortment of tools for downstream functional analyses. On the other hand, nonmodel organisms may have a low-quality (draft) genome reference or none at all, and a limited number of tools for function and pathway analyses. For microbial organisms with a small genome, genome sequencing is cheap and may solve the problem; however, as the

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_8,

genome is larger and contains more repeat regions, achieving a genome reference becomes more expensive, and it is often preferred to use a reference transcriptome assembled from the RNA-Seq reads themselves [1–3].

In this chapter we will discuss the considerations in choosing a sequencing technology and the reference for read mapping, and then describe the data analysis steps from quality assessment and preprocessing of the reads to transcriptome assembly and annotation, read alignment and quantitation, statistical testing for differential expression, clustering, and function enrichment analyses. In each of the data analysis steps, we will concentrate on the aspects that are peculiar to nonmodel organisms and suggest relevant methods and resources. Finally, we will present a complete RNA-Seq workflow that uses some of these methods.

2 Materials

The tools described throughout the chapter are listed in Table 1.

Table 1
Available tools for RNA-Seq analysis in nonmodel organisms

Tool/resource	Mentioned in Subheading (s)	Function	CL[a]	GUI[b]	References	Link
NCBI Genome	3.2	Database of sequenced genomes	+	+		https://www.ncbi.nlm.nih.gov/genome/
Ensembl Genomes	3.2	Database of sequenced genomes	+	+		http://ensemblgenomes.org/
JGI Genome Portal	3.2	Database of sequenced genomes	+	+		https://genome.jgi.doe.gov/portal/
Trimmomatic	3.3	Quality trimming of sequence reads	+		[12]	http://www.usadellab.org/cms/?page=trimmomatic
Trim Galore!	3.3	Read quality trimming of sequence reads	+			https://www.bioinformatics.babraham.ac.uk/projects/trim_galore/
FastQC	3.3	QA of raw reads	+	+		https://www.bioinformatics.babraham.ac.uk/projects/fastqc/

(continued)

Table 1
(continued)

Tool/resource	Mentioned in Subheading (s)	Function	CL[a]	GUI[b]	References	Link
MultiQC	3.3, 3.9	Summary QA for NGS experiments	+		[13]	https://multiqc.info/
BWA-MEM	3.3	Read alignment	+		[15]	https://github.com/lh3/bwa
Samtools	3.3	Utilities for post-processing of alignments files	+		[16]	http://www.htslib.org/
FastQ Screen	3.3	Screen reads for contamination	+			https://www.bioinformatics.babraham.ac.uk/projects/fastq_screen/
Kaiju	3.3	Taxonomic classification of reads	+	+	[17]	http://kaiju.binf.ku.dk/
BBSplit	3.4	Splits reads by organism of origin	+			https://jgi.doe.gov/data-and-tools/bbtools/bb-tools-user-guide/bbmap-guide/
Trinity	3.5, 3.9	Transcriptome de novo assembly	+		[18, 19]	https://github.com/trinityrnaseq/trinityrnaseq/wiki
RSEM	3.5, 3.9, 3.10	Transcript and gene quantification	+		[21]	https://github.com/deweylab/RSEM
DESeq2	3.6, 3.9, 3.10	Differential expression testing	+		[22]	http://bioconductor.org/packages/release/bioc/html/DESeq2.html
CD-HIT	3.6	Sequence clustering	+	+	[23]	http://weizhongli-lab.org/cd-hit/
RapClust	3.6	Transcriptome clustering	+		[24]	https://github.com/COMBINE-lab/RapClust
QUAST	3.7	Transcriptome QA	+		[25]	http://quast.sourceforge.net/
BUSCO	3.7	QA of transcriptome completeness	+		[26]	https://busco.ezlab.org/
Blast2GO	3.8, 3.11.2	Functional annotation and enrichment analysis		+	[27]	https://www.blast2go.com/
Trinotate	3.8, 3.11	Transcriptome annotation	+		[28]	https://github.com/Trinotate/Trinotate.github.io/wiki

(continued)

Table 1
(continued)

Tool/resource	Mentioned in Subheading (s)	Function	CL[a]	GUI[b]	References	Link
TransDecoder	3.8	Find coding regions within transcripts	+			https://github.com/TransDecoder/TransDecoder/wiki
edgeR	3.9	Differential expression testing	+		[29, 31]	https://bioconductor.org/packages/release/bioc/html/edgeR.html
SVA	3.9	Batch effect removal	+		[32]	https://bioconductor.org/packages/release/bioc/html/sva.html
DEBrowser	3.9, 3.10	Differential expression analysis and visualization		+	[33]	https://github.com/UMMS-Biocore/debrowser
MORPHEUS	3.10	Clustering and other analyses		+		https://software.broadinstitute.org/morpheus/
ClustVis	3.10	Clustering and PCA		+	[34]	https://biit.cs.ut.ee/clustvis/
Expander	3.10, 3.11.5	Gene expression analysis and visualization		+	[35]	http://acgt.cs.tau.ac.il/expander/
DAVID	3.11	Functional enrichment		+	[38]	https://david.ncifcrf.gov/
KEGG Pathway Database	3.11, 3.11.4	Pathway database		+	[39]	https://www.genome.jp/kegg/pathway.html
STRING	3.11	Protein–protein association networks, functional enrichment	+	+	[40]	https://string-db.org/
g:Profiler	3.11	Functional enrichment		+	[41]	https://biit.cs.ut.ee/gprofiler/gost
clusterProfiler	3.11, 3.11.3, 3.11.5	Functional enrichment	+		[42]	https://bioconductor.org/packages/release/bioc/html/clusterProfiler.html
AnnotationHub	3.11, 3.11.3	Annotation	+			https://bioconductor.org/packages/release/bioc/html/AnnotationHub.html
AgriGO	3.11	GO analysis for agriculture		+	[43]	http://bioinfo.cau.edu.cn/agriGO/

(continued)

Table 1
(continued)

Tool/resource	Mentioned in Subheading (s)	Function	CL[a]	GUI[b]	References	Link
UniProt Retrieve/ID mapping	3.11	Protein annotation	+	+		https://www.uniprot.org/uploadlists/
KAAS	3.11, 3.11.4	KEGG annotation		+	[44]	https://www.genome.jp/kegg/kaas/
BiNGO	3.11.1	GO enrichment		+	[45]	https://www.psb.ugent.be/cbd/papers/BiNGO/Home.html
biomaRt	3.11.3	Annotation	+		[48]	https://bioconductor.org/packages/release/bioc/html/biomaRt.html
goseq	3.11.5	Functional enrichment	+		[49]	https://bioconductor.org/packages/release/bioc/html/goseq.html
WebGestalt	3.11.5	Functional enrichment	+	+	[50]	http://www.webgestalt.org/
FunRich	3.11.5	Functional enrichment		+	[52]	http://www.funrich.org/
GSEA	3.11.5	Functional enrichment	+	+	[47]	https://www.gsea-msigdb.org/gsea/index.jsp
NeatSeq-Flow	3.12	Workflow management platform with ready to use, customizable pipelines	+	+		https://neatseq-flow.readthedocs.io/en/latest/
AUGUSTUS	*see* **Note 1**	Gene prediction	+	+	[53]	http://bioinf.uni-greifswald.de/augustus/
BRAKER	*see* **Note 1**	Gene prediction	+		[54]	http://bioinf.uni-greifswald.de/bioinf/braker/

[a] *CL* command line
[b] *GUI* graphical user interface

3 Methods

3.1 Strategic Decision: Which Sequencing Technology to Use

The decision on a sequencing technology involves consideration of cost, sequencing depth, sequencing accuracy, contiguity of assembly, and whether expression is assessed at gene or splicing isoform level. Illumina sequencing has been the dominant RNA-seq platform for many years. Its reads' error rates are extremely low;

affordable high sequencing depth allows for detection of lowly expressed transcripts; and established analysis methods are available. The main limitations of Illumina sequencing are the presence of artifactual chimeras and fragmented genes in the assembled transcriptome, and the inability to accurately detect and reconstruct splice variants. On the other hand, long-read sequencing by Oxford Nanopore Technologies (ONT) and Pacific BioSciences (PacBio) bring new promise in the ability to sequence full-length transcripts from end to end and thus identify complex splice isoforms and measure their expression. ONT and PacBio still suffer from high error rates and lower throughput compared to Illumina. Hybrid sequencing ("Hybrid-Seq") brings the best of both worlds, by integrating short- and long-read sequencing. Application of Hybrid-Seq to transcriptome analysis in the absence of a reference genome is described in [4, 5].

In the current chapter we will elaborate on the assembly and analysis of transcriptomes obtained with Illumina technology alone. Transcript profiling using long-read sequencing has been described by Bayega et al. in a previous volume of this series [6], and reviewed in recent articles [7–9].

3.2 Strategic Decision: Which Reference to Use

In RNA-Seq, we typically compare gene or transcript expression among tissues, conditions, points in time, and so on. To enable comparison, reads from all samples need to be aligned to a common reference. The decision of which reference to use is crucial to the analysis. If a genome sequence of the same species is available, this is usually the preferred option due to its completeness with regard to gene content compared to transcriptome assembly. For expression quantification, contiguity of the genome is not an important factor, as long as most or all genes are represented in it, therefore even a draft genome may suffice. Yet draft genomes of high eukaryotes should be regarded with cautious, as gene structure (location of transcription start sites, exons, and introns) may not always be predicted accurately, and the genome may suffer from artifactual or inaccurate duplications. It is possible, though, to improve gene prediction, as explained in **Note 1**. Using the genome of another species as a reference is only helpful if the species are very close [10, 11] (*see* **Note 2**). To check the availability and to download sequenced genomes, the user may search NCBI Genome (https://www.ncbi.nlm.nih.gov/genome/), Ensembl Genomes (http://ensemblgenomes.org/) or JGI Genome Portal (https://genome.jgi.doe.gov/portal/). In the absence of a reference genome, if sequencing the genome is too expensive, a common practice is to de novo assemble a reference transcriptome from the RNA-Seq reads. The higher the depth of sequencing and the larger the variety of tissues and conditions of sequence origin, the more comprehensive the assembly will be, and the higher the chances to assemble rare transcripts at their full length. Reads from previous RNA-Seq

projects of the same organism may also be included in the assembly, as long as they are from a very close genetic background, otherwise inaccurate chimeras may be produced (*see* also **Note 3**). Optimally, the reads should be as long as possible (at least 100 bp paired end reads). In large experiments with many samples, some people prefer to use very short reads, or even only 3′ or 5′ sequencing, for the expression quantification, and prepare a separate library with longer reads for the transcriptome assembly. In such cases, one has to make sure that the transcriptome assembly is built from pooled RNA from all conditions assessed in the experiment, otherwise the reference will be incomplete with regard to the RNA-Seq data at hand.

3.3 Quality Assessment of the Raw Sequence Reads

Raw sequence reads from all samples, for both expression profiling and assembly, need to undergo quality assessment (QA). In principle, reads destined for de novo assembly need more stringent cleaning, as the reference will be the basis for subsequent analyses. Popular tools for read trimming and filtering are Trimmomatic [12] and Trim Galore! (https://www.bioinformatics.babraham.ac.uk/projects/trim_galore/). Summary statistics and diagnostic plots of read quality, before and after trimming, can be obtained using FastQC (Barbaham Bioinformatics https://www.bioinformatics.babraham.ac.uk/projects/fastqc/) and MultiQC [13]. The reader is referred to [14] for further explanations on the QA process.

Depending on the library preparation protocol, it may be necessary to check for the proportion of rRNA sequences before transcriptome assembly, and remove them if necessary. To this end, reads are aligned to a database of rRNA sequences and unaligned reads are retained. If a collection of rRNA reference sequences is unavailable for the organism at hand, one may retrieve relevant sequences from the broader taxonomic category (e.g., crustaceans) by searching NCBI Entrez with a search term like "ribosomal rna[Title] OR rrna[Title]) AND "Crustacea"[Organism]". Alignment may be performed with BWA-MEM [15]. The proportion of mapped reads can be assessed using the Samtools [16] flagstat command. FASTQ files of unmapped read pairs may be extracted from the alignment BAM files using the Samtools fastq command with the *–f 12* parameter. The number argument of *–f* denotes a combination of read alignment properties encoded in the "flag" field of the SAM or BAM file. The number 12 means that both reads per pair were unmapped. For other combinations, see https://broadinstitute.github.io/picard/explain-flags.html. Read more about SAM format and the flag field in [14] Sect. 4.2.2 and Fig. 15.

For transcriptome assembly it is crucial that the vast majority of the reads are derived from the expected organism. It is possible to screen for contamination using FastQ Screen (https://www.bioinformatics.babraham.ac.uk/projects/fastq_screen/), where the user supplies a set of genomes or sequence collections against

which the reads will be searched. The program reports what proportion of the reads could be mapped, either uniquely or to more than one location, against each of the specified reference genomes. Alternatively, it is possible to assess the taxonomic composition of the reads using tools designed for shotgun metagenomics. For example, in Kaiju [17] the user can identify microbial taxa using one of the protein databases provided by Kaiju, or build a protein database that encompasses all archaea, bacteria, and eukaryote organisms from NCBI nr. The latter is achieved by including the NCBI taxonomy ID for "Cellular organisms," 131,567, in the kaiju-taxonlistEuk.tsv file and then executing the "kaiju-makedb" script with the *–s nr_euk* argument (*see* section "Creating the reference database and index" in https://github.com/bioinformatics-centre/kaiju). Kaiju output file shows the taxon assignment per read, whereas the kaiju2table program creates summary taxonomy reports for any given taxonomic level (species, genus, family, etc.). Once a contaminant organism is identified, reads originating from that organism can be filtered out using the same approach as described above for rRNA cleaning.

3.4 Reads Separation Between Organisms

Symbiont organisms pose special challenges for RNA-Seq experiments, as the sequencing data contain a mixture of reads originating from two or more distinct species. Successful separation of the reads to their species of origin, a process called binning, will benefit from the availability of reference genomes or transcriptomes of the original species. If these are not available, effort may be made to gather all available protein and nucleotide sequences of closely related species and use them as a reference. The raw reads can then be aligned to the references and separated accordingly. In a naïve approach, each read is tagged as mapped to species A, mapped to species B, mapped to both species (ambiguous) and unmapped. Care should be taken with reads that map to multiple locations in one of the species and thus regarded as unmapped in that species, although they do belong there. The BBSplit tool can be used to map the reads to multiple references simultaneously. It uses the BBMap aligner, and assigns each read to the reference to which it aligns best (https://jgi.doe.gov/data-and-tools/bbtools/bb-tools-user-guide/bbmap-guide/). The program also offers disambiguation options, such that reads that map to multiple references can be binned with all of them, none of them, one of them, or put in a special "ambiguous" file for each of them. Paired reads are always kept together. After separation, reads can be de novo assembled for each species separately.

3.5 Transcriptome De Novo Assembly

The most popular transcriptome assembly program for short reads is Trinity [18, 19], which unlike most other publicly available assemblers was developed specifically for transcriptome and not genome assembly [2], considering splicing isoforms. The reader is referred to comparative evaluations of Trinity vs. other assemblers

in [11, 20]. The Trinity package also contains companion scripts for downstream analyses of the assembly, including filtering, annotation, translation, read mapping, transcript abundance estimation, and statistical testing for differential expression. Commands and practical tips for running Trinity are presented in [1, 19]. Best practices for de novo transcriptome assembly with Trinity are presented here: https://informatics.fas.harvard.edu/best-practices-for-de-novo-transcriptome-assembly-with-trinity.html

Typically, Trinity produces a very large number of transcripts (hundreds of thousands are not rare), which are clustered to putative genes. The software does not produce a "consensus" sequence per gene, but this is usually not necessary, since RSEM [21] computes an abundance estimation per gene (in addition to that per transcript) based on the transcript counts. For gene annotation purposes, the user may use the Trinity script "filter_low_expr_transcripts.pl" with the *--highest_iso_only* parameter to choose the most highly expressed transcript per gene. The representative transcripts can then be searched against sequence databases for function prediction. The representative transcripts may also be used for transcriptome QA (*see* Subheading 3.7).

3.6 Transcriptome Filtering

Often, the number of genes predicted by Trinity is still much higher than expected. In principle, this would have influenced the FDR multiple testing adjustment in the differential expression analysis, and cause many genes to not pass the significance cutoff. Fortunately, DESeq2 [22] addresses this problem. The *results* function of DESeq2 performs independent filtering and it omits from multiple testing adjustment all genes with mean normalized counts below a filtering threshold. By default, it chooses a threshold that maximizes the number of genes found at a user-specified target FDR cutoff (e.g., 0.1). The adjusted p-values for the genes which do not pass the filter threshold are set to NA.

If the user still wishes to reduce the number of genes in the dataset, for example to save time on annotation or to facilitate viewing the full dataset in Excel, we recommend filtering the genes such that only genes which are "reliably" expressed in at least one of the treatment groups will be retained. For example, we may require that retained genes will have at least three TPM (normalized) counts in at least X (a certain portion) of the replicates in at least one of the treatment groups.

Another option, aimed at reducing transcriptome redundancy, is to cluster the transcriptome based on sequence similarity, for example with CD-HIT [23] or RapClust [24], and then choose a representative transcript per cluster. *See* [11], and code examples in https://github.com/trinityrnaseq/trinity_community_codebase/wiki/Trinity-best-transcript-set and https://github.com/trinityrnaseq/trinityrnaseq/wiki/There-are-too-many-transcripts!-What-do-I-do%3F .

3.7 Transcriptome QA

It is essential to assess the quality of the assembled transcriptome before continuing the analysis. Two popular tools for this purpose are QUAST [25] and BUSCO [26]. Both are also useful for comparing assemblies, for example after different filtering procedures. Whereas QUAST assesses inherent properties of the assembly, such as the number of contigs at various length cutoffs, N50, L50, and others, BUSCO assesses the completeness of the assembly in terms of gene content, through comparing the transcriptome to a lineage-specific dataset of universal single-copy genes, termed "BUSCOs." BUSCO reports on the number and proportion of complete and single-copy, complete and duplicated, fragmented, and missing BUSCOs. Comparing BUSCO results of the full transcriptome vs. the set of gene representatives (Subheading 3.5) will typically show a slightly higher proportion of complete BUSCOs (single copy + duplicated) in the full transcriptome, but a much higher ratio of complete single copy–complete duplicated in the gene representatives, as expected.

3.8 Transcriptome Functional Annotation

In transcriptome annotation we try to assign functional meaning to the transcripts. Annotation is typically done at the gene level; therefore, it is advisable to first create a dataset containing a representative transcript per gene, selected based on length or coverage (*see* Subheading 3.5). Annotation is then inferred through sequence similarity to a functionally annotated gene or protein from a database. A protein-based search is more sensitive, so one may first run a "translated" blastx search against a protein database, and then, optionally, an additional blastn search for identifying noncoding RNAs. There are many options for selecting a protein database, each with its own pros and cons. A generic database, such as NCBI nr or UniProt will provide a more comprehensive annotation, whereas searching against a single closely related annotated species may yield more relevant annotations. Using a less redundant and more curated protein database, such as NCBI's RefSeq or the Swiss-Prot section of UniProt, reduces computation time and offers more reliable annotations, at the expense of comprehensiveness (*see* **Note 4**).

A major limitation of sequence similarity-based annotation is that the best BLAST hit achieved in a database search is sometimes described as "hypothetical protein," "predicted protein," "uncharacterized protein," or the like, whereas subsequent significant hits do have informative title. The desktop application Blast2GO [27] offers a very useful tool, BLAST Description Annotator (BDA), which applies a language processing algorithm to extract the best possible protein description from the set of n top BLAST hits passing the specified e-value cutoff (default: up to 20 proteins with e-value <0.001). The basic, free version of Blast2GO runs BLAST against NCBI server through the Web, which makes it impractically slow. A faster BLAST search on Blast2GO cloud is

possible in the payed version only. However, it is possible to provide Blast2GO with BLAST results in XML format, after running BLAST locally with the option *–outfmt 14*. Additional features of Blast2GO are described in Subheading 3.11.2.

Trinotate [28], developed by the Trinity team, offers a comprehensive annotation suite designed for automatic functional annotation of transcriptomes. It makes use of a number of different methods for functional annotation including homology search to known sequence data (BLAST+ against Swiss-Prot), protein domain identification (HMMER against Pfam), protein signal peptide and transmembrane domain prediction (SignalIP and TMHMM), rRNA prediction (RNAmmer), and leveraging various annotation databases (eggNOG/GO/KEGG databases). *See* also **Note 5**.

TransDecoder (https://github.com/TransDecoder/TransDecoder/wiki), also from the Trinity group, identifies candidate coding regions within transcripts and translates them. Optionally, it can identify open reading frames with homology to known proteins via BLAST or Pfam searches. The proteome produced by TransDecoder may be used, for example, as a reference for MAS-Spec proteomics profiling of the same species under similar conditions as the RNA-Seq experiment.

The approaches described above may also be useful for improving or updating the annotation of draft reference genomes, as long as they provide FASTA files of predicted cDNA, coding regions (CDS), and/or protein sequences (*see* also **Note 1**).

3.9 Read Alignment and Quantitation, and Their QA

Alignment of the reads to the reference transcriptome and quantitation of gene abundance per sample can easily be performed using the Trinity script "align_and_estimate_abundance.pl", which runs external tools (e.g., Bowtie2 and RSEM). Alternatively, they can be done directly with RSEM [21], which automatically runs the Bowtie, Bowtie2, or STAR alignment programs and then estimates abundance.

Assembled transcriptomes are typically much more redundant than genomes. Therefore, many reads do not map uniquely to a single transcript but rather to multiple transcripts. RSEM algorithm was designed to specifically handle this problem. It estimates the number of reads (or read pairs) that are derived from each given transcript or gene, taking into account the multiple alignments of the reads. This results in noninteger values, which are then rounded to imitate raw counts necessary for differential expression testing methods such as edgeR [29–31] or DESeq2.

Summary plots of the alignment efficiency and the fractions of uniquely and ambiguously mapped reads per sample can be produced with MultiQC. More importantly, it is highly recommended to create a matrix of normalized counts per gene per sample, and use it for diagnostic plots such as principal component analysis

(PCA), correlation heatmap and hierarchical clustering, in order to assess global effects on the data and to identify exceptional samples. Technical biases (batch effects) may mask additional, milder effects in the data, therefore it is sometimes necessary to repeat the diagnostic plots after adjusting the count data for the batch effects.

A matrix of sample-wise normalized counts may be produced by the Trinity script "abundance_estimates_to_matrix.pl". By default, the script generates Trimmed Mean of M (TMM) values, using a normalization method originally implemented by the edgeR package [29, 30]. Alternatively, it is possible to use the vst or rlog functions in DESeq2 to generate normalized count matrix through variance stabilizing transformation or regularized log transformation, respectively. For QA purposes, it is advisable to make sure that the *blind* parameter of these DESeq2 functions is set to TRUE, in order for the normalization to be blind to the experimental design (i.e., to sample attribution to treatment groups). Batch correction may be achieved using the *ComBat* function of the SVA R package [32].

DESeq2 tutorial (http://bioconductor.org/packages/release/bioc/vignettes/DESeq2/inst/doc/DESeq2.html) demonstrates useful R functions for drawing diagnostic plots. Beyond that, the R graphic packages ggplot2 (https://ggplot2.tidyverse.org/) and Plotly (https://plot.ly/r/scientific-charts/) may be of great help. Nonprogrammers may benefit from DEBrowser [33], an interactive differential expression analysis and visualization tool for count data. The application provides a rich and interactive graphical user interface built on R's Shiny infrastructure. Starting from raw counts, it enables users to carry out an entire RNA-Seq analysis, while visualizing the data in each step with various types of graphs.

3.10 Differential Expression and Clustering Analysis

Statistical testing for identifying differentially expressed (DE) genes is typically performed with DESeq2 or edgeR R packages. Both are designed specifically for count data, and provide high flexibility in designing the right statistical model for the research question and data at hand. It is important to remember that the input to DE testing in both packages is a raw count data matrix (or RSEM abundance estimations), without prior normalization and without batch correction. Normalization is done internally and known batch effects should be included in the statistical model. DESeq2 and edgeR User Guides provide clear and detailed explanations on carrying out the analysis for various experimental designs.

The dplyr R library for data manipulation can provide great help in further handling of the results. For example, in extracting sets of genes passing predefined FDR adjusted *p*-value and fold change cutoffs per comparison; Intersecting and/or unifying the gene sets; preparing the data for clustering, plotting and functional analyses; adding annotations and normalized counts for display in Excel and so on.

Hierarchical clustering of DE gene sets may be carried out using, for example, pheatmap or eclust in R. Nonprogrammers may use the friendly web-based tools MORPHEUS (https://software.broadinstitute.org/morpheus/) or ClustVis [34] (https://biit.cs.ut.ee/clustvis/).

The entire analysis may be carried out through a graphical user interface using DEBrowser [33], mentioned above, or the Expander desktop application for transcriptome analysis [35]. Among other clustering methods, Expander also offers its unique CLICK algorithm [36], which partitions the genes to distinct clusters so as to maximize the homogeneity (similarity in expression) within each cluster and the separation (dissimilarity) between clusters. It does so without prior assumptions on the number of clusters or their structure.

3.11 Pathway/ Function Enrichment Analysis

Lists of DE genes from treatment comparisons as well as groups of co-expressed genes from clustering analysis (collectively termed "genes of interest") may be further analyzed to find enriched biological functions and pathways within them [37]. These analyses require databases which associate the organism genes with functional categories such as ontology terms, conserved domains, metabolic & regulatory pathways, shared regulators or targets etc. For model organisms, there is a vast assortment of powerful and friendly analysis tools that include the corresponding databases. With nonmodel organisms, however, the situation is more challenging. Some of the servers do harbor a large number of organisms, so we better off start by searching them for our organism. These include, for example, DAVID [38], KEGG [39] (*see* **Note 6**), STRING [40], and g:Profiler [41]. The R/Bioconductor package clusterProfiler [42] supports organisms available in Bioconductor's OrgDb, and users can build OrgDB objects for additional organisms via the AnnotationHub package (Morgan M (2019). *AnnotationHub: Client to access AnnotationHub resources.* R package version 2.18.0) (*see* **Note 7**). Another option is to search for resources that are specific to certain species. One such example is AgriGO [43], which offers a GO analysis toolkit and database for the agricultural community, currently including 45 species. If our species is closely related to a species that is available in an existing server, we can perform a BLAST search of our transcriptome against that species, and use the BLAST best-hit IDs for the functional enrichment analysis. AgriGO's BLAST4ID tool offers such functionality. In other cases, we may run the BLAST ourselves.

When our organism cannot be handled directly by the above-mentioned approaches, a more generic approach is needed. After completing the transcriptome annotation step (*see* Subheading 3.8) we typically have a BLAST best-hit ID for the majority of our transcripts. These IDs are from different organisms, and unfortunately, most functional enrichment servers will not accept them as

they require to specify the transcriptome's species. Therefore, we may go one step further, and compute ourselves the association between transcriptome genes and universal (non–species-specific) functional categories. Some enrichment tools can accept user-prepared tables of functional category per gene, and continue from there. It is important to remember that enrichment is tested against a "background," which is usually the entire transcriptome. Therefore, biological category assignment should be prepared for all genes in the transcriptome and not just the genes of interest.

GO assignment per gene may be obtained from Trinotate results (*see* above). Alternatively, it can be obtained by running blastx of the transcriptome against UniProt, and extracting the GO IDs of the best BLAST hit per gene through the UniProt "Retrieve/ID mapping" service (https://www.uniprot.org/uploadlists/ and *see* **Note 8**). For KEGG pathway enrichment, the user can assign universal KEGG orthology IDs (KO) to the representative gene sequences of the transcriptome through the KAAS server [44] (*see* explanation below).

Below, we describe several software packages and servers which may be considered as universal enrichment tools, and may thus be very useful for enrichment analysis in new organisms.

3.11.1 BiNGO

BiNGO [45] is an easy-to-use application for GO enrichment analysis of model and nonmodel organisms, which is operated from within the Cytoscape platform [46]. It requests the following input: (1) a list of genes of interest, or multiple gene clusters ("Batch Mode," https://www.psb.ugent.be/cbd/papers/BiNGO/User_Guide.html) (2) a GO ontology file, typically the up-to-date go.obo file from GO website (http://geneontology.org/docs/download-ontology/), and (3) an annotation file, which associates GO term(s) to genes. As explained above, the annotation file should optimally contain all transcriptome genes having a GO annotation. Then, in the "Select reference set" of BiNGO, the user may choose "Use whole annotation as reference set" to use that as the background list for enrichment testing.

Although BiNGO provides several default GO ontologies, as well as annotations for certain organisms, the BiNGO website states that they are not updated and recommends to use user-provided files.

The GO ontology file is by definition non–species-specific, as it only contains the hierarchy of the GO terms and their definition. In addition to go.obo, which contains the core GO ontology, additional GO subsets are available for download at GO website. The GO subsets, also known as "GO slims," give a broad overview of the ontology content without the detail of the specific fine-grained terms.

Annotation files for selected species can also be downloaded from GO website (http://current.geneontology.org/products/pages/downloads.html). Otherwise, the user can prepare a custom annotation file, in a simple format explained at https://www.psb.ugent.be/cbd/papers/BiNGO/Customize.html.

BiNGO results are presented as an interactive graph, which visualizes the significantly enriched GO terms in the context of the GO hierarchy. The graph has a form of a network, with GO terms as the nodes. Node size is proportional to the number of genes in our "genes of interest" list which were annotated to that GO term. The color of the node represents the enrichment p-value. White nodes are not significantly enriched, whereas the other ones have a color scale ranging from yellow (the specified significance cutoff) to dark orange (*p*-value 5 orders of magnitude smaller than the cutoff). Due to the interdependency of functional categories in the GO hierarchy, it is very likely that not one category, but a whole branch of the GO hierarchy lights up as being significantly over-represented. In such cases, interpretation can be more difficult. The darkest orange nodes, which are furthest down the hierarchy, are probably the ones that you are looking for.

When the "Save BiNGO data file in" option in the BiNGO settings window is checked, the GO enrichment results are exported as an easy to understand tab-delimited file.

3.11.2 Blast2GO

Blast2GO, now part of OmicsBox, is a bioinformatics platform for high-quality functional annotation and analysis of genomic datasets. Basic Blast2GO is free of charge, whereas the full version requires a payed subscription. The software is easy to use on a desktop computer. The user will typically upload a FASTA file of the assembled transcriptome, run a blastx search against a protein database, and then perform the "mapping" and "annotation" steps, which will borrow GO terms from the top BLAST hits and select representative GO terms for each transcript. In addition, Blast2GO can scan the transcriptome against the InterPro database of conserved domains and motifs. GO terms of the retrieved InterPro domains can then be transferred to the transcripts and merged with the already existing GO terms. Given a list of genes of interest, Blast2GO can then perform GO enrichment analysis using Fisher's exact test or Gene Set Enrichment Analysis (GSEA) [47] (the latter is only available in the payed version), while using the entire transcriptome as background for the enrichment. In the free version, it is impractical to run the BLAST and InterProScan searches from within Blast2GO, however the BLAST search can be performed outside of Blast2GO as long as the BLAST results are entered to Blast2GO in XML or XML2/JSON format. These formats can be obtained using the BLAST parameter *-outfmt 14* (preferred), *13*, *15*, or *16*. If GO annotations are produced outside of Blast2GO,

they can be loaded to the software after they are converted to "Blast2GO Annotation File (.annot)" format. Detailed explanations are provided in Blast2GO user manual (http://docs.blast2go.com/user-manual/).

3.11.3 clusterProfiler

The R/Bioconductor's clusterProfiler package [42] implements methods to analyze and visualize functional profiles of gene lists and gene clusters. It supports an impressive collection of ontology and pathway databases (*see* https://bioconductor.org/packages/release/bioc/vignettes/clusterProfiler/inst/doc/clusterProfiler.html), and it can test for enrichment using both hypergeometric test and GSEA. It internally supports GO analysis of about 20 species, KEGG analysis with all species that have annotation available in the KEGG database, DAVID annotation and Reactome Pathway (via ReactomePA for several species). In addition, clusterProfiler can be easily used to analyze unsupported organisms. Organisms whose annotated genome is available at the AnnotationHub Web Service can be accessed through the AnnotationHub R package (*see* Subheading 3.11 and **Note 7**). Any organism that is available in Ensembl can be accessed through the biomaRt R package [48] (*see* **Note 9**). Otherwise, the user can provide clusterProfiler *enricher* function with his/her own functional annotation as two tables: TERM2GENE specifies the functional category assignment per gene (e.g., for GO, the columns will be GO ID, Gene ID), and TERM2NAME provides the description of each functional category (e.g., for GO: GO ID, GO term description). *See* an example here: http://guangchuangyu.github.io/2015/05/use-clusterprofiler-as-an-universal-enrichment-analysis-tool/.

ClusterProfiler GO enrichment results can be further processed by the *simplify* function, which calculates similarity of GO terms and removes those highly similar terms by keeping one representative term. This reduces the redundancy in the enrichment results and facilitates their interpretation. Another function, *dropGO*, can be used to remove very general GO terms, by indicating specific unwanted levels of the GO tree.

3.11.4 KEGG

The Kyoto Encyclopedia of Genes and Genomes (KEGG) Pathway database is a collection of manually drawn pathway maps representing known molecular interaction and reaction networks. The maps are represented in terms of the KEGG Orthology (KO) groups so that experimental evidence in specific organisms can be generalized to other organisms through genomic information. Organism-specific pathway maps are computationally generated from the universal maps.

For each pathway, KEGG presents reference maps and species-specific maps. In the "KO" reference map, the genes and proteins are identified by universal "orthology IDs" (e.g., K04527), which

are non–species-specific. This allows for using the KEGG database for organisms that are unavailable in the species-specific maps. To connect our transcriptome genes to KEGG orthology IDs, we may either use an indirect route, through known orthology IDs of the gene's best hit from the annotation step, or a direct route through BLAST against KEGG proteins. Practically, for the indirect way we can submit the best hit IDs to UniProt Retrieve/ID mapping tool, and there, choose to add to the results the "KO" column from the "Phylogenomic" section. This column will contain the KEGG orthology ID. For the direct way, it is recommended to use the KEGG Automatic Annotation Server (KAAS, https://www.genome.jp/kegg/kaas/). In KAAS home page, press on the link to "KAAS job request (BBH method)," upload the sequences of the representative transcript per gene and under the "GENES data set" section choose the organisms against which you wish to compare your sequences. In the results, you will find a link to a text file, which maps KEGG orthology IDs to your sequences.

Next, we can use the "KEGG Mapper – Search&Color Pathway" (https://www.genome.jp/kegg/tool/map_pathway2.html) to overlay the transcriptome genes on KEGG reference maps. It is recommended to upload a table with two columns: KEGG orthology ID and a color indicating whether the gene is significantly upregulated, significantly downregulated, or not DE. Alternatively, the color may indicate the cluster number from clustering analysis (e.g., from CLICK), with additional color for non-DE genes. Do not forget to select "Reference" in the "Search mode" field, and it is advisable to check the "Use uncolored diagrams" option. If your studied organism is present in KEGG (to check, *see* **Note 6**), select "Organism-specific" and specify your organism in the "Search mode" field; in the uploaded table use one of the supported gene IDs (from KEGG, UniProt or NCBI). As a result, you will receive a list of KEGG pathway maps, sorted by the number of genes from the uploaded list found in the pathway. In the pathway maps themselves, the found genes will be colored according to their results in the RNA-Seq analysis, as specified in the uploaded table. Please note that the "KEGG Mapper – Search&Color Pathway" tool does not compute any statistical enrichment test, but only counts the number of found genes per pathways. If you wish to perform KEGG enrichment testing, you may use external tools, such as those described in the next section (Subheading 3.11.5). These tools may need, in addition to KEGG orthology ID per gene, also a table mapping KEGG orthology IDs to KEGG map IDs, and a table specifying KEGG map name for each KEGG map ID. This data may be retrieved through KEGG REST API (https://www.kegg.jp/kegg/rest/keggapi.html) or through the R KEGGREST package (Tenenbaum D (2019). *KEGGREST: Client-side REST access to KEGG*. R package version 1.26.1.).

3.11.5 Tools Supporting Enrichment Analysis with User-Provided Functional Categories

As mentioned above, there are software tools which offer a generic enrichment analysis, where the algorithm is "indifferent" to both the organism and the functional categories. The user will typically provide: a list of gene IDs representing the entire transcriptome (for background calculation); a specification of one or more lists of genes of interest; mapping between gene IDs and "category" IDs; and, optionally, a description of each "category." The format of the input files is software-specific, and therefore the user will usually have to covert the data to the required format on his own.

R users may find clusterProfiler (*see* Subheading 3.11.3) and goseq [49] very convenient. clusterProfiler can analyze multiple gene clusters simultaneously, while goseq can take into account gene lengths.

For nonprogrammers, the WebGestalt [50] web server will be of great help. In WebGestalt homepage, select "others" in the "Organism of Interest" field; upload files with category-to-gene and category-to-description tables in the "Upload functional Database" and "Upload Database Description File" fields, respectively (*see* **Note 10**); upload IDs of the genes of interest in "Upload Gene List" and IDs of all transcriptome genes in the "Select Reference Set" field. In addition to the default Over Representation Analysis (ORA), which uses the hypergeometric test, WebGestalt also provides two more advanced methods for enrichment testing, namely, GSEA and Network Topology–based Analysis (NTA) [51]. The obtained enriched categories may be further post-processed to reduce their redundancy through either affinity propagation or weighted set cover, as explained in the WebGestalt publication [50]. WebGestalt core calculations are also available in the WebGestaltR R package [50] (https://cran.r-project.org/web/packages/WebGestaltR/index.html).

Desktop applications enabling enrichment analysis with user-provided functional categories include Expander [35], FunRich [52], and GSEA [47] (*see* **Note 11**).

3.12 A Ready-to-Use Pipeline

A full pipeline for RNA-Seq analysis of nonmodel organisms, starting from raw sequence reads down to statistical, clustering and enrichment analysis, is available on the NeatSeq-Flow platform (https://neatseq-flow.readthedocs.io/projects/neatseq-flow-modules/en/latest/). Nonprogrammers can execute the pipeline through a friendly graphical user interface, whereas advanced users may run it through the command line. Both interfaces support further customization of the pipeline by the user. Workflow execution is parallelized on both samples and analysis steps, and progress can be tracked in real time. Installation of NeatSeq-Flow and its required analysis programs is easily achieved with Conda, thus enabling its usage on a local computer, or, preferably, computer cluster. Alternatively, NeatSeq-Flow may be installed on the cloud using a dedicated script.

The NeatSeq-Flow RNA-Seq pipeline for nonmodel organisms includes many of the programs that were described in this article. A schematic diagram of the workflow is presented in Fig. 1. Installation instructions and documentation are available at https://neatseq-flow.readthedocs.io/projects/neatseq-flow-modules/en/latest/.

4 Notes

1. It is possible to improve gene prediction and annotation in a draft genome. Importantly, AUGUSTUS [53] enables integration of raw RNA-Seq reads as part of the gene prediction process. The section "RNAseq integration (raw reads)" in AUGUSTUS Tutorials page (http://bioinf.uni-greifswald.de/bioinf/wiki/pmwiki.php?n=Augustus.Augustus) describes several alternatives for incorporating the reads in the prediction, each using a different read mapper. A modification which uses STAR for mapping is explained here: https://fossies.org/linux/augustus/docs/tutorial2015/ittrain.html. Another option is to use BRAKER [54], which performs RNA-Seq-based genome annotation with GeneMark-ET and AUGUSTUS. Functional annotation of the predicted genes may be carried out as described in Subheading 3.8.
2. Read more on mapping to a genomic reference that is evolutionarily diverged from the sequenced species in [11] under "Read mapping to reference genome."
3. It is not recommended to use as a reference a de novo assembled transcriptome from another experiment, since that transcriptome may not contain transcripts that were expressed only in the conditions used in the current experiment.
4. Another option, which stands in between searching the entire UniProt and the Swiss-Prot section, is using one of the clustered versions of UniProt called UniRef (UniProt Reference Clusters) [55]. For example, in UniRef90, each cluster is composed of sequences that have at least 90% sequence identity to, and 80% overlap with, the longest sequence of the cluster. In this case, the BLAST search is performed against the representative (best annotated) protein per cluster.
5. By default, Trinotate runs its BLAST searches vs. the Swiss-Prot section of UniProt. For organisms from less studied lineages, this may result in reduced proportion of annotated genes. It is possible though to search any other protein database and load the results as a custom protein database to Trinotate. Searching Swiss-Prot, however, is critical to Trinotate, because that is where it retrieves the various KEGG, GO, Eggnog, and other annotations from.

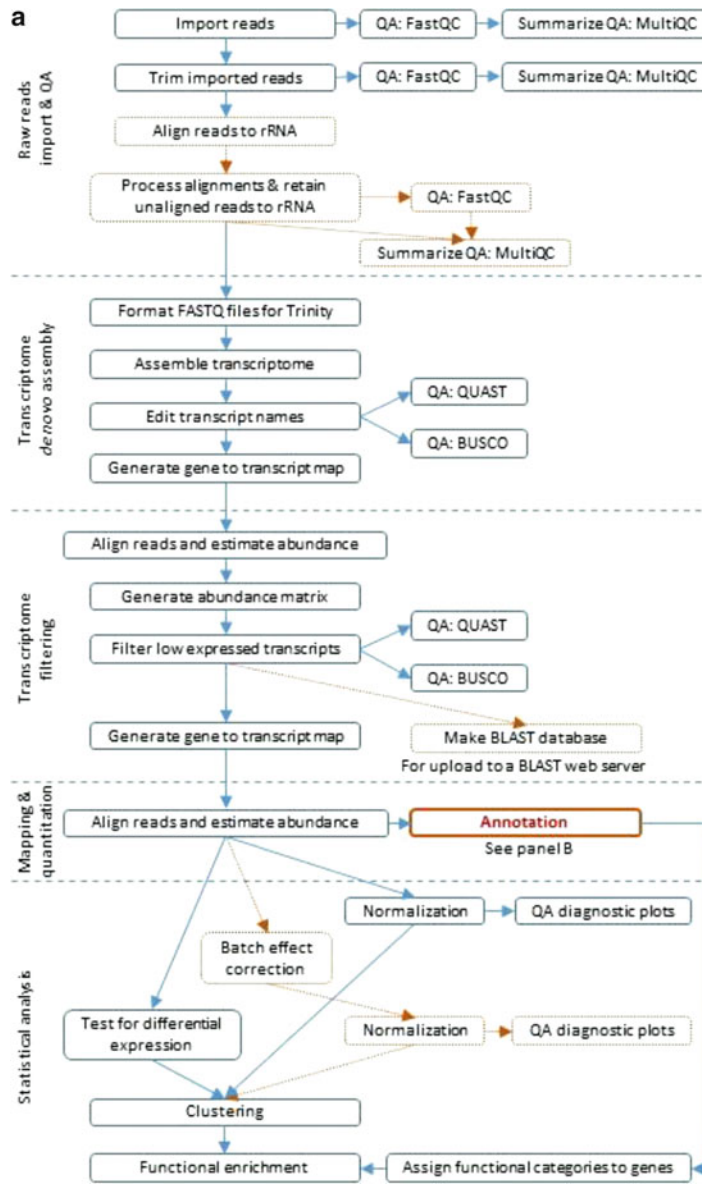


Fig. 1 NeatSeq-Flow ready-to-use pipeline for RNA-Seq analysis in nonmodel organisms. Panel **a**: main workflow. Panel **b**: transcriptome annotation part. Plain rectangles represent workflow steps. Bold rectangles connect between the panels. Dashed rectangles represent optional steps. Software tools used by the pipeline: Raw reads import and QA: Trim Galore!, FastQC, MultiQC, BWA-MEM, Samtools; Transcriptome de novo assembly: Trinity, QUAST, BUSCO; Transcriptome filtering: Trinity, RSEM, bowtie2, QUAST, BUSCO, BLAST; Mapping and quantitation: Trinity, RSEM, bowtie2; Statistical analysis: DESeq2, SVA, pheatmap, eclust (factoextra), biomaRt, clusterProfiler; Select representative transcripts: Trinity, QUAST, BUSCO; Function prediction: Trinity, Trinotate, Transdecoder. In addition, most parts also contain in-house scripts. The NeatSeq-Flow pipeline for nonmodel organisms was developed by Menachem Sklarz, Liron Levin, and Olabiyi Obayomi, the Bioinformatics Core Facility, Ben-Gurion University

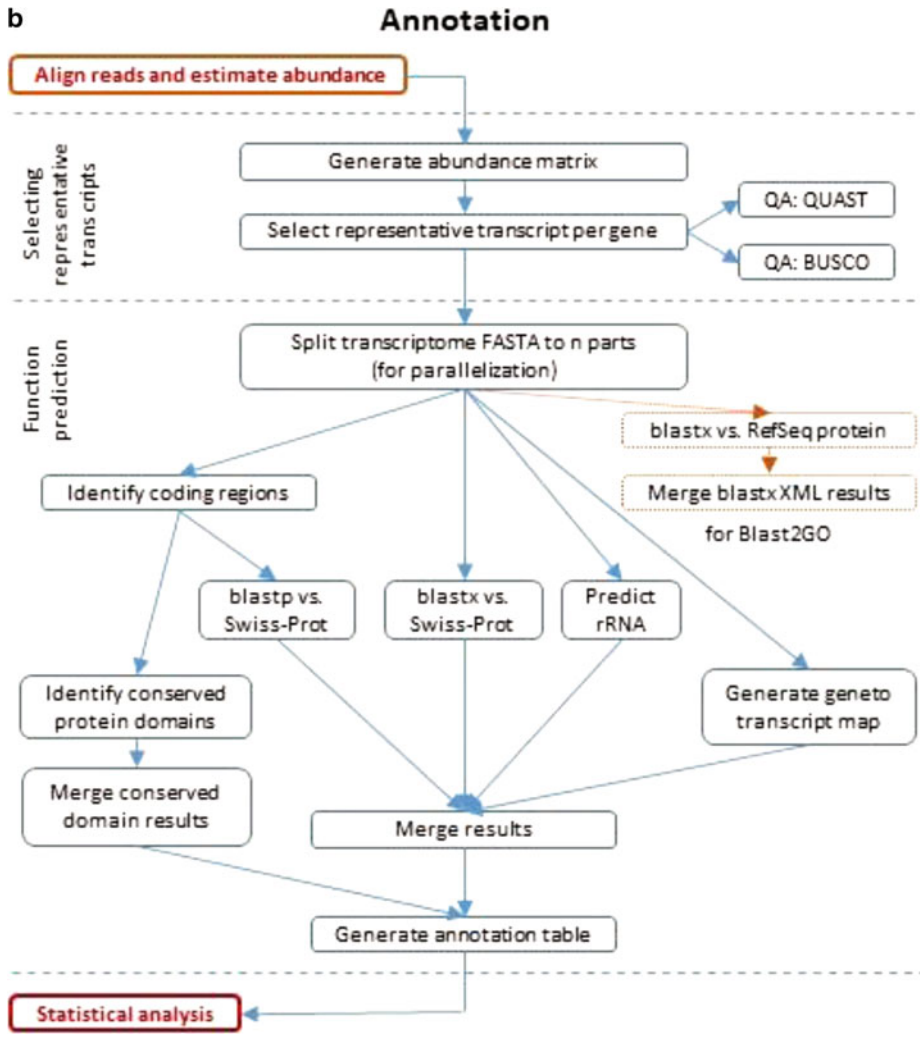


Fig. 1 (continued)

6. A list of KEGG organisms is available at https://www.genome.jp/kegg/catalog/org_list.html
7. Instructions on using clusterProfiler and AnnotationHub are available at https://guangchuangyu.github.io/2016/01/go-analysis-using-clusterprofiler/ and https://bioconductor.org/packages/devel/bioc/vignettes/AnnotationHub/inst/doc/AnnotationHub-HOWTO.html
8. To assign GO IDs to a list of UniProt IDs, enter your IDs to UniProt's Retrieve ID/mapping page, request to map them from UniProtKB AC/ID to UniProtKB and press on "Submit." Press on the "Columns" button above the results table, and mark a "V" next to "Gene Ontology IDs" under the section "Gene Ontology (GO)." Then press on the "Save"

button on top. A column with GO IDs is now added to your UniProt IDs list. Press on the Download button above the table and select the Tab-separated format to download the results to your computer. For programmatic access to the Retrieve ID/mapping tool see https://www.uniprot.org/help/uploadlists.

9. GO annotations retrieved from biomaRt should be expanded to include ancestor ("parent") GO terms in addition to the direct terms, using the clusterProfiler *buildGOmap* function.
10. Examples with required formats for WebGestalt are available at http://www.webgestalt.org/external_example/externalExample.php and explained in the User's manual http://www.webgestalt.org/WebGestalt_2019_Manual.pdf, Sect. 3.1 "Select a functional database / For 'others' organism."
11. In Expander User Manual, http://acgt.cs.tau.ac.il/expander/help/ver8.0Help/html/, look for "Custom Enrichment Analysis." In FunRich documentation look for "custom background database." In GSEA User Manual, http://software.broadinstitute.org/gsea/doc/GSEAUserGuideFrame.html, go to "Preparing Data Files for GSEA" / "Gene Sets" / "Creating Gene Sets."

References

1. Cheng H, Wang Y, Sun MA (2018) Comparison of gene expression profiles in nonmodel eukaryotic organisms with RNA-Seq. Methods Mol Biol 1751:3–16. https://doi.org/10.1007/978-1-4939-7710-9_1
2. Eldem V, Zararsiz G, Taşçi T et al (2017) Transcriptome analysis for non-model organism: current status and best-practices. In: Marchi F (ed) Applications of RNA-Seq and omics strategies. IntechOpen, pp 55–77. https://doi.org/10.5772/intechopen.68983
3. Sundaram A, Tengs T, Grimholt U (2017) Issues with RNA-seq analysis in non-model organisms: a salmonid example. Dev Comp Immunol 75:38–47. https://doi.org/10.1016/j.dci.2017.02.006
4. Fu S, Ma Y, Yao H et al (2018) IDP-denovo: de novo transcriptome assembly and isoform annotation by hybrid sequencing. Bioinformatics 34(13):2168–2176. https://doi.org/10.1093/bioinformatics/bty098
5. Ning G, Cheng X, Luo P et al (2017) Hybrid sequencing and map finding (HySeMaFi): optional strategies for extensively deciphering gene splicing and expression in organisms without reference genome. Sci Rep 7:43793. https://doi.org/10.1038/srep43793
6. Bayega A, Wang YC, Oikonomopoulos S et al (2018) Transcript profiling using long-read sequencing technologies. Methods Mol Biol 1783:121–147. https://doi.org/10.1007/978-1-4939-7834-2_6
7. Stark R, Grzelak M, Hadfield J (2019) RNA sequencing: the teenage years. Nat Rev Genet. https://doi.org/10.1038/s41576-019-0150-2
8. Wang B, Kumar V, Olson A et al (2019) Reviving the transcriptome studies: an insight into the emergence of single-molecule transcriptome sequencing. Front Genet 10:384. https://doi.org/10.3389/fgene.2019.00384
9. Zhao L, Zhang H, Kohnen MV et al (2019) Analysis of transcriptome and epitranscriptome in plants using PacBio Iso-Seq and nanopore-based direct RNA sequencing. Front Genet 10:253. https://doi.org/10.3389/fgene.2019.00253
10. Benjamin AM, Nichols M, Burke TW et al (2014) Comparing reference-based RNA-Seq mapping methods for non-human primate

data. BMC Genomics 15:570. https://doi.org/10.1186/1471-2164-15-570

11. Paya-Milans M, Olmstead JW, Nunez G et al (2018) Comprehensive evaluation of RNA-seq analysis pipelines in diploid and polyploid species. Gigascience 7(12). https://doi.org/10.1093/gigascience/giy132
12. Bolger AM, Lohse M, Usadel B (2014) Trimmomatic: a flexible trimmer for Illumina sequence data. Bioinformatics 30 (15):2114–2120. https://doi.org/10.1093/bioinformatics/btu170
13. Ewels P, Magnusson M, Lundin S et al (2016) MultiQC: summarize analysis results for multiple tools and samples in a single report. Bioinformatics 32(19):3047–3048. https://doi.org/10.1093/bioinformatics/btw354
14. Normand R, Yanai I (2013) An introduction to high-throughput sequencing experiments: design and bioinformatics analysis. Methods Mol Biol 1038:1–26. https://doi.org/10.1007/978-1-62703-514-9_1
15. Li H (2013) Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. arXiv arXiv:1303.3997
16. Li H, Handsaker B, Wysoker A et al (2009) The Sequence Alignment/Map format and SAMtools. Bioinformatics 25 (16):2078–2079. https://doi.org/10.1093/bioinformatics/btp352
17. Menzel P, Ng KL, Krogh A (2016) Fast and sensitive taxonomic classification for metagenomics with Kaiju. Nat Commun 7:11257. https://doi.org/10.1038/ncomms11257
18. Grabherr MG, Haas BJ, Yassour M et al (2011) Full-length transcriptome assembly from RNA-Seq data without a reference genome. Nat Biotechnol 29(7):644–652. https://doi.org/10.1038/nbt.1883
19. Haas BJ, Papanicolaou A, Yassour M et al (2013) De novo transcript sequence reconstruction from RNA-seq using the Trinity platform for reference generation and analysis. Nat Protoc 8(8):1494–1512. https://doi.org/10.1038/nprot.2013.084
20. Wang S, Gribskov M (2017) Comprehensive evaluation of de novo transcriptome assembly programs and their effects on differential gene expression analysis. Bioinformatics 33 (3):327–333. https://doi.org/10.1093/bioinformatics/btw625
21. Li B, Dewey CN (2011) RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome. BMC Bioinformatics 12:323. https://doi.org/10.1186/1471-2105-12-323
22. Love MI, Huber W, Anders S (2014) Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biol 15(12):550. https://doi.org/10.1186/s13059-014-0550-8
23. Fu L, Niu B, Zhu Z et al (2012) CD-HIT: accelerated for clustering the next-generation sequencing data. Bioinformatics 28 (23):3150–3152. https://doi.org/10.1093/bioinformatics/bts565
24. Srivastava A, Sarkar H, Malik L et al (2016) Accurate, fast and lightweight clustering of de novo transcriptomes using fragment equivalence classes. arXiv arXiv:1604.03250
25. Gurevich A, Saveliev V, Vyahhi N et al (2013) QUAST: quality assessment tool for genome assemblies. Bioinformatics 29(8):1072–1075. https://doi.org/10.1093/bioinformatics/btt086
26. Seppey M, Manni M, Zdobnov EM (2019) BUSCO: assessing genome assembly and annotation completeness. Methods Mol Biol 1962:227–245. https://doi.org/10.1007/978-1-4939-9173-0_14
27. Gotz S, Garcia-Gomez JM, Terol J et al (2008) High-throughput functional annotation and data mining with the Blast2GO suite. Nucleic Acids Res 36(10):3420–3435. https://doi.org/10.1093/nar/gkn176
28. Bryant DM, Johnson K, DiTommaso T et al (2017) A tissue-mapped axolotl de novo transcriptome enables identification of limb regeneration factors. Cell Rep 18(3):762–776. https://doi.org/10.1016/j.celrep.2016.12.063
29. Robinson MD, McCarthy DJ, Smyth GK (2010) edgeR: a bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26 (1):139–140. https://doi.org/10.1093/bioinformatics/btp616
30. Robinson MD, Oshlack A (2010) A scaling normalization method for differential expression analysis of RNA-seq data. Genome Biol 11 (3):R25. https://doi.org/10.1186/gb-2010-11-3-r25
31. McCarthy DJ, Chen Y, Smyth GK (2012) Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation. Nucleic Acids Res 40 (10):4288–4297. https://doi.org/10.1093/nar/gks042
32. Leek JT, Johnson WE, Parker HS et al (2012) The sva package for removing batch effects and other unwanted variation in high-throughput experiments. Bioinformatics 28(6):882–883.

https://doi.org/10.1093/bioinformatics/bts034

33. Kucukural A, Yukselen O, Ozata DM et al (2019) DEBrowser: interactive differential expression analysis and visualization tool for count data. BMC Genomics 20(1):6. https://doi.org/10.1186/s12864-018-5362-x
34. Metsalu T, Vilo J (2015) ClustVis: a web tool for visualizing clustering of multivariate data using Principal Component Analysis and heatmap. Nucleic Acids Res 43(W1):W566–W570. https://doi.org/10.1093/nar/gkv468
35. Hait TA, Maron-Katz A, Sagir D et al (2019) The EXPANDER integrated platform for transcriptome analysis. J Mol Biol 431 (13):2398–2406. https://doi.org/10.1016/j.jmb.2019.05.013
36. Sharan R, Maron-Katz A, Shamir R (2003) CLICK and EXPANDER: a system for clustering and visualizing gene expression data. Bioinformatics 19(14):1787–1799. https://doi.org/10.1093/bioinformatics/btg232
37. Reimand J, Isserlin R, Voisin V et al (2019) Pathway enrichment analysis and visualization of omics data using g:Profiler, GSEA, Cytoscape and EnrichmentMap. Nat Protoc 14 (2):482–517. https://doi.org/10.1038/s41596-018-0103-9
38. da Huang W, Sherman BT, Lempicki RA (2009) Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. Nat Protoc 4(1):44–57. https://doi.org/10.1038/nprot.2008.211
39. Kanehisa M, Furumichi M, Tanabe M et al (2017) KEGG: new perspectives on genomes, pathways, diseases and drugs. Nucleic Acids Res 45(D1):D353–D361. https://doi.org/10.1093/nar/gkw1092
40. Szklarczyk D, Gable AL, Lyon D et al (2019) STRING v11: protein-protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. Nucleic Acids Res 47(D1): D607–D613. https://doi.org/10.1093/nar/gky1131
41. Raudvere U, Kolberg L, Kuzmin I et al (2019) g:Profiler: a web server for functional enrichment analysis and conversions of gene lists (2019 update). Nucleic Acids Res 47(W1): W191–W198. https://doi.org/10.1093/nar/gkz369
42. Yu G, Wang LG, Han Y et al (2012) clusterProfiler: an R package for comparing biological themes among gene clusters. OMICS 16 (5):284–287. https://doi.org/10.1089/omi.2011.0118
43. Tian T, Liu Y, Yan H et al (2017) agriGO v2.0: a GO analysis toolkit for the agricultural community, 2017 update. Nucleic Acids Res 45 (W1):W122–W129. https://doi.org/10.1093/nar/gkx382
44. Moriya Y, Itoh M, Okuda S et al (2007) KAAS: an automatic genome annotation and pathway reconstruction server. Nucleic Acids Res 35 (Web Server issue):W182–W185. https://doi.org/10.1093/nar/gkm321
45. Maere S, Heymans K, Kuiper M (2005) BiNGO: a Cytoscape plugin to assess overrepresentation of gene ontology categories in biological networks. Bioinformatics 21 (16):3448–3449. https://doi.org/10.1093/bioinformatics/bti551
46. Shannon P, Markiel A, Ozier O et al (2003) Cytoscape: a software environment for integrated models of biomolecular interaction networks. Genome Res 13(11):2498–2504. https://doi.org/10.1101/gr.1239303
47. Subramanian A, Tamayo P, Mootha VK et al (2005) Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. Proc Natl Acad Sci U S A 102(43):15545–15550. https://doi.org/10.1073/pnas.0506580102
48. Durinck S, Moreau Y, Kasprzyk A et al (2005) BioMart and Bioconductor: a powerful link between biological databases and microarray data analysis. Bioinformatics 21 (16):3439–3440. https://doi.org/10.1093/bioinformatics/bti525
49. Young MD, Wakefield MJ, Smyth GK et al (2010) Gene ontology analysis for RNA-seq: accounting for selection bias. Genome Biol 11 (2):R14. https://doi.org/10.1186/gb-2010-11-2-r14
50. Liao Y, Wang J, Jaehnig EJ et al (2019) WebGestalt 2019: gene set analysis toolkit with revamped UIs and APIs. Nucleic Acids Res 47 (W1):W199–W205. https://doi.org/10.1093/nar/gkz401
51. Wang J, Ma Z, Carr SA et al (2017) Proteome profiling outperforms transcriptome profiling for coexpression based gene function prediction. Mol Cell Proteomics 16(1):121–134. https://doi.org/10.1074/mcp.M116.060301
52. Pathan M, Keerthikumar S, Ang CS et al (2015) FunRich: an open access standalone functional enrichment and interaction network analysis tool. Proteomics 15(15):2597–2601. https://doi.org/10.1002/pmic.201400515
53. Stanke M, Waack S (2003) Gene prediction with a hidden Markov model and a new intron

submodel. Bioinformatics 19 Suppl 2: ii215–ii225. https://doi.org/10.1093/bioinformatics/btg1080

54. Hoff KJ, Lange S, Lomsadze A et al (2016) BRAKER1: unsupervised RNA-Seq-based genome annotation with GeneMark-ET and AUGUSTUS. Bioinformatics 32(5):767–769. https://doi.org/10.1093/bioinformatics/btv661
55. Suzek BE, Wang Y, Huang H et al (2015) UniRef clusters: a comprehensive and scalable alternative for improving sequence similarity searches. Bioinformatics 31(6):926–932. https://doi.org/10.1093/bioinformatics/btu739

Chapter 9

Deep Learning Applied on Next Generation Sequencing Data Analysis

Artem Danilevsky and Noam Shomron

Abstract

Deep learning is defined as the group of computational techniques allowing for the discovery of latent information within large amounts of data. Recently, many fields have seen the immense potential of deep learning to solve various tasks in ways which outperformed many other traditional methods. Genomic research could be the next frontier to take advantage of deep learning, as it has the perfect combination of vast amounts of data and diverse tasks. Here we present the platform we generated to combine deep learning and genomic sequencing data. We tested the platform on publicly available sequencing data from the gut microbiome of cancer patients. We showed that our platform is capable of classifying patients with higher accuracy than other methods, with some caveats. Overall, we believe genomic research is the next frontline for deep learning as there are exciting avenues waiting to be explored. We think that our platform, presented here, could serve as the basis for such future research.

Key words Deep learning, Computational techniques, Genomic research, Cancer detection

1 Introduction

1.1 Deep Learning

Deep learning is a recently reemerging group of computational techniques. Deep learning aims to capture the latent representation of complex data by training a deep neural network model on large amounts of data using backpropagation [1]. The first deep learning model was presented in 1968 [2] and since then, many deep learning studies improved on it throughout the years [3, 4]. The recent rise in popularity of deep learning is attributed, at least in part, by a 2012 work where Krizhevsky et al. utilized a neural network model which achieved higher image classification accuracy than any other existing algorithms [5]. Subsequently, the rise in computational power and available large data sets, together with the combination of previously developed algorithms in this field, allowed researchers to train complex deep learning models very efficiently. After the potential of deep learning was shown in 2012, it was used on various types of data such as visual, audio,

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_9,

and text data. Deep learning techniques produced state-of-the-art results in various tasks like classification and detection [6–9], proving this technology to be very versatile and viable for different types of unstructured data.

1.2 Deep Learning in Medicine

Along with many other fields, within the field of medical research, there is a continuous rise in studies involving deep learning to solve various tasks—even despite the oftentimes limited amount of clean, tagged, and freely available medical data. Since deep learning has many well established models and frameworks to process and train with visual data, the most readily available application of deep learning in the medical field is in image analysis [10]. When building models for visual data analysis, such as images from medical scanning technologies or microscopy images, one can likely use an established deep learning model such as ResNet [11] or VGG [12]. The main challenge for visual data analysis, however, is fine-tuning the models [13] for specific medical tasks while using a limited number of samples to train the models [14, 15]. Recently, the advent of deep learning models has also been applied to analyzing medical textual data. Many studies tackled the analysis of electronic health records (EHR), where the information includes a mixture of data, that is, structured, free-text, and time series data. Combining these three types of data, however, proves to be challenging when tested with many possible deep learning architectures, and where the size of the data and methods of cleaning and organizing the data greatly affects the success of the model [16, 17]. The potential to use deep learning techniques within the medical field is clear, however, there are many hurdles yet to overcome and new models to create in order to accommodate and properly analyze the different types of medical data.

1.3 Deep Learning in Genomic Research

The genomic field is starting to see its fair share of deep learning research as well [18]. This will probably increase as it did in other fields (*see* Fig. 1). Genomic data became highly abundant with the development of affordable and efficient sequencing technologies. These are known as next generation sequencing (NGS), or deep sequencing. For example, the Sequence Read Archive (SRA) database currently stores over 10 PB of nucleotide sequence data [19]. Another example of the vast amounts of genomic data that exists is in an individual experiment we used in our study that includes fecal microbiota sequencing data for early colorectal cancer detection [20]. The data possesses over 700 GB of information (available online). Although the scale of data is large, it is important to note there are only a few individual samples which could pose a challenge for deep learning analysis (we will address this later in the text). Compared to other fields which have datasets ranging from few megabytes to several gigabytes, this huge amount of genomic data can be used to train even deeper and more complex models to

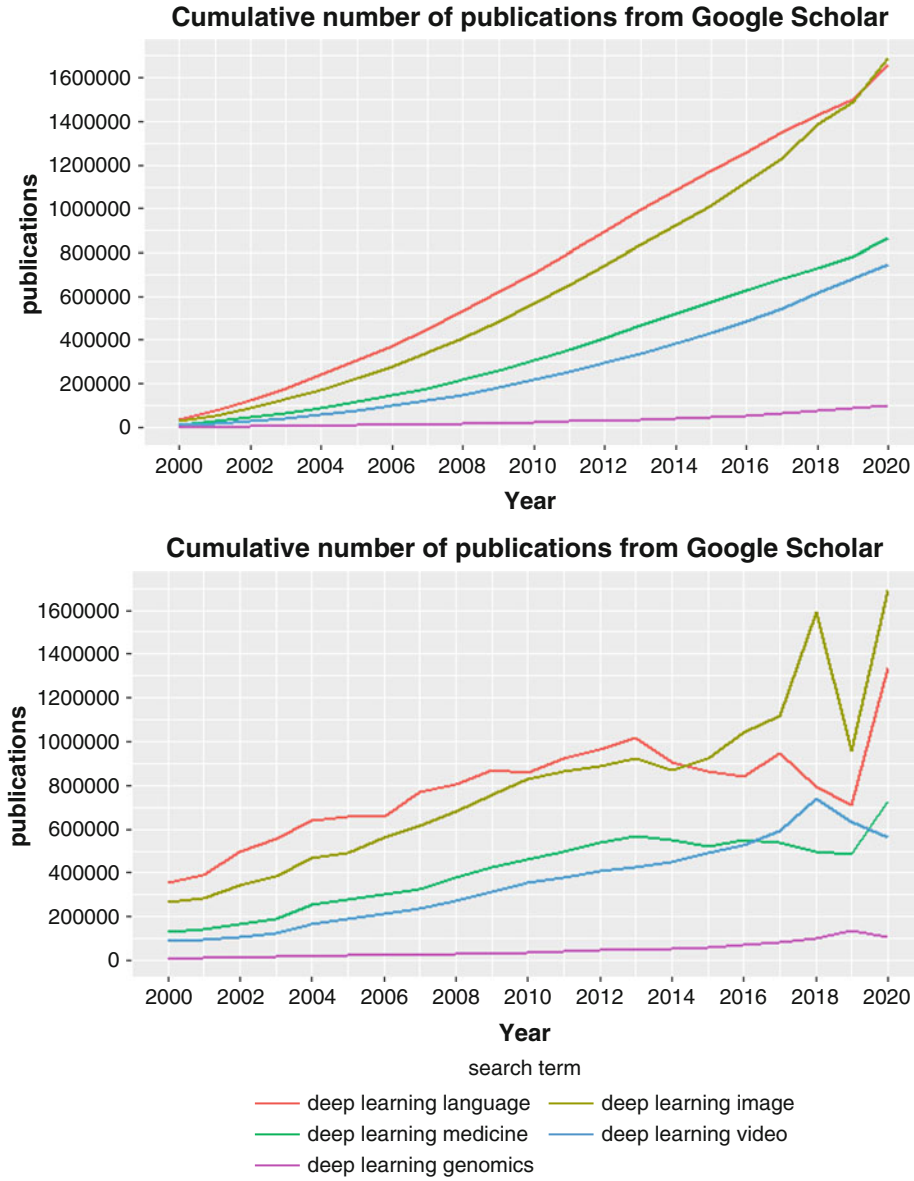


Fig. 1 The number of publications with the term "deep learning" and "X," where X is either "image," "video," "genomics," "medicine," "language." Top, a plot of cumulative number of publications per search term in Google Scholars; Bottom, similarly to the top graph only presenting the raw number of publications. Values for year 2020 were estimated based on the first 2 months

gain insights and perform difficult tasks in genomics. In one of the earlier studies involving deep learning in genomic research, deep learning was used to create a variant caller termed DeepVariant [21]. Variant calling is the process of analyzing genetic sequencing and determining what different variants are present in the sequenced DNA compared to a reference genome. This deep

convolutional neural network approach outperformed other common variant callers and furthermore, was able to run on several different sequencing platforms. Other studies used raw sequencing information like analyzing individual reads to diagnose cancer [22] and identify viral genomes [23]. Other groups successfully employed deep learning for functional genomics [24], for finding patterns in DNA sequences [25], and for looking for enhancers in the DNA sequence [26].

The genetic code is a complex and sophisticated biological system that stores and propagates important information that is passed onto generations. Human language could be described similarly, yet it is a much simpler model that was developed chronologically very recently compared to the genetic code. In natural language, in order to understand a large body of text, one needs to understand the local relationship between neighboring words and to connect concepts found in different parts of the text. Similarly, in the genetic code, it is important to understand the local context of the nucleotides. For example, a change in one nucleotide in the DNA could change a triplet of nucleotides from a coding region and lead to the translation into a different amino acid, consequently changing a given protein's structure and function. In addition, it is equally important to understand the long-distance relations between different parts of the genome, such as between promoters, enhancers, transcription factors, and others, which are relations that could have major genomic consequences. Thus, in this study, we borrow many of the ideas from a more established deep learning field—natural language processing (NLP)—and apply them to genomics.

1.4 Deep Learning for Cancer Diagnosis

In our study we approached cancer diagnosis using genomic sequences via a unique perspective. We used deep learning to predict a label (healthy or sick) for each patient using only the raw reads from a sequencing experiment without aligning the reads to the genome or performing any preprocessing. This task presented several interesting challenges. The core issue was the way we combined the genomic data with the deep learning model. Some of the previous studies used models designed for visual data [21], while others employed those appropriate for textual data [27]. However, since we tackled the problem from a new angle, we decided to build a custom model and not to rely on pretrained models. After creating the deep learning model we faced the challenge of loading the large amount of data for processing during the training of the model. Since the amount of data is by far much larger than other types of deep learning datasets, and no standardized method for loading genomic data to a deep learning model exists, we were required to optimize the data input and to find appropriate tactics to allow training in an efficient manner. Finally, after putting together all the components we tested the platform on a publicly

available genomic dataset where the researcher's goal was to detect colorectal cancer by sequencing stool samples of patients and healthy individuals. They did this by building a classifier based on the metagenomics composition of the samples [20]. We used our deep learning model to tell apart colorectal cancer patients from healthy individuals.

2 Methods

2.1 Description of Data

In order to train a classifier for two labels (healthy vs cancer patient) we used the sequencing data from 53 cancer patients and 61 healthy individuals from the French population [20]. The samples were randomly separated into two groups, 80% of samples were used to train the model while the rest were used exclusively to test the accuracy after the training. This separation between samples was crucial for the correct assessment of our model's accuracy. This method of sample-based separation eliminates the possibility that some technical variance originating from the laboratory protocols could have been used by the model to "memorize" each sample and classify the samples based on some arbitrary technical artifact which is unrelated to the illness. To check the ability of our model to generalize between samples from a different distribution we used 16 samples of cancer patients and healthy individuals from Germany from the same study.

The researchers that generated the data removed from the dataset any reads which aligned to the human genome; therefore, the majority of the reads originated from bacterial genomes of the gut microbiome. The main principle by which the researchers classified the samples is based on aligning the reads to a bacterial genome database, creating a list of the bacterial abundance for each sample, and training a penalized regression model to classify samples based on the metagenomic composition. Their method then achieved better results compared to other novel cancer detection techniques such as immunochemical fecal occult blood test (FOBT) and epigenetic assays [20].

2.2 Data Manipulation and Processing

The large data size we work with (which is around 400 GB for this project) prevents us from loading all the data to memory at once like many machine learning workflows do. Therefore, we were required to find an efficient strategy to process and load this data throughout the training. Moreover, it would not be technically possible to process all the reads of one sample in one batch as it would require an extensive amount of computations. For this reason, we decided to present the deep learning model with a very limited amount of reads in each batch. After some empirical testing we decided to use only 4096 reads for each batch, this was near the maximum limit of reads we could process simultaneously

in a modest size deep learning model. There are several advantages in using a small amount of reads for the classification. For one, it allows us to obtain multiple batches from a single sample, which is useful for the data intensive requirement of deep learning training. Another critical advantage is the possibility of using the model on samples that have an extremely low number of reads, which could be useful when performing large-scale screening with many samples multiplexed together. A final requirement for our data processing is to shuffle the batches of reads between each training epoch, otherwise the model could possibly "memorize" the combination of reads for each sample instead of making a decision based on the genomic information in the reads.

In light of these requirements and limitations we started building the data-loader with the most basic approach by using a python wrapper for Samtools [28] called Pysam [29]. This approach involved shuffling the indices of each read into random groups of 4096 reads for each sample at the beginning of an epoch, then at the time of training retrieving the reads from the batch of 4096, one by one, straight from the BAM files which stored them. This method should have worked in theory but in practice there was too much overhead computation and the training was very slow with the bottleneck being the data loading. For our next attempt we decided to save the reads in a separate binary file storing only the nucleotide sequence, here we tested both the standard python method for reading binary files and the memory mapped file support (mmap package from the standard python library). In both cases the method worked very well when limiting the number of lines to retrieve from the file. Unfortunately when working with the entire file the cache/buffer memory of the machine filled up quickly and the training halted. In our final attempt we added another modification to the data-loader; at the start of each epoch the binary files storing the reads were shuffled, at training time batches of 4096 of adjacent reads were extracted from the files. Thus, instead of accessing the file 4096 times for each read in a batch our method accesses the file only once and, in doing so, minimizes the amount of input/output (IO) operations required.

2.3 Description of Model Structure

The first part of any deep learning model consists of layers for manipulating the input data. Therefore, in order to design these layers, our first question was how to present the nucleotide sequence to the model, since deep learning models can only accept numbers as input and not characters. One common approach to represent nucleotide sequence consisting of the four letters "ATGC" is to turn it into a one-hot vector such as a matrix with four rows (for each nucleotide) with the same length as the nucleotide sequence. We compared this approach to a more straightforward approach where each nucleotide is converted to a different integer ("ATGC" - > [0, 1, 2, 3]), both methods showed similar

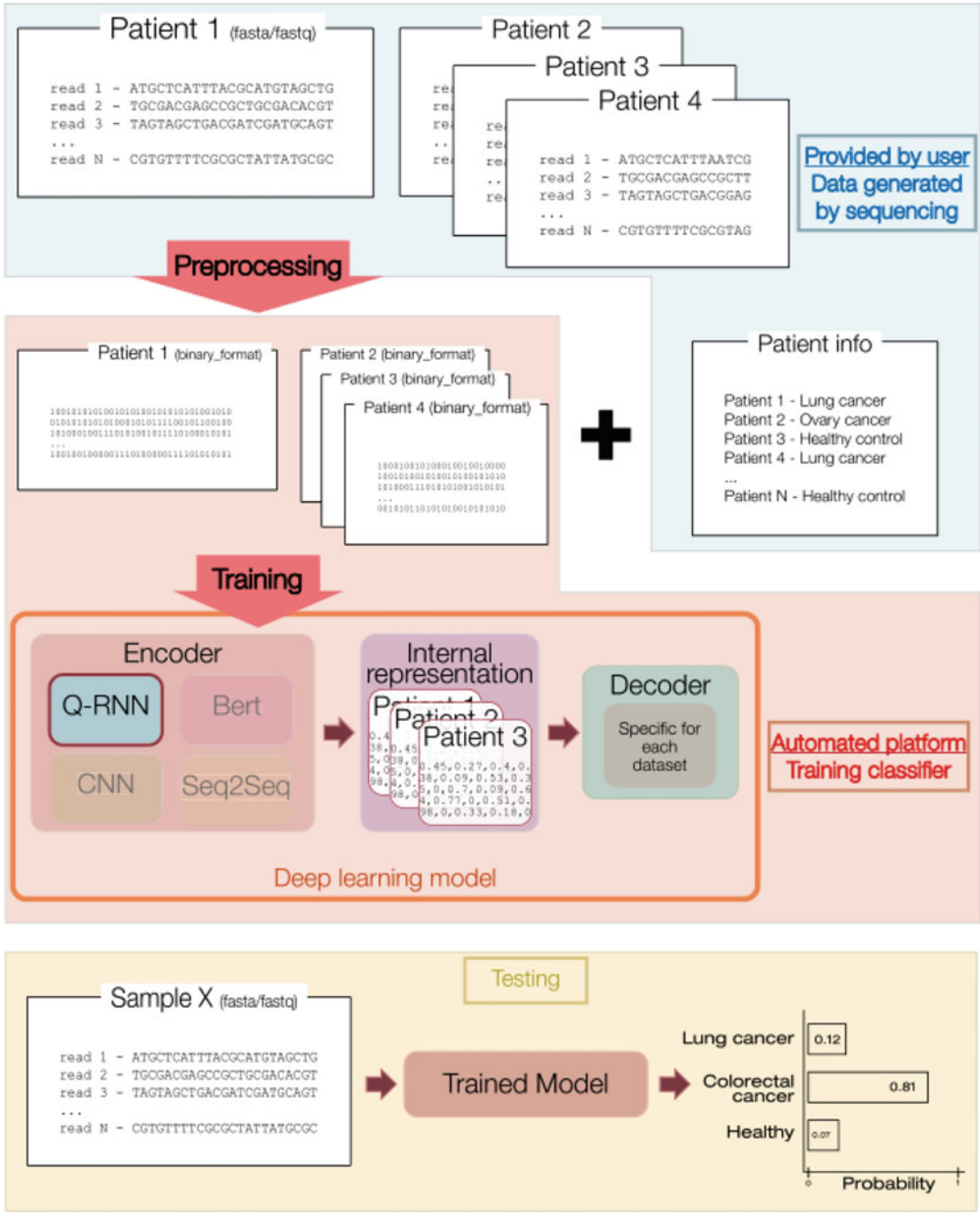


Fig. 2 A schematic representation of the deep learning platform we used

results after training. We continued with the second approach, because this method allows for easier model adaptability to future input with different notations such as the addition of "N" or additional modified nucleotides.

After we understood how the input would look like, we proceeded to design the model itself. The main principal for designing our model was separating the model into two parts; the "encoder" which receives the raw reads and produces a representation vector of each read and the "decoder" which receives the representation vector of a group of reads and classifies the sample into a cancer or healthy type. There are several reasons to separate the entire task into two models, the most important of which is to train the encoder on large amount of data through unsupervised learning and to later use the encoder for specific tasks with minimal additional training. The second reason is to reduce GPU memory usage during training since we can train the models separately. Another feature we added to our platform design is modularity, we hypothesized that certain deep learning architectures could be better for some tasks but not for others; therefore, we built our platform to allow for easy exchange between different encoder and decoder models (*see* Fig. 2).

For the default encoder we decided to use Quasi-Recurrent Neural Network (QRNN) architecture for its high efficiency during training and for its comparable results to other RNN models [30]. In addition, the platform allows for the possibility to change the encoder to an LSTM or a GRU type model, though through testing results, it seems that an LSTM or GRU type model would perform much slower and with similar or worse results. The encoder produces representation vectors for a group of reads and passes those vectors to the decoder to perform the classification of the sample. The decoder in our platform is a custom built model consisting of several 1D-CNN layers, the most unique aspect of the decoder is the fact that convolutions are performed in two dimensions of a group of reads; meaning that several 1D-CNN layers perform convolutions using all of the reads but look at several values of the representation vector in the kernel (column convolutions) while other sets of 1D-CNN layers look at only several representation vectors in one convolution but the whole length of the vector is considered (row convolutions). The result produced by the CNN layers is combined with a bilinear layer and passed through additional linear layers to finally produce a prediction for the sample.

3 Results and Discussion

3.1 Data Loading and Manipulation Impact on Training Efficiency and Speed

During initial training of the model we encountered limiting factors in the form of computational bottlenecks unrelated to deep learning model computations. These were mostly based on data loading procedures. We started with a simple data-loader that extracted reads straight from the basic fastq files generated during the sequencing experiment. This approach relayed on the basic file

handling operations used in python and was too slow for the platform to allow efficient training of deep learning models. After converting the data files into compressed binary files there was a significant increase in speed of the training. However, this occurred initially, while after a few batches the training halted. Our troubleshooting indicated the problem occurred due to overflowing of the operating system buffer and cache memory due to the default python method for reading parts of binary files. To overcome this issue we integrated another change to the platform to pull out large groups of reads at once instead of getting reads one by one. This aspect of deep learning is often overlooked since most of the research is performed on data that is either in a format already accessible by the default data-loaders or could be loaded easily into the memory. As deep learning is adopted in more fields, there will be an increasing need for custom data-loaders that are able to handle large amounts of data and perhaps perform manipulations required by the specific challenge. These might be highly efficient extractions of randomized group of reads, as seen in the project. Our study here presents one approach to solve such issues.

3.2 Results of Model Performance on Microbiome Dataset Sample Classification

Training the model to classify samples into healthy samples or samples originating from a cancer patient yielded a model outperforming the classifier proposed by the original study (these results come with a major caveat mentioned in Subheading 3.3). The classifier based on the metagenomic composition of the sample built by the originals authors achieved an AUC of 0.84 and if incorporating another clinical test (FOBT), they achieved an AUC of 0.87. The deep learning classifier trained on the raw sequencing output from the experiments achieved an AUC of 0.93 with the receiver operating characteristic curve (ROC), surpassing the results from the original study, and both FOBT and Wif-1 methylation tests (as seen in Fig. 3). Although the means by which the model performed the classification and surpassed the results is unknown, this could be explained by several possibilities; perhaps the deep learning model could learn to deduce the metagenomic composition from the raw sequencing reads and since the deep learning model could better model the interactions between different reads (and consequently between different bacteria) in the batch it could derive more accurate predictions. Another possible explanation is that the model could extract information about the genes represented by those sequences. Thus it could acquire information about the entire gene pool present in the samples. This could be useful since the original study showed different gene pool compositions in patients with cancer, which could be caused by tumor–host interactions. A final route for the model to perform the prediction is to use the information present in the sequences besides those originating from the gut microbiota given that around 50% of the reads did not map to the bacterial database.

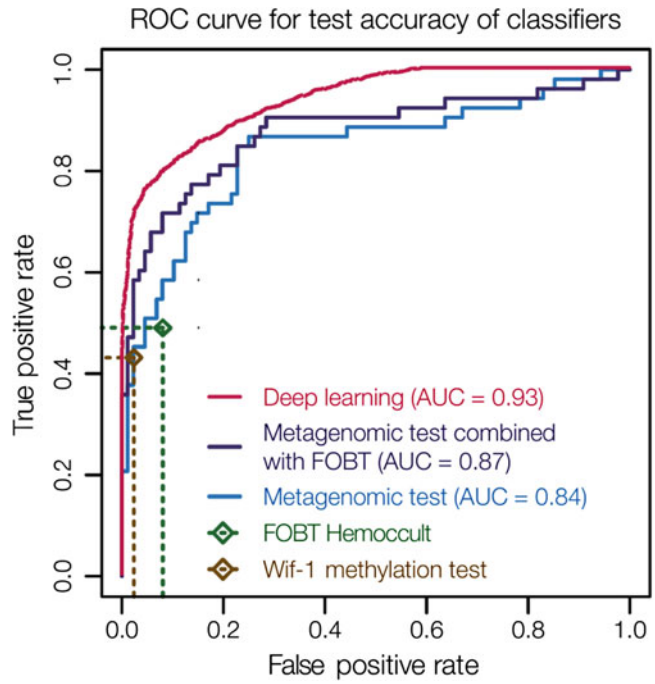


Fig. 3 Our deep learning model outperforms experimental clinical tests FOBT and Wif-1 and yield better AUC compared to the previous classifiers [20]. *See* comments in the text. With permission from Professor Peer Bork

We hope that the deep learning model combines all of these methods and even utilizes information still unexplored by traditional bioinformatic techniques. These could eventually be discovered by analyzing the model and is expected to lead to novel biological and medical discoveries.

3.3 Deteriorating Results for Population from Different Countries

The study collected samples from several clinical centers in France and Germany. Since the primary group of samples was collected in France, we used these samples to develop our model—the separation of samples into train and test groups involved samples exclusively from France. After developing the model and acquiring good results on the test samples of French origin we moved our attention to the German samples. Unfortunately, the performance on the samples of German origin plummeted to an accuracy of almost 50%. This important test shows that although deep learning can show impressive results on some data it is crucial to verify them on data from a different distribution/source. The machine learning classifiers generated by the researchers, which are based on metagenomic composition, showed better generalization performance with minimal change in accuracy after training on one population and testing on another. This makes their classifier much more suitable for real world applications compared to the deep learning

model in its current state. The difference in performance of the deep learning model indicates that the method by which the model performs the prediction is not entirely based on deduction of metagenomic composition from the raw reads, otherwise the model would perform similarly on different populations. The deterioration in performance could be explained by a possible technical batch effect present in the samples from France but not present in samples from Germany. Another reason could be unrelated to sample collection or preparation procedures but related to different patient treatment or patient behavior that could affect the microbiome environment, such as different diet, medication, and clinical procedures, between France and Germany. The solution to these issues could be standardization of patient treatment and sample collection procedures across different clinical centers participating in the study and between healthy and sick individuals. This might minimize any possible technical batch effect in the results. If the issue is caused by intrinsic differences between populations then training the model on several populations together could produce a model with better generalization abilities. Alternatively, training the model for each sub-population separately could mitigate the issue as well. In any case, advanced understanding of the model and its decision-making process is crucial to develop superior models and deploying them to real world scenarios.

3.4 The Advantages and Limitations of the DL Models

Deep learning promises great potential to becoming a standard tool alongside traditional bioinformatic workflows. In some cases it could even replace them. While traditional bioinformatic algorithms work on specific portions of information separately (mutation analysis, structural variants, methylation, metagenomic, etc.), deep learning can learn the hidden representations from large datasets and make connection between different parts of those representations. Thus, it could lead to more accurate decisions based on the composition of all data available and not only a specific aspect of it. Additionally, when more advanced deep learning models will be developed to work with genomic data, these could be pretrained on the huge amount of freely available genomic data and later adopted for certain tasks in the field with much less training such as what today is done with large NLP models. Another crucial advantage of deep learning presented in our study is the ability to analyze raw data from experiments without any preprocessing required by traditional bioinformatic workflows. This could mean that researchers could collect genetic data from an experiment and could potentially use our deep learning platform to test its ability to solve sequencing related scientific queries. Alternatively, it could pinpoint to other interesting insights within the medical data.

There are several limitations to applying deep learning on genomics. The training requires large amount of data, which is not always available in biological experiments. As we demonstrated

in our results, the models could perform well for some cases but not for others. Unfortunately, there is no clear indication for when the model will fail nor an explanation for the reason of failure. Compared to more traditional approaches, as in the original study, the authors could analyze the classifier and check the contribution of each bacterial genome to the decision-making process. This allows understanding of when and why it failed, which was carried out by the authors and why they achieved the optimal performing classifier by specifically selecting 22 bacterial species. This leads to the final limitation of deep learning that is critical in the medical arena—the lack of explainability. Traditional bioinformatic analysis is structured in a way that the researchers could pinpoint to a specific gene/region/bacteria causing the phenotype, while in deep learning (especially in our case where we use the raw reads from sequencing) it is very difficult to understand how and why the model reached the decision.

3.5 Future Directions and Exploring Explainability

The study presented here is work-in-progress and should be treated as such. Various improvements and changes are being implemented to eliminate the shortcomings of the platform at its current state. The main priority for further research is incorporating more complex models than the simple RNNs implementation right now. Specifically, the latest methods in NLP such as BERT [31], GPT-2 [32], and ULMFiT [33]. The larger and more complex models should learn better representations of the genetic information and hopefully provide better predictions utilizing more of the hidden information from the raw data. As previously mentioned, explainability of the model is another critical aspect of the current model that should be explored before the model could be effectively utilized in clinical settings. Explainability is an active research topic in the deep learning field and many studies aim to understand the decision-making process of different models. For example, in one study researchers used Grad-CAM methods to highlight parts of the image that was important to make certain classification of the image. Thus, in images classified as cat, the part of the image containing the cat was highlighted. We are working on how to try to tackle the problem of explainability of the model shown here. We are working to prove that in the issue of cancer diagnosis, our model gives higher importance to reads originating from cancer related genes than other genes in the human genome.

3.6 Code Availability

The code will be available upon approval of Tel Aviv University on the github repository. https://github.com/nshomron/DeepLearning_NGS

References

1. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521:436–444. https://doi.org/10.1038/nature14539
2. Berners-Lee CM (1968) Cybernetics and forecasting. Nature 219:202–203. https://doi.org/10.1038/219202b0
3. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9:1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735
4. Lecun Y, Bottou L, Bengio Y et al (1998) Gradient-based learning applied to document recognition. Proc IEEE 86:2278–2324. https://doi.org/10.1109/5.726791
5. Krizhevsky A, Sutskever I, Hinton GE (2012) ImageNet classification with deep convolutional neural networks. In: Pereira F, Burges CJC, Bottou L et al (eds) Advances in neural information processing systems 25. Curran Associates, Red Hook, pp 1097–1105
6. van den Oord A, Dieleman S, Zen H et al (2016) WaveNet: a generative model for raw audio. ArXiv160903499Cs
7. Russakovsky O, Deng J, Su H et al (2015) ImageNet large scale visual recognition challenge. Int J Comput Vis 115:211–252. https://doi.org/10.1007/s11263-015-0816-y
8. Conneau A, Schwenk H, Barrault L et al (2016) Very deep convolutional networks for text classification. ArXiv160601781 Cs
9. Deng L, Yu D (2014) Deep learning: methods and applications. Found Trends® Signal Process 7:197–387. https://doi.org/10.1561/2000000039
10. Litjens G, Kooi T, Bejnordi BE et al (2017) A survey on deep learning in medical image analysis. Med Image Anal 42:60–88. https://doi.org/10.1016/j.media.2017.07.005
11. He K, Zhang X, Ren S et al (2015) Deep residual learning for image recognition. ArXiv151203385 Cs
12. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. ArXiv14091556 Cs
13. Yosinski J, Clune J, Bengio Y et al (2014) How transferable are features in deep neural networks? In: Ghahramani Z, Welling M, Cortes C et al (eds) Advances in neural information processing systems 27. Curran Associates, Red Hook, pp 3320–3328
14. Puch S, Sánchez I, Rowe M (2019) Few-shot learning with deep triplet networks for brain imaging modality recognition. ArXiv190810266 Cs Stat
15. Zhao A, Balakrishnan G, Durand F et al (2019) Data augmentation using learned transforms for one-shot medical image segmentation
16. Rajkomar A, Oren E, Chen K et al (2018) Scalable and accurate deep learning with electronic health records. Npj Digit Med 1:1–10. https://doi.org/10.1038/s41746-018-0029-1
17. Xiao C, Choi E, Sun J (2018) Opportunities and challenges in developing deep learning models using electronic health records data: a systematic review. J Am Med Inform Assoc 25:1419–1428. https://doi.org/10.1093/jamia/ocy068
18. Yue T, Wang H (2018) Deep learning for genomics: a concise overview. ArXiv180200810 Cs Q-Bio
19. SRA database growth. https://www.ncbi.nlm.nih.gov/sra/docs/sragrowth/. Accessed 29 Dec 2019
20. Zeller G, Tap J, Voigt AY et al (2014) Potential of fecal microbiota for early-stage detection of colorectal cancer. Mol Syst Biol 10:766. https://doi.org/10.15252/msb.20145645
21. Poplin R, Chang P-C, Alexander D et al (2018) A universal SNP and small-indel variant caller using deep neural networks. Nat Biotechnol 36:983–987. https://doi.org/10.1038/nbt.4235
22. Kothen-Hill ST, Zviran A, Schulman RC et al (2018) Deep learning mutation prediction enables early stage lung cancer detection in liquid biopsy
23. Tampuu A, Bzhalava Z, Dillner J et al (2019) ViraMiner: deep learning on raw DNA sequences for identifying viral genomes in human samples. PLoS One 14:e0222271. https://doi.org/10.1371/journal.pone.0222271
24. Eser U, Churchman LS (2016) FIDDLE: an integrative deep learning framework for functional genomic data inference. bioRxiv 081380. https://doi.org/10.1101/081380
25. Busia A, Dahl GE, Fannjiang C et al (2019) A deep learning approach to pattern recognition for short DNA sequences. bioRxiv 353474. https://doi.org/10.1101/353474

26. Liu F, Li H, Ren C et al (2016) PEDLA: predicting enhancers with a deep learning-based algorithmic framework. Sci Rep 6:1–14. https://doi.org/10.1038/srep28517
27. Zeng W, Wu M, Jiang R (2018) Prediction of enhancer-promoter interactions via natural language processing. BMC Genomics 19:84. https://doi.org/10.1186/s12864-018-4459-6
28. Li H, Handsaker B, Wysoker A et al (2009) The Sequence Alignment/Map format and SAMtools. Bioinformatics 25:2078–2079. https://doi.org/10.1093/bioinformatics/btp352
29. (2020) pysam-developers/pysam. pysam-developers
30. Bradbury J, Merity S, Xiong C et al (2016) Quasi-recurrent neural networks. ArXiv161101576 Cs
31. Devlin J, Chang M-W, Lee K et al (2019) BERT: pre-training of deep bidirectional transformers for language understanding. ArXiv181004805 Cs
32. Radford A, Wu J, Child R et al (2019) Language models are unsupervised multitask learners. OpenAI Blog 1:9
33. Howard J, Ruder S (2018) Universal language model fine-tuning for text classification. ArXiv180106146 Cs Stat

Chapter 10

Interrogating the Accessible Chromatin Landscape of Eukaryote Genomes Using ATAC-seq

Georgi K. Marinov and Zohar Shipony

Abstract

The ATAC-seq assay has emerged as the most useful, versatile, and widely adaptable method for profiling accessible chromatin regions and tracking the activity of *cis*-regulatory elements (cREs) in eukaryotes. Thanks to its great utility, it is now being applied to map active chromatin in the context of a very wide diversity of biological systems and questions. In the course of these studies, considerable experience working with ATAC-seq data has accumulated and a standard set of computational tasks that need to be carried for most ATAC-seq analyses has emerged. Here, we review and provide examples of common such analytical procedures (including data processing, quality control, peak calling, identifying differentially accessible open chromatin regions, and variable transcription factor (TF) motif accessibility) and discuss recommended optimal practices.

Key words Regulatory elements, Transcription factors, Chromatin accessibility, ATAC-seq, High-throughput sequencing

1 Introduction

In most eukaryotic cells, the genome is packaged by nucleosomal octamer particles comprised of the four core nucleosomal histones H3, H4, H2A, and H2B. Nucleosomes have a refractory effect to transcription and to the binding to DNA by most regulatory proteins. Thus, active *cis*-regulatory elements in eukaryotes tend to be depleted of nucleosomes. This is a highly useful property as it allows for active cREs to be specifically labeled and mapped in a variety of ways, as first recognized decades ago when the hypersensitivity to cleavage by DNase enzymes of enhancer and promoter elements was initially reported [1–3]. DNase hypersensitivity continued to be the primary way of mapping cREs into the genomic era, first, by coupling it to microarrays [4–6], and later to high-throughput massively parallel sequencing [7–9]. Numerous alternative and complementary methods have been also developed in recent years, based on the preferential enzymatic/chemical cleavage/

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_10,

modification of accessible DNA. In order to map open chromatin regions in the genome, these assays employ methyltransferases [10–14], restriction enzymes [15], nicking enzymes [16], small molecules [17], viral integration [18], and the preferential insertion into unprotected DNA by transposomes [19].

The latter approach, in the form of the ATAC-seq assay [19], has emerged as the most convenient, versatile, and widely used method for studying the chromatin state of the eukaryotic cell. In an ATAC-seq reaction (Fig. 1), isolated nuclei are subjected to

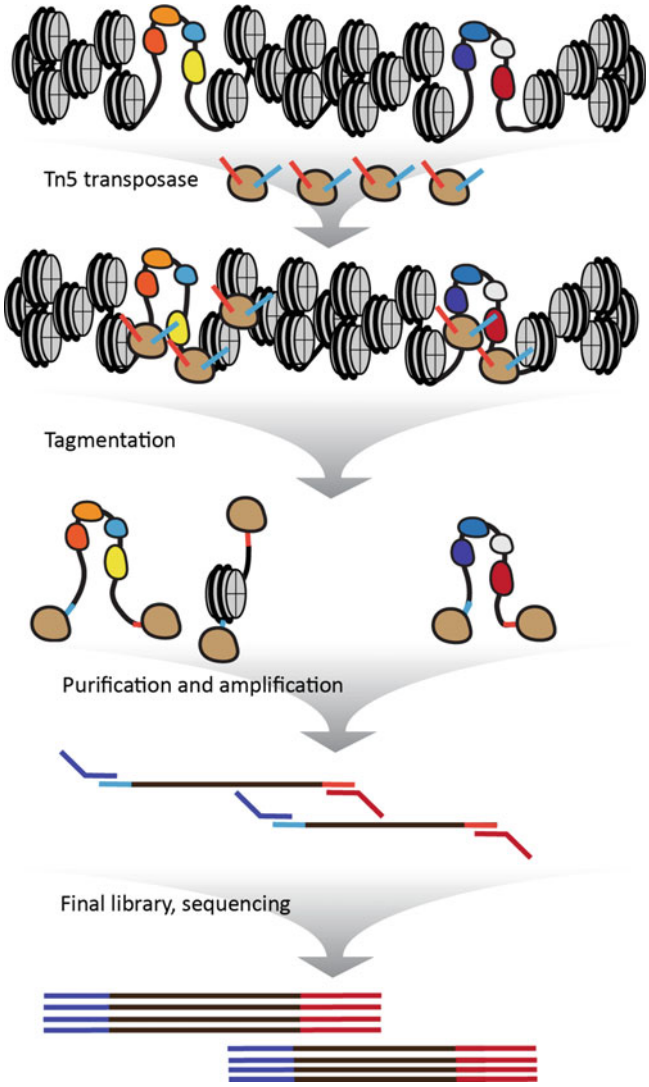


Fig. 1 Overview of the ATAC-seq experimental protocol. Chromatin is subjected to incubation with an active Tn5 transposase carrying adapter sequences that can be directly used for PCR amplification. The transposase preferentially inserts into accessible regions in the genome, such as active cREs. DNA is then purified and PCR amplification is carried out from the adapter sequences deposited by Tn5

treatment with a Tn5 transposase enzyme carrying DNA adapters that are inserted into DNA where DNA is accessible. These adapters then enable the direct amplification of open chromatin fragments, eliminating most of the complex intermediate enzymatic conversion steps that were part of previous approaches such as DNase-seq. This allows for the whole protocol to be completed in just a few hours. It also dramatically lowers the input requirements (50,000 mammalian cells are typically used for an ATAC reaction), including down to single-cell level [20, 21].

Due to these advantages, ATAC-seq has become the method of choice for profiling open chromatin. In the process, a standard set of processing, quality assessment, and downstream analyses practices has begun to emerge (Fig. 2). In this chapter, we review

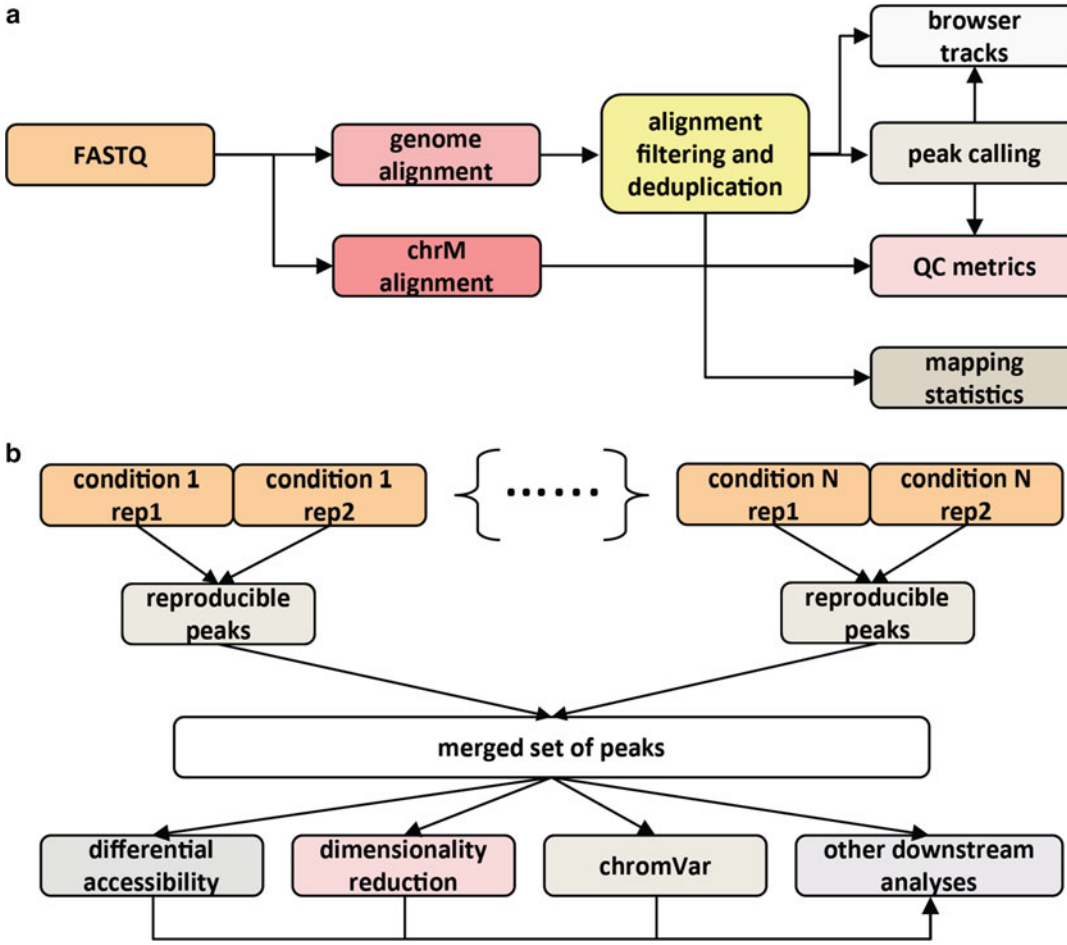


Fig. 2 Overview of a general ATAC-seq analysis workflow. (**a**) Individual samples are aligned both against the nuclear genome and also against the mitochondrial genome (the latter is for quality control purposes as described in the main text), alignments are filtered, and peak calls, browser tracks and mapping statistics and quality metrics are compiled. (**b**) For multiple samples and replicates in a study, reproducible peaks are identified, then combined to derive a unified merged set of peaks. This set of peaks is used to carry out most downstream analyses, including differential accessibility, dimensionality reduction, variable motif accessibility, and others

the optimal approaches to carrying out these tasks, and illustrate their application using publicly available ATAC-seq datasets from the ENCODE Project Consortium [22] as examples.

2 Materials

The analyses described are designed to run on standard Linux systems through the UNIX command line. The maximal memory usage depends on the size of the datasets but is usually less than 15GB.

2.1 Genomic Sequence and Annotation Files

1. A FASTA file containing the GRCh38 version of the human genome can be downloaded from the UCSC Genome Browser at http://hgdownload.soe.ucsc.edu/goldenPath/hg38/bigZips/hg38.fa.gz. Genome files can also be obtained from ENSEMBL (http://ensemblgenomes.org/) and from the NCBI website (http://www.ncbi.nlm.nih.gov/assembly/). However, it has to be noted that in the case of the human genome, reference FASTA files available in public repositories contain alternative haplotype contigs, i.e. alternative versions of sequences already present in the assembly. The inclusion of such sequences means that their version in the main chromosomes loses its unique mappability with short reads, and becomes effectively "invisible" to downstream analysis. This is, in almost all cases, an undesirable behavior, thus alternative haplotypes should be removed from reference files before use. The ENCODE Project [22] provides such filtered files from its portal at https://www.encodeproject.org/data-standards/reference-sequences/. For the purposes of ATAC-seq processing, a separate fasta file containing only the mitochondrial genome is also needed; this sequence can be extracted from the whole-genome FASTA file.
2. The same page on the ENCODE Portal also provides "blacklist" regions [23], i.e. locations in the genome that are artifactually enriched in sequencing assays and should be filtered out from peak call sets (discussed further below).
3. Genome annotations in GTF format can be obtained from UCSC, ENSEMBL, NCBI, or ENCODE. For the purposes of ATAC-seq analysis, we prefer to work with the RefSeq annotation, which can be obtained from https://www.ncbi.nlm.nih.gov/projects/genome/guide/human/index.shtml. See discussion in the relevant section below for more details.

2.2 Software Packages

1. `Bowtie` [24] (http://bowtie-bio.sourceforge.net/index.shtml) or `Bowtie2` [25] (http://bowtie-bio.sourceforge.net/bowtie2/index.shtml). Also see the discussion on alignment below for more details.

2. `samtools` [26]: http://www.htslib.org/
3. `PicardTools` https://broadinstitute.github.io/picard/
4. `MACS 2.1.0` [27]: https://github.com/taoliu/MACS/
5. IDR [28] analysis code (version 2.0.4): https://github.com/kundajelab/idr
6. UCSC Genome Browser [29, 30] utilities: http://hgdownload.cse.ucsc.edu/admin/exe/
7. R: https://www.r-project.org/
8. Python (version 2.7 or higher) https://www.python.org/
9. UMAP https://umap-learn.readthedocs.io/en/latest/
10. DESeq2 [31] https://bioconductor.org/packages/release/bioc/html/DESeq2.html
11. chromVAR [32]: https://github.com/GreenleafLab/chromVAR
12. `deepTools` [33]: https://deeptools.readthedocs.io/en/develop/index.html
13. `BEDtools` [34]: https://bedtools.readthedocs.io/en/latest/#
14. featureCounts [35] (from the Subread package): http://subread.sourceforge.net/
15. JASPAR2018: https://bioconductor.org/packages/release/data/annotation/html/JASPAR2018.html
16. pheatmap: https://cran.r-project.org/web/packages/pheatmap/index.html
17. TFBSTools: https://bioconductor.org/packages/release/bioc/html/TFBSTools.html
18. BSgenome.Hsapiens.UCSC.hg38: https://bioconductor.org/packages/release/data/annotation/html/BSgenome.Hsapiens.UCSC.hg38.html
19. Additional scripts: https://github.com/georgimarinov/GeorgiScripts. Contains python scripts used in the examples shown below; some of the scripts depend on having `pysam` (https://pysam.readthedocs.io/en/latest/index.html) and `pyBigWig` (https://github.com/deeptools/pyBigWig) installed.
20. TGL Kmeans: https://github.com/tanaylab/tglkmeans

3 Methods

The typical ATAC-seq analysis procedure is outlined in Fig. 2, and can be split in two parts—first, processing, evaluation and analysis at the level of individual samples, then followed by integrative analysis of multiple samples.

Individual sample processing consists of the following steps:

1. Aligning reads against the whole genome.
2. Aligning reads against the mitochondrial genome alone.
3. Filtering poor quality and multireads alignments.
4. Deduplication of alignments.
5. Generation of genome browser tracks.
6. Calculation of mapping statistics and other quality control metrics.
7. Per-replicate/sample peak calling.

Typical multisample analysis tasks include:

1. IDR
2. Merging peaks
3. Dimensionality reduction and data exploration
4. Identifying clusters of accessible regions that behave similarly across samples
5. Identifying regions that are differentially accessible between conditions
6. Analyzing variable motif occupancy

In addition, one might also be interested in examining ATAC-seq footprints around transcription factor binding sites and nucleosome occupancy around particular genomic landmarks.

Procedures and considerations for carrying out these task are described in the subsequent sections.

3.1 Preparation of Genomic Files

1. Download and uncompress genome reference files:

```
wget https://www.encodeproject.org/files/
GRCh38_no_alt_analysis_set_GCA_000001405.15/
@@download/GRCh38_no_alt_analysis_set_GCA_000001405.15.fasta.gz
 -O hg38_no_alt.fasta.gz

 gunzip hg38_no_alt.fasta.gz
```

2. Create a `bowtie` genome index file:

```
mkdir genomes/hg38/bowtie-indexes
cd genomes/hg38/bowtie-indexes
ln ../hg38_no_alt.fa
bowtie-build -f hg38_no_alt.fa hg38_no_alt
```

With `Bowtie2`:

```
mkdir genomes/hg38/bowtie2-indexes
cd genomes/hg38/bowtie2-indexes
ln ../hg38_no_alt.fa
bowtie2-build -f hg38_no_alt.fa hg38_no_alt
```

3. Create a `bowtie` mitogenome ("chrM") index file using only the mitochondrial genome as input:

```
mkdir genomes/hg38/bowtie-indexes
cd genomes/hg38/bowtie-indexes
ln ../chrM.fa
bowtie-build -f chrM.fa chrM
```

With `Bowtie2`:

```
mkdir genomes/hg38/bowtie2-indexes
cd genomes/hg38/bowtie2-indexes
ln ../chrM.fa
bowtie2-build -f chrM.fa chrM
```

4. Create chromosome size info (`chrom.sizes`) files:

```
python makeChromSizesFromFasta.py
              hg38_no_alt.fa hg38_no_alt.chrom.sizes
```

Chromosome size files consist of one line per chromosome in the following format:

```
chr <tab> chromosome_size
```

They identify the end points of chromosomes/contigs and are used at multiple steps in the processing of high-throughput sequencing data.

3.2 Read Mapping

The currently most widely used sequencing platforms are the Illumina NextSeq, HiSeq, MiSeq, and NovaSeq instruments. Sequencing kits sufficient for 75-, 100-, 150-, 300-, and 600-cycle runs are available for these machines in various configurations. In the case of ATAC-seq data, it is strongly recommended that sequencing runs are carried in a paired-end format, first, because the information provided by having two insertion sites per fragment rather than one is helpful for a number of analyses (such as TF footprinting), and second, because fragment distribution is a standard part of the quality assessment of ATAC-seq libraries.

Another point to consider is that while it is often common to see ATAC-seq libraries sequenced as 2×75mers or longer (driven either by the thinking that longer sequences provide better

mapping of short reads, by logistic constraints at sequencing facilities, or by other factors), this is in fact not necessary and only increases the cost of sequencing (by at least twofold). This is because the fragment length distribution of ATAC-seq libraries usually peaks at around 45–50 bp (Fig. 5), and as a result, a majority of fragments are already fully covered by 2×36mer reads.

For these reasons, we carry out all our ATAC-seq sequencing runs as 2×36mers, and we also analyze all ATAC-seq datasets as 2×36mers even if they were sequenced as 2×75mers (or longer), in order to maintain uniformity across all datasets we work with (as longer reads do indeed map uniquely more often than shorter reads when fragments originate from areas of the genome that are not uniquely mappable, the possibility exists for mappability and alignment bias in some samples to generate misleading results if read length is not uniform).

However, under certain circumstances (e.g. when examining allele-biased accessibility or the effect of sequence variants on accessibility) it can be beneficial to use longer reads and to use the full length of fragments. In those cases, reads need to be trimmed of adapters, which can be done using the Trimmomatic [36] or TrimGalore/Cutadapt [37] programs.

Read mapping can in principle be carried out with any of the numerous short read aligners developed over the years, with equivalent results, but two of them—Bowtie2 [25] and BWA [38]—have emerged as the standard tools for carrying this task. In our practice we use primarily Bowtie, as follows.

1. Trim reads (both ends) to 36mers:

```
zcat SAMPLE.end1.fastq.gz | python trimfastq.py - 36 -stdout |
     gzip > SAMPLE.end1.36mers.fastq.gz
zcat SAMPLE.end2.fastq.gz | python trimfastq.py - 36 -stdout |
     gzip > SAMPLE.end2.36mers.fastq.gz
```

2. Map 2×36mer reads to whole genome.

   With Bowtie:

```
python PEFastqToTabDelimited.py
  SAMPLE.end1.36mers.fastq.gz SAMPLE.end2.36mers.fastq.gz |
  bowtie hg38/bowtie-indexes/hg38_no_alt -p 16 -v 2 -k 2 -m 1
  -t --best --strata -q --sam-nh -X 1000 --sam --12 - |
  samtools view -F 4 -bT hg38/sequence/hg38_no_alt.fa - |
  samtools sort - SAMPLE.2x36mers.unique
```

   This retains uniquely mapping read pairs with up to 2 mismatches relative to the reference.

   With Bowtie2:

```
bowtie2 -x hg38/bowtie2-indexes/hg38_no_alt
  -1 SAMPLE.end1.36mers.fastq.gz -2 end2.fastq.gz -p 16 -t
  -X 1000 --no-mixed --no-discordant -
  | samtools -F 1804  view -bT hg38/sequence/hg38_no_alt.fa - |
    samtools sort - SAMPLE.2x36mers.unique
```

This command also filters out all alignments with poor quality and non-unique alignments.

Alignments are stored in the BAM format (a binary version of the SAM format [26]) for all subsequent analyses.

3. Map 2×36mer reads to the mitochondrial genome.

   This step is necessary for the proper counting of mitochondrial reads.

   As ATAC-seq relies on the preferential insertion of Tn5 into accessible DNA, the mitochondrial genome tends to be extremely strongly enriched in ATAC-seq libraries. This is in part because there are hundreds to thousands of copies of it in each mammalian cells, but primarily because it is not packaged by nucleosomes and is thus highly susceptible to transposase insertion. As a result, in early versions of the ATAC-seq protocol mitochondrial reads often constituted the majority of the library. The assay has subsequently been optimized to greatly reduce mitochondrial contamination [39], but estimating the chrM-mapping reads is still a core part of the quality assessment of ATAC libraries.

   The simplest approach to estimating mitochondrial contamination is to calculate the number of alignments mapping to chrM. However, this underestimates the actual number of such reads, and does so to an extent that greatly varies between species and even different assemblies of the same species. This is because of the presence of the so-called NUMTs (**NU**clear **M**i**T**ochondrial sequences) in nuclear genomes due to the still ongoing process of endosymbiotic gene transfer (EGT), in which DNA from the mitochondrion (or from other endosymbionts in eukaryotic cells) is inserted into the nuclear genome [40]. Very recent NUMT insertions have essentially the same sequence as the mitochondrial genome, and as a result make the homologous regions of chrM not uniquely mappable. Depending on the exact content of an assembly, this effect can affect from a minor fraction to almost the entire mitochondrial genome. Examination of chrM unique mappability in different species shows, for example, that up to half of the mouse and nearly the whole *Drosophila melanogaster* mitochondrial genomes are not uniquely mappable with short reads [41].

   For these reasons, if mitomapping reads are to be accurately counted, it is optimal to carry out a separate alignment to

the mitochondrial genome alone, as the great majority of reads mapping to it are expected to derive from the mitochondrion and not from NUMTs (which are chromatinized and individually at most diploid in copy number, compared to the thousands of copies of the mitochondrial genome in the cell). In addition, because there can be sequences that are not uniquely mappable even within the mitochondrial genome itself (this is not the case with mammalian mitogenomes, but does happen quite frequently in the organellar genomes of other lineages), this alignment step can be carried out allowing for multimapping reads.

With Bowtie1:

```
python PEFastqToTabDelimited.py
  SAMPLE.end1.36mers.fastq.gz SAMPLE.end2.36mers.fastq.gz |
  bowtie hg38/bowtie-indexes/chrM -p 16 -v 2 -a
  -t --best --strata -q --sam-nh -X 1000 --sam --12 - |
  samtools view -F 4 -bT hg38/sequence/hg38_no_alt.fa - |
  samtools sort - SAMPLE.2x36mers.chrM
```

With Bowtie2:

```
bowtie2 -x hg38/bowtie2-indexes/chrM
  -1 SAMPLE.end1.36mers.fastq.gz -2 end2.fastq.gz -p 16 -t
  -X 1000 --no-mixed --no-discordant -
  | samtools -F 1804  view -bT hg38/sequence/hg38_no_alt.fa - |
    samtools sort - SAMPLE.2x36mers.chrM
```

4. Index bam files with `samtools`:

```
samtools index SAMPLE.2x36mers.unique.bam
samtools index SAMPLE.2x36mers.chrM.bam
```

3.3 Filtering and Deduplicating Alignments

As it is the nuclear genome that is typically of interest in an ATAC-seq dataset, reads mapping to the mitochondrion represent a confounding factor, as they affect the total library size and normalization factors if retained for downstream analyses. For these reasons, mitochondrial reads are filtered out of BAM files after the initial alignment step, as follows:

1. Filter out mitochondrial reads.

```
samtools view SAMPLE.2x36mers.unique.bam | egrep -v chrM |
samtools view -bT hg38/sequence/hg38_no_alt.fa - -o
SAMPLE.2x36mers.unique.nochrM.bam
```

Note that, depending on the species one is working with, the mitochondrial chromosome need not be named "chrM," need not be a single chromosome (multipartite mitochondrial genomes are found in numerous species [42]), and need not be the only organellar genome to be filtered out (for example, photosynthesizing eukaryotes also have plastids). Change the filtering command accordingly, if necessary.

2. Index the resulting BAM file:

```
samtools index SAMPLE.2x36mers.unique.nochrM.bam
```

3. Remove duplicate alignments.

   As ATAC-seq is typically performed on a very small number of cells (the equivalent of 50,000 mammalian cells), meaning that a limited initial population of original molecules is used as a starting point for library construction, and because it is sequenced in a paired-end format, fragments with exactly the same coordinates are more likely than not to represent PCR duplicates rather than different original fragments. Thus it is a standard step in ATAC-seq processing to remove duplicate fragments. This is carried out using the `MarkDuplicates` program in the `PicardTools` suite, as follows:

```
module load java; java -Xmx4G -jar
picard-tools-1.99/MarkDuplicates.jar
INPUT=SAMPLE.2x36mers.unique.nochrM.bam
OUTPUT=SAMPLE.2x36mers.unique.nochrM.dedup.bam
METRICS_FILE=SAMPLE.2x36mers.unique.nochrM.dedup.metrics
VALIDATION_STRINGENCY=LENIENT
ASSUME_SORTED=true REMOVE_DUPLICATES=true
```

4. Index the resulting BAM file:

```
samtools index SAMPLE.2x36mers.unique.nochrM.dedup.bam
```

3.4 Genome Browser Track Generation

The next step in the processing is to generate genome-wide coverage tracks. Two types of tracks can be generated, a "coverage" track that adds to the score of each base that a mapped fragment covers, and "5′" tracks, which only represent the end points (or "cute sites") of fragments. While the latter type of tracks is also used in the analysis of DNAse-seq, ChIP-exo, and other sequencing-based functional genomic assays, there is a small caveat when working with ATAC-seq datasets—as the transposase itself has a footprint of about 9 bp [19], fragment ends are shifted by 5 bp or 4 bp (depending on which strand they map to) in order to more accurately represent the actual "cut site."

For the purpose of allowing comparison between different samples, it is optimal to normalize the tracks relative to the total set of mapped and dedupped reads, e.g. in RPM (Reads Per Million mapped reads) units.

1. Generate RPM-normalized coverage tracks (using the `bamCoverage` program in `deepTools`):

```
bamCoverage --bam SAMPLE.2x36mers.unique.nochrM.dedup.bam
-o SAMPLE.2x36mers.unique.nochrM.dedup.coverage.bigWig
--binSize 100 --normalizeUsingRPKM --extendReads
--numberOfProcessors {threads}
```

2. Generate RPM-normalized "5′" tracks (using the `alignmentSieve` and `bamCoverage` programs in `deepTools`):

```
alignmentSieve --numberOfProcessors {threads}
--ATACshift --bam SAMPLE.2x36mers.unique.nochrM.dedup.bam
-o tmp.bam

samtools sort -O bam -o
SAMPLE.2x36mers.unique.nochrM.dedup.shifted.bam tmp.bam

samtools index SAMPLE.2x36mers.unique.nochrM.dedup.shifted.bam

bamCoverage --bam
SAMPLE.2x36mers.unique.nochrM.dedup.shifted.bam
-o SAMPLE.2x36mers.unique.nochrM.dedup.shifted.coverage.bigWig
--binSize 100 --normalizeUsingRPKM --extendReads
--numberOfProcessors {threads}

rm tmp.bam
```

Examples of coverage and 5′ tracks are shown for the *IVL* locus in the context of keratinocyte differentiation in Fig. 3.

3.5 Mapping Statistics and ATAC-seq Quality Assessment

How well the experiment worked and whether its properties could negatively affect data analyses and interpretation are critical questions about every high-throughput sequencing library. Quality control (QC) evaluation is therefore an essential step of processing pipelines. Some of these metrics are common across most functional genomic assays, e.g. estimating the molecular complexity of a library (generally, the fewer original molecules are represented in the final library, the worse the dataset is) and calculating read mapping statistics, while others are specific to the nature of the data type at hand.

In addition to these more general QC statistics, several assay-specific metrics are employed when working with ATAC-seq data.

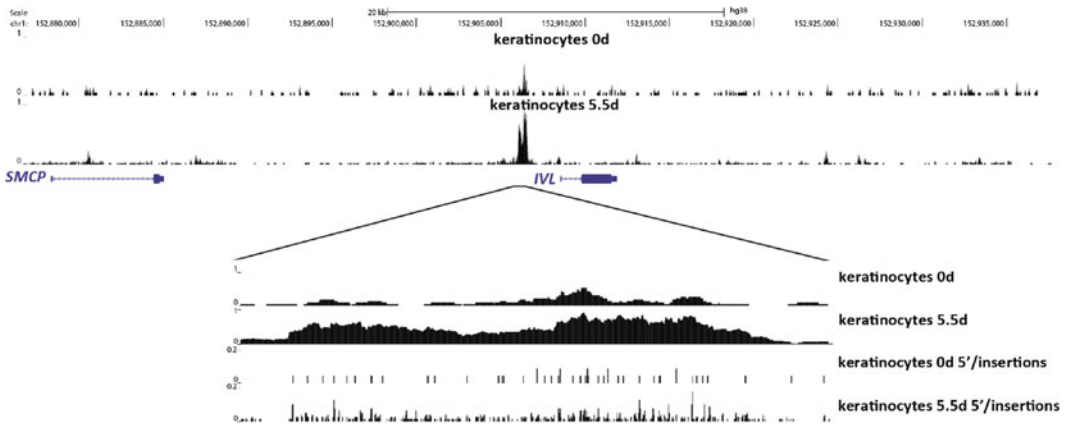


Fig. 3 Example of ATAC-seq coverage and 5′ tracks. Shown are UCSC browser snapshots displaying the changes in chromatin accessibility during keratinocyte differentiation around the *IVL* (involucrin) gene. The lower panel shows both coverage and Tn5 insertion/fragment 5′ tracks

These include the fragment length distribution, the fraction of mitochondrial reads, and transcription start site (TSS) enrichment. The typical goals of ATAC-seq QC are to evaluate the extent of mitochondrial contamination, the fragment length distribution and the molecular complexity of libraries, and, most importantly, how strongly enriched for open chromatin regions an ATAC library is.

1. Count raw reads:

```
zcat SAMPLE.fastq.gz | wc -l
```

   Divide by 4 to get the number of reads (as each read is represented by 4 lines in a FASTQ file).

2. Calculate mapping statistics for the unfiltered BAM file:

```
python SAMstats.py SAMPLE.2x36mers.unique.bam
SAMstats-SAMPLE.2x36mers.unique -bam hg38.chrom.sizes
samtools -paired
```

3. Calculate mapping statistics for the chrM-mapping BAM file:

```
python SAMstats.py SAMPLE.2x36mers.chrM.bam
SAMstats-SAMPLE.2x36mers.chrM -bam hg38.chrom.sizes
samtools -paired
```

   Record the total number of reads mapping to the mitochondrion ($|R_M|$).

4. Calculate mapping statistics for the chrM-filtered pre-deduplication BAM file:

```
python SAMstats.py SAMPLE.2x36mers.unique.nochrM.bam
SAMstats-SAMPLE.2x36mers.unique.nochrM
-bam hg38.chrom.sizes samtools -paired
```

   Record the total number of reads mapping to the nuclear genome ($|R_N|$).

5. Calculate mapping statistics for the chrM-filtered post-deduplication BAM file:

```
python SAMstats.py SAMPLE.2x36mers.unique.nochrM.dedup.bam
SAMstats-SAMPLE.2x36mers.unique.nochrM.dedup
-bam hg38.chrom.sizes samtools -paired
```

6. Estimate library complexity and total library size.

   Several simple metrics have been used in the literature to characterize the apparent molecular complexity of sequencing libraries [43], such as the Non-Redundant read Fraction *NRF* and the PCR Bottlenecking Coefficients *PBC*1 and *PBC*2, defined as follows:

$$NRF = U_P / U_R \tag{1}$$

   Where U_P is the set of genomic positions to which 5′ ends of reads map uniquely and U_R is the total number of uniquely mapped reads.

$$PBC1 = M_1 / M_0 \tag{2}$$

$$PBC2 = M_1 / M_2 \tag{3}$$

   Where M_0, M_1, and M_2 are the numbers of distinct genomic locations to which at least one, exactly one, and exactly two reads map uniquely, respectively.

   These three metrics should be calculated on the chrM-filtered pre-deduplication BAM file (as the deduplication procedure eliminates redundant reads with the same coordinates).

   However, as ATAC-seq is generally carried out from the same amount of starting material, the direct estimation of absolute library size can be a useful metric that can be directly compared across different datasets. Multiple, more or less advanced in their mathematical sophistication approaches have been presented for estimating absolute library complexity (e.g. Preseq [44]). As ground truth is inherently difficult to establish in this case, it is not clear to what extent the estimates that these methods provide are accurate, but we have found

them useful in our practice as rough guides. We estimate absolute library size using the `EstimateLibraryComplexity` program in `PicardTools` as follows:

```
module load java; java -Xmx4G -jar
picard-tools-1.99/EstimateLibraryComplexity.jar
INPUT=SAMPLE.2x36mers.hg38-no-haps.unique.nochrM.bam
OUTPUT=SAMPLE.2x36mers.unique.nochrM.est_lib_complex_metrics.txt
```

Generally, high library size values are desired.

Total library sizes for the example ENCODE datasets used for illustration here are shown in Fig. 4.

7. Calculating the extent of mitochondrial contamination.

   The fraction of mitochondrial reads is calculated according to the following formula:

$$MRF = \frac{|R_M|}{|R_M| + |R_N|} \tag{4}$$

   Where R_M and R_N are as defined above.

   While low *MRF* values are generally desirable, it should be pointed out that a high (though perhaps not extremely high) fraction of mitochondrial contaminants does not directly correspond to low degree of enrichment for open chromatin regions. However, it is a highly useful metric for assessing the performance of the experimental protocol and/or the properties of the cells studied (highly metabolically active cell types, e.g. muscle cells and some cancer cell lines, tend to have many more mitochondria in each cell, and accordingly exhibit higher degrees of mitochondrial contamination in ATAC-seq datasets [41]), which can be used to suggest improvements in the experimental procedures used leading to significant cost savings in terms of sequencing expenditures.

   MRF values for illustrative ENCODE datasets are shown in Fig. 6b.

8. Estimating the fragment length distribution.

   The fragment length distribution of libraries is evaluated based on the chrM-filtered post-deduplication BAM file (including the mitochondrial-mapping fragments can result in misleading results, as mitochondria are not packaged in nucleosomal particles). It is carried out as follows:

```
python PEInsertDistFromBAM.py
SAMPLE.2x36mers.unique.nochrM.dedup.bam hg38.chrom.sizes
SAMPLE.2x36mers.unique.nochrM.dedup.InsLen
-uniqueBAM -normalize
```

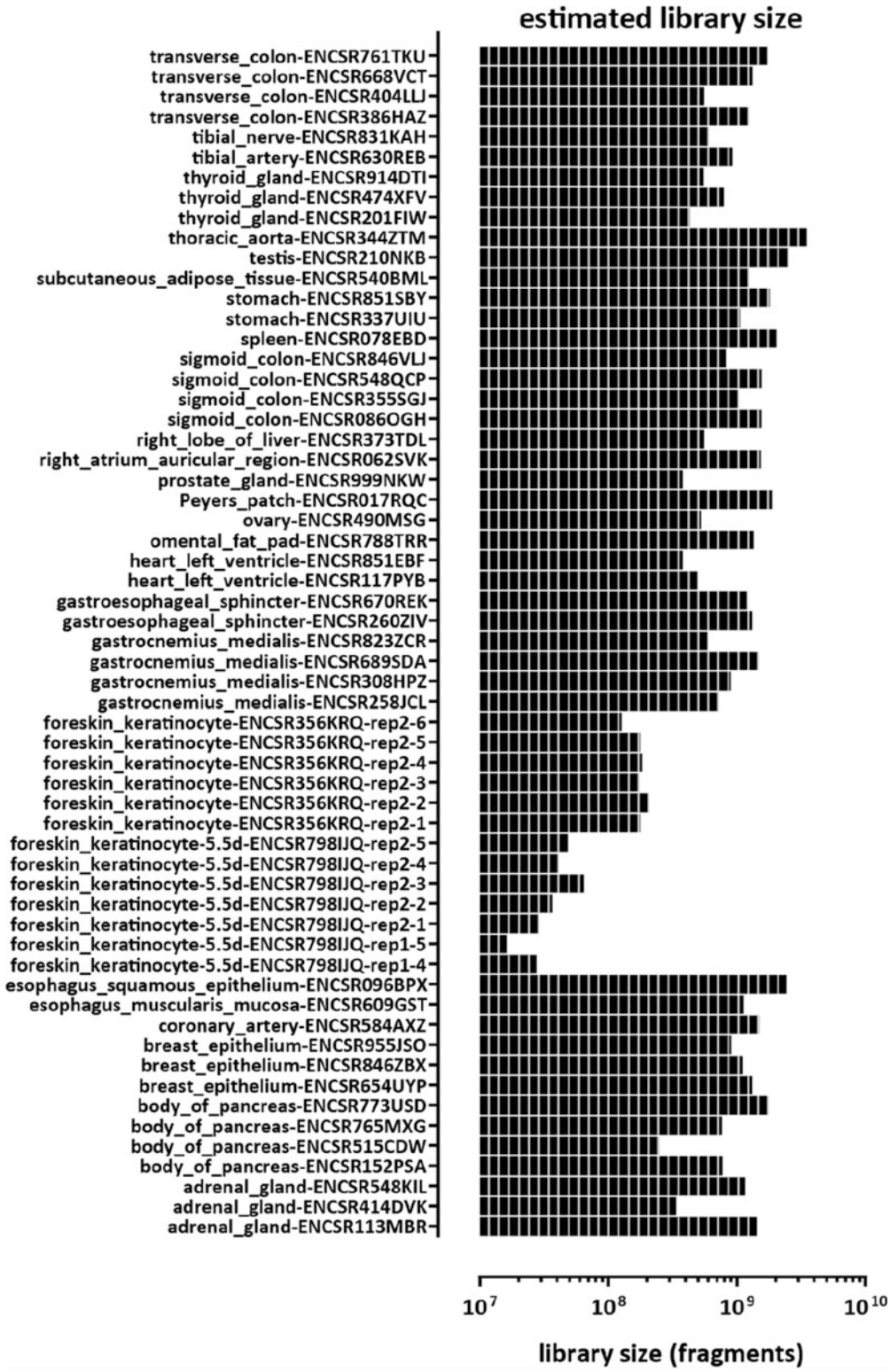


Fig. 4 Estimated absolute library sizes in example ATAC-seq datasets from the ENCODE Project Consortium. Values were calculated using PicardTools

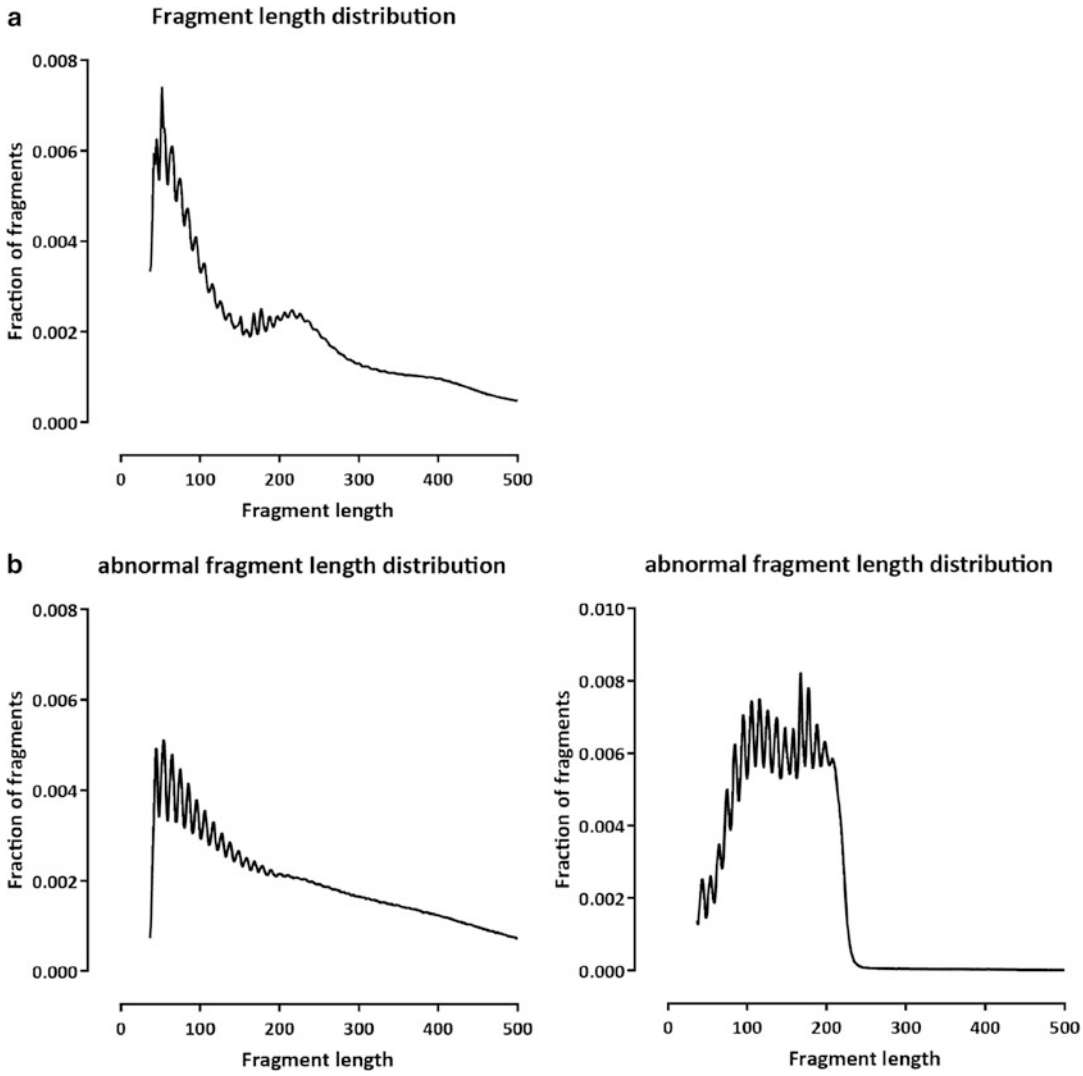


Fig. 5 Fragment length distribution in ATAC-seq datasets. (**a**) Typical ATAC-seq fragment length distributions display a prominent subnucleosomal peak and a peak corresponding to fragments encompassing one nucleosome, as well as a much smaller peak corresponding to dinucleosomal fragments (the example shown is ENCODE accession ID ENCSR404LLJ). (**b**) Examples of abnormal fragment length distributions (ENCODE accession IDs ENCSR031HDN and ENCSR939XWM)

A typical ATAC-seq fragment length distribution (Fig. 5a) is characterized by a prominent subnucleosomal component as well as smaller peaks corresponding to mononucleosomal and dinucleosomal fragments. In addition, a 10-basepair periodicity with a smaller amplitude is overlaid onto this general pattern. It corresponds to the length of the helical turn of DNA in the context of the wrapping of DNA molecules by nucleosomes and other proteins.

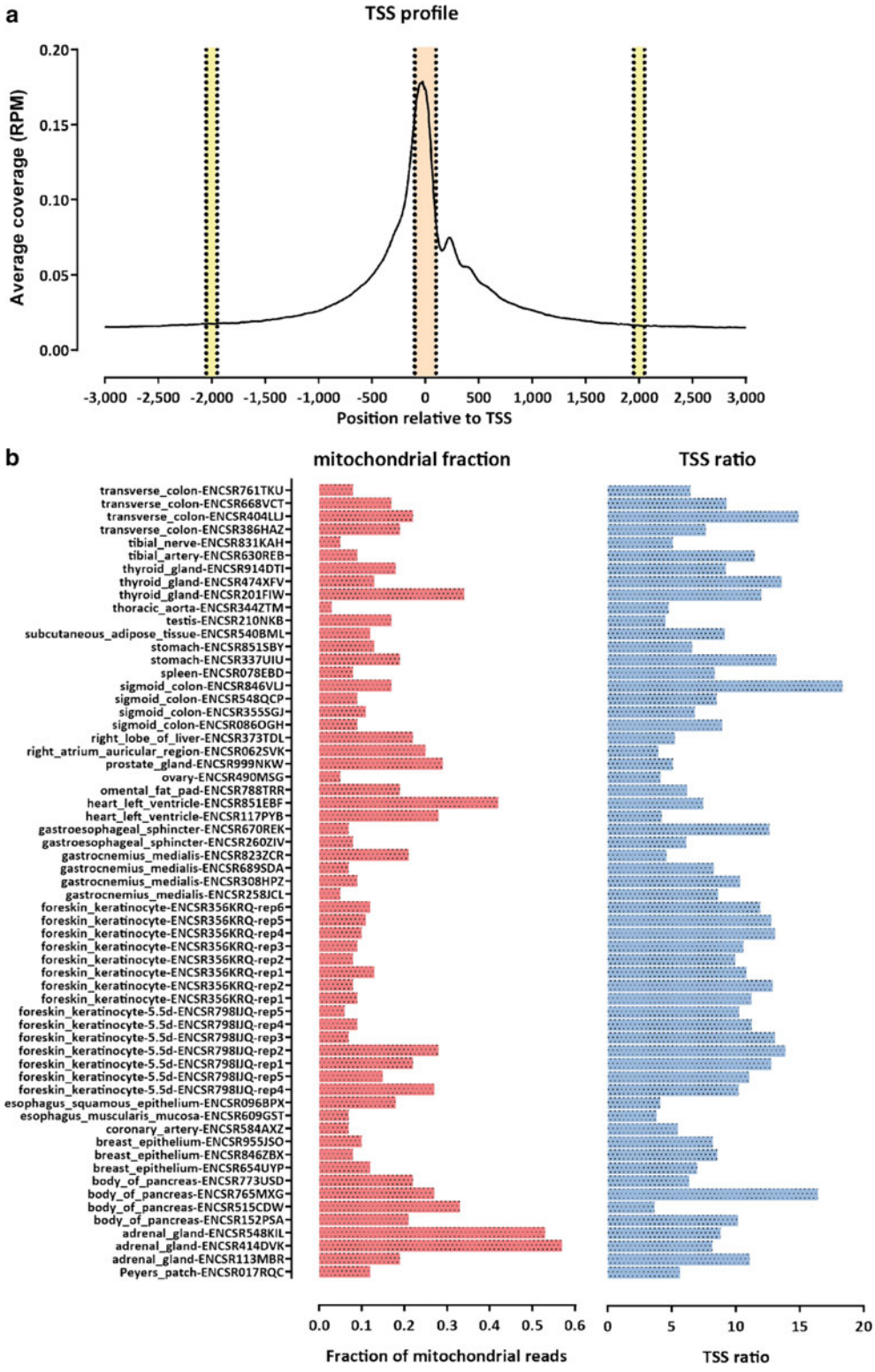


Fig. 6 ATAC-seq quality assessment metrics. (**a**) The TSS ratio quantifies the extent of enrichment in an ATAC-seq library in an internally controlled, independent from peak calling thresholds way. It is calculated by compiling an aggregate TSS profile plot around annotated protein coding TSSs, then dividing the integrated accessibility signal in the 100-bp radius-window around the TSS point to the integrated accessibility signal observed in 100-bp windows at the points ± 2 kbp away from the TSS. (**b**) Mitochondrial read fractions and TSS ratios in selected ENCODE ATAC-seq datasets

Abnormal fragment length distributions can be related to poor enrichment for open chromatin regions, though this is not necessarily always the case. They are, however, a sign of some deviation from standard practices in the experimental protocol and estimating them is thus highly useful for optimizing wet lab procedures. Examples of atypical/abnormal fragment length distributions are shown in Fig. 5b.

9. Evaluating the degree of enrichment for open chromatin.

The most important for downstream analysis QC metrics concern the extent of enrichment for accessible regions of the genome in the library.

The simplest such metric is FRiP [43] (the Fraction of Reads in Peaks), which calculates the fraction of reads in a library that fall within called peaks. However, it depends on the specific thresholds and peak definitions employed by the peak calling algorithm used, which makes it suboptimal for evaluating enrichment across many datasets.

In the context of the ChIP-seq assay, very helpful peak calling-independent metrics for assessing enrichment have been developed based on cross-correlation analysis [43, 45]. However, they are not applicable to ATAC-seq as there are no characteristic strand asymmetry patterns around fixed points in the genome in ATAC-seq datasets.

Instead, the most powerful peak calling-independent enrichment metric for ATAC-seq is TSS enrichment, which is based on creating an aggregate profile curve around the transcription start sites of protein coding genes, and calculating the ratio of the average signal in a small (typically 100-bp radius) window around the TSS versus the combined average signal in the two 100-bp long windows flanking the TSS at a distance of 2 kbp, i.e. as follows (also illustrated in Fig. 6a):

$$TSS_E = \frac{|R \in [TSS \pm 100]|}{\begin{array}{r}|R \in [TSS - 2050, TSS - 1950]| + \\ |R \in [TSS + 1950, TSS + 2050]|\end{array}} \quad (5)$$

Another advantage of the TSS enrichment metric is that it is largely independent of sequencing depth—just a few thousand reads can be used to quite accurately evaluate the enrichment of a given library. Thus it is ideally suited for small initial QC-focused sequencing runs, before commitment to deep sequencing of many libraries, and it can also be applied at the level of individual cells in the context of scATAC-seq.

Several cautionary notes need to be mentioned regarding the metric. As it is calculated relative to annotated TSSs, it is sensitive to the annotation used.

First, highly complex annotations may not be optimal for the purpose of calculating the TSS enrichment as they contain

numerous noncoding transcripts and alternative promoters. More reliable sets of TSSs, including only protein coding genes and few alternative isoforms per gene, are typically the optimal choice.

Second, different species can exhibit widely varying TSS_E scores, depending both on the properties of their genomes and the available annotations. High-quality ATAC-seq datasets in mouse and human exhibit TSS_E scores ≥ 10, which are also the species for which the vast majority of ATAC-seq datasets are generated. The TSS_E scores and the calibrations developed so far are a reliable way to evaluate ATAC libraries from these two organisms. These guidelines are, however, not similarly applicable even for other mammals due to the absence of reliable gene annotations. Very often 5′ UTRs are either incorrectly annotated or completely missing, leading to an artificial depression of the apparent TSS_E scores as there is no proper centering around the accessibility peak at the TSS. Species with smaller, more compact genomes also tend to exhibit lower TSS_E scores (e.g. in the $TSS_E = 2–3$ range of yeast and flies), due to a combination of poor TSS annotation and generally higher and denser transcriptional activity.

Yet when the same species is analyzed with the same annotation, the metric is consistent, robust, and the most reliable way to evaluate the enrichment of an ATAC-seq library. It can also be applied to other open chromatin enrichment assays such as DNase-seq.

As a one-time step, create a TSS 0-radius BED file, as follows (in this case, using the refSeq annotation for the human genome):

```
python TSS_bed_FromGTF.py refSeq.gtf 0 0 refSeq.TSS-0bp.bed
```

For each sample, generate an average profile around TSSs, e.g. as follows, or using equivalent `deepTools` commands:

```
python signalAroundCoordinate-BW.py
refSeq.TSS-0bp.bed 0 1 3 4000
SAMPLE.2x36mers.unique.nochrM.dedup.coverage.bigWig
SAMPLE.2x36mers.unique.nochrM.dedup.coverage.TSS_profile
 -normalize
```

Then calculate TSS_E values:

```
python ATACTSSscore.py
SAMPLE.2x36mers.unique.nochrM.dedup.coverage.TSS_profile
 100 2000 >> ATACTSSscore.txt
```

TSS_E values for illustrative ENCODE datasets are shown in Fig. 6b.

3.6 Peak Calling and Identification of Reproducible Peaks

Once the quality of libraries is ensured, the core analysis steps (Fig. 2b) in a typical ATAC-seq workflow can be carried out. Identifying open chromatin regions/peaks is the first step of that process most of the time. Peak calling tools specifically designed with ATAC-seq in mind are now becoming available [46], but they remain to be thoroughly benchmarked. The MACS2 peak caller [27], originally developed for ChIP-seq datasets, has been the workhorse for ATAC-seq peak calling, with some modifications applied to the settings it is run with.

However, the default output of MACS2 is typically not used as the final set of peak calls. The reason is that it, as is also true for all other peak callers used in isolation, sets arbitrary and not necessarily optimal thresholds to define peak calls. In addition, a properly designed and executed experiment has two or more replicate libraries, and the analysis would ideally incorporate information from these replicates to identify robustly reproducible peaks.

In order to accomplish these goals, the IDR (Irreproducible Discovery Rate) framework has been developed, as part of the efforts of the ENCODE Project Consortium [28]. The objective of IDR analysis is to identify and separate the set of peaks that are reproducible between replicates from those that are not reproducible using the peak ranks (according to any metric suitable for the purpose of ranking peaks). This operation can be carried out in isolation on any two sets of peaks, but the actual procedure used to derive a final set of peaks involves running several different IDR comparison steps, as described below (Fig. 7).

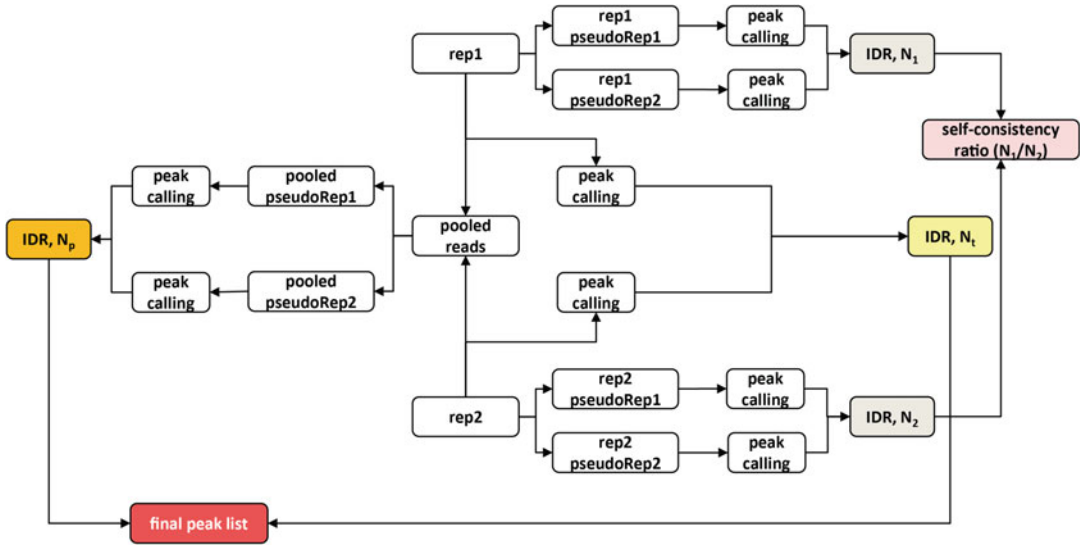


Fig. 7 Overview of the IDR analysis procedure. IDR is run both on pooled pseudoreplicates and on individual replicates, and these runs are used to identify the final set of reproducible across replicates peaks. IDR is also run on individual pseudoreplicates in order to identify discrepancies between the individual replicates used as input

An important caveat is that for IDR to work correctly, the algorithm needs to be presented with sufficiently large samples of both the reproducible and the irreproducible components of the peaks [43]. For this reason, peak calls used as input to IDR are generated with extremely relaxed peak calling settings (in order to allow irreproducible peaks into the peak call set).

Starting from two replicate ATAC-seq libraries, the standard IDR procedure for identifying reproducible peaks can be summarized as follows (Fig. 7). First, peaks are called with relaxed settings on each of the individual replicates and IDR is run to derive a "true replicate" set of peaks N_t. Then reads from the two replicates are pooled and "pseudoreplicate" BAM files are created by randomly dividing the pooled set of reads into two halves. Peaks are called with relaxed settings on each of the two pseudoreplicates, then IDR is run to derive a set of peaks N_p reproducible between pseudoreplicates. Peaks are also called with relaxed settings on the pooled set of reads, and these peak calls are used to extract the final set of peaks, typically by taking the top $max(N_t, N_p)$ peaks, though one can also use the usually more conservative N_t cutoff.

The use of pseudoreplicates allows one to compensate for cases when there are significant discrepancies in the sequencing depth and/or quality of one of the replicates. It also enables a peak caller threshold-independent approach to calling peaks at the level of an individual replicate, by splitting it into individual pseudoreplicates and identifying peaks that are reproducible between them. This step is a standard part of the IDR pipeline where it is used as a quality control measure to identify discordant sets of replicates (i.e. where there are significant differences in the N_1 and N_2 individual pseudoreplicate peak call sets).

3.6.1 IDR Analysis Pipeline

1. Run MACS2 on individual replicates:

```
macs2 callpeak -t Rep1.unique.nochrM.dedup.bam -n Rep1.MACS2
-g hs -f BAMPE --to-large -p 1e-1 --keep-dup all --nomodel
macs2 callpeak -t Rep2.unique.nochrM.dedup.bam -n Rep2.MACS2
-g hs -f BAMPE --to-large -p 1e-1 --keep-dup all --nomodel
```

2. Merge BAM files for the individual replicates:

```
samtools merge Rep1Rep2.pooled.bam
               Rep1.unique.nochrM.dedup.bam
               Rep2.unique.nochrM.dedup.bam
```

3. Sort the merged BAM files:

```
samtools sort Rep1Rep2.pooled.bam Rep1Rep2.pooled.sorted.
bam
```

4. Generate pseudoreplicates for the pooled replicates:

```
python BAMPseudoReps.py Rep1Rep2.pooled.sorted.bam
       samtools hg38/sequence/hg38_no_alt.fa
```

5. Generate pseudoreplicates for each individual replicate:

```
python BAMPseudoReps.py Rep1.unique.nochrM.dedup.bam
       samtools hg38/sequence/hg38_no_alt.fa
python BAMPseudoReps.py Rep2.unique.nochrM.dedup.bam
       samtools hg38/sequence/hg38_no_alt.fa
```

6. Call peaks on the pooled dataset as previously shown.
7. Call peaks on the pooled pseudoreplicates as previously shown.
8. Call peaks on individual pseudoreplicates as previously shown.
9. Take the top 300,000 of each of the relaxed peak call sets using the MACS2 *p*-value as a ranking measure, e.g. for replicate 1:

```
cat Rep1.MACS2_peaks.narrowPeak | sort -k 8nr,8nr |
   awk 'BEGIN{OFS="\t"}{$4="Peak_"NR ; print $0}' |
   head -300000 | gzip -c >
   Rep1.MACS2_peaks.narrowPeak.sorted.300K.gz
```

10. Run IDR on individual replicates:

```
idr-2.0.4.2/bin/idr --samples
Rep1.MACS-2.1.0.p1e-1_peaks.narrowPeak.sorted.300K.gz
Rep2.MACS-2.1.0.p1e-1_peaks.narrowPeak.sorted.300K.gz
--input-file-type narrowPeak --output-file Rep1Rep2.indRep.
IDR
--peak-list Rep1Rep2.pooled.sorted.narrowPeak.sorted.300K.gz
--rank p.value --soft-idr-threshold 0.05 --plot
```

11. Run IDR on pooled pseudoreplicates as above.
12. Run IDR on individual pseudoreplicates as above.

Example of IDR analysis output is shown in Fig. 8.

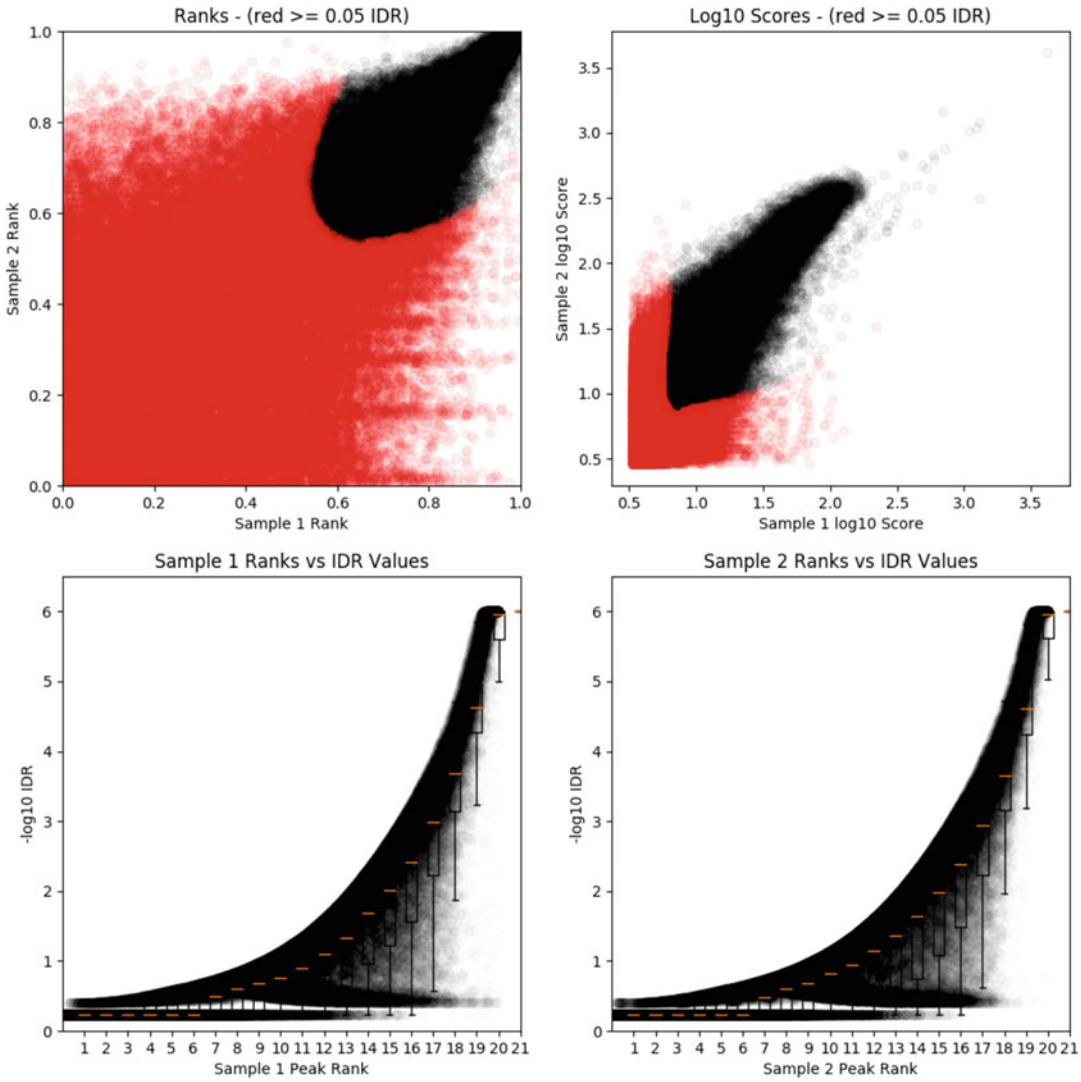


Fig. 8 Example IDR analysis output showing peak reproducibility as a functioning of peak ranks. A true replicate comparison for ENCODE dataset ENCSR260ZIV is shown

13. Examine the IDR output files to determine the N_t, N_p, N_1, N_2 values, e.g. for N_t:

```
awk '$11 <= 0.02 {print $0}' Rep1Rep2.indRep.IDR | wc -l
```

The examples provided here use an individual replicate IDR cutoff of 0.05 which works reasonably well for most purposes. If more stringent peaks are wanted, individual replicate thresholds of 0.02 or 0.01 can also be applied.

14. Pick the top $max(N_t, N_p)$ peaks from the peak calls generated from the pooled replicates:

```
zcat Rep1Rep2.pooled.sorted.narrowPeak.sorted.300K.gz |
  head -n $max(N_t,N_p) | cat > Rep1Rep2.IDR0.05.bed
```

If more than two biological replicate samples are available, IDR can be run between all pairs of replicates to determine the maximum set of reproducible peaks.

3.6.2 Removing Known Artifacts

Most genome assemblies include problematic regions that generate strong artifactual enrichment in most sequencing-based assays, e.g. due to the presence of collapsed repeats that are included as a single copy in the assembly but exist in numerous copies in real cells. In order to avoid biases introduced by including such regions in downstream analyses, it is necessary to exclude these so-called blacklist regions. Blacklist sets have been generated for multiple species by the ENCODE Project Consortium [23], and are readily available from its portal.

The post-IDR set of peaks is filtered against blacklists as follows:

```
bedtools intersect -v -a Rep1Rep2.IDR0.05.bed
-b hg38_blacklist.bed -wa > Rep1Rep2.IDR0.05.noBlacklist.bed
```

3.7 Merging Peaks and Creating Multisample Data Matrices

The typical next step in the analysis workflow is to create a data matrix that combines all samples that are to be analyzed together. To this end, the regions derived from the peak calling procedure in individual samples/conditions, each of which has different coordinates even when overlapping a peak from another sample/condition, need to be merged together so that only one set of coordinates remains for each location of enrichment in the genome. There is no consensus strategy for merging peaks and multiple approaches have been used with generally equal success. For ATAC-seq, perhaps the simplest strategy is to split the genome in bins, e.g. 500 bp each, and to only retain bins that overlap called peaks. Another strategy is to iteratively merge peaks if their summits are within a specified distance from each other, with a new summit corresponding to the summit of the stronger of the two peaks; the final list of peaks is derived by extending the resulting final list of summits by a fixed distance.

Each of these procedures results in a uniform set of coordinates, which are used to compile a final data matrix. Such a matrix can contain read counts, RPM values, or some other measure.

3.8 Identifying Differentially Accessible Regions

A common task in ATAC-seq analysis is to identify regions that are differentially accessible between two conditions. Standard tools originally developed for the analysis of RNA-seq datasets, such as DESeq2 [31], edgeR [47], limma [48], and others can be used for this purpose. Most such packages require read counts as inputs, as the statistical models that they employ are based on discrete distributions (e.g. negative binomial in the case of DESeq/edgeR). The major difference between differential analysis in the context of RNA-seq and its application to ATAC-seq is that stable static gene annotations are used in the former case, while for ATAC-seq no such set of features is available, and it has to be compiled through the peak merging procedures discussed in the previous section.

Once peaks have been merged, read counts can be generated for all samples combined as follows:

```
featureCounts -F SAF -a Merged_peaks.saf -o All_samples.counts
Sample1.bam....SampleN.bam
```

Note: when using `featureCount`, a file in `.saf` format is needed, i.e. coordinates are specified as follows:

```
ID <tab> chr <tab> start  <tab> end
```

Once a read count matrix has been compiled, it can be used as input to DESeq/DESeq2. While DESeq2 allows for arbitrarily complex design matrices to be employed for differential analysis, here we illustrate its use with a simple two-condition two-replicates-per-condition use case.

Within `R`, we first invoke the DESeq2 library:

```
library("DESeq2")
```

Then the count matrix is converted into an `R` object:

```
pasillaCountTable <- read.table("samples.counts.table",
                                header=TRUE, row.names=1)
samples <- data.frame(row.names=c(
           "condition1-rep1", "condition1-rep2",
           "condition2-rep1", "condition2-rep2"),
           condition=c("condition1", "condition2"))
```

After that, mean and dispersion estimates for the two conditions are derived:

```
dds <- DESeqDataSetFromMatrix(countData = pasillaCountTable,
                      colData=samples, design=~condition)
dds_1 <- DESeq(dds)
```

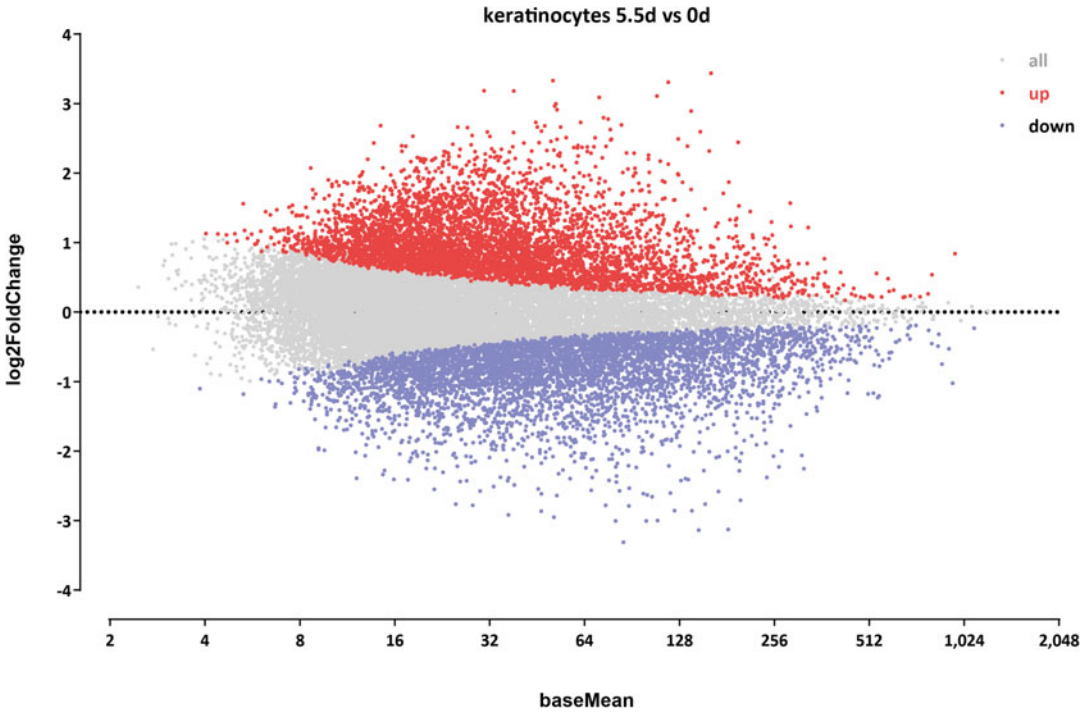


Fig. 9 MA plot showing an example of differential accessibility analysis results. Differentially accessible regions were identified between the 0-day and the 5.5-day time points of a skin differentiation times course (ENCODE accessions ENCSR798IJQ and ENCSR356KRQ)

Finally, differential fold change and significance values are generated for downstream filtering and analysis:

```
res <- results(dds_1, contrast=c(
          "condition", "condition1", "condition2" ))
write.table(res, "samples.condition1-vs-condition2.csv")
```

An example of the results from differential accessibility analysis is shown in Figs. 9 and 10 using the ENCODE keratinocyte differentiation time course ATAC-seq dataset.

3.9 Visualizing Signal Around Genomic Features Using Heatmaps

We use the case of differential accessibility to demonstrate another common task encountered in the analysis of ATAC-seq (and other functional genomic data), visualizing ATAC signal around a set of genomic features, often across multiple datasets.

In this case, we start with modified DESeq output files in the following format, one each for the set of "up" and "down" peaks:

```
#chr left right baseMean log2FoldChange
```

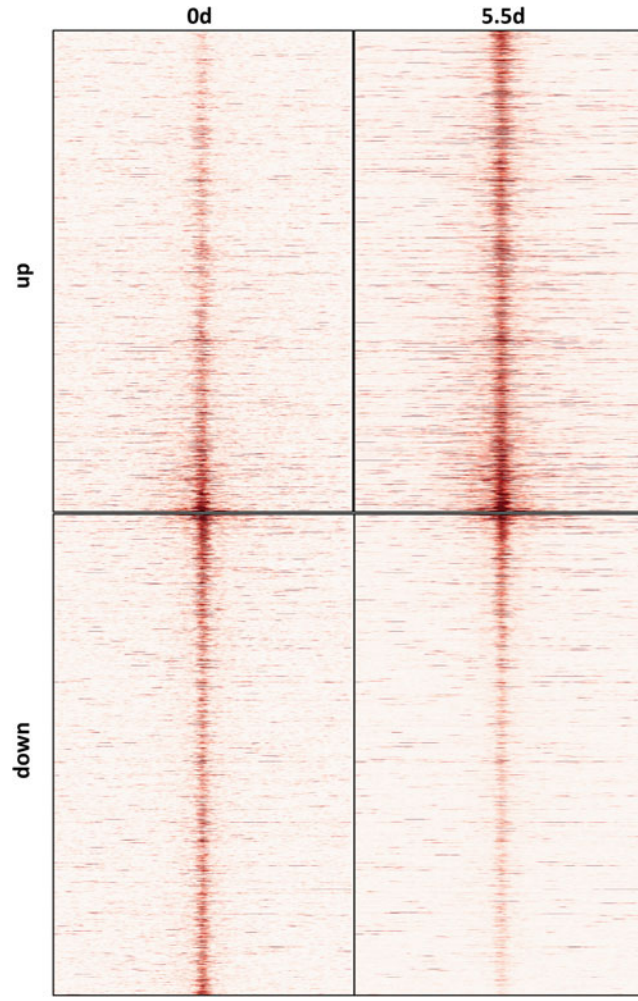


Fig. 10 Heatmaps of normalized ATAC-seq signal for differentially accessible regions during keratinocyte differentiation. Differentially accessible regions were identified between the 0-day and the 5.5-day time points of a skin differentiation times course (ENCODE accessions ENCSR798IJQ and ENCSR356KRQ). Regions were sorted according to their fold change values across the two conditions as estimated by DESeq2

We calculate a coverage matrix containing the ±2 kb profile around the midpoint of each differentially accessible peak (sorting the matrix by the $\log_2$(FoldChange) value):

```
python signalAroundCoordinate-individual.py
   foreskin_keratinocyte.0d-vs-5.5d.up.bed
   0 midPoint -noStrand 2000 2000
     foreskin_keratinocyte-5.5.2x36mers.unique.nochrM.dedup.
bigWig
   foreskin_keratinocyte.0d-vs-5.5d.up.5.5d.matrix -sortby 4
```

We also generate the same matrices for the "down" set of peaks and for the "0d" condition. The resulting matrices can then be visualized (Fig. 10) using a number of tools (`R`, `matplotlib`, and others). A similar procedure can also be implemented using functions from the `deepTools` package.

3.10 Data Exploration

In order to examine the relationship between samples in a high-dimensional dataset, it is usually useful to apply a dimensionality reduction technique that performs a mapping of the data onto a lower-dimension space (e.g. one with two or three dimensions) that is possible to visualize.

The most well-known such technique is PCA (Principle Component Analysis), which carries out a linear transformation of the data in such a way that the variance along each of the principle components identified is maximized in a decreasing order. Data points are then projected along some combination of principle components in a low-dimensional space (most often, for data exploratory purposes, the first and the second).

A variety of PCA implementations exist in all statistical and programming languages. A PCA analysis can be carried out in `R` as follows:

```
library(ggfortify)
library(tidyverse)
rpm_fixed = read_tsv("samples.RPM.table", skip = 1)
pca = prcomp((as.matrix(rpm_fixed))

autoplot(pca, x = 1, y = 2)
```

Figure 11 shows the first two PCA projections for ENCODE ATAC-seq datasets.

With the advent of single-cell genomic methods (such as scRNA-seq and scATAC-seq), datasets of increasingly high dimensionality and complex structure have become available, for which simple dimensionality reduction methods such as PCA are ill suited. Instead, novel methods have been employed (e.g. *t*-SNE, or *t*-distributed Stochastic Neighbor Embedding, [49]), or specifically developed for the purpose of working with such datasets (e.g. UMAP, or Uniform Manifold Approximation and Projection for Dimension Reduction, [50]).

These techniques can also be applied to bulk ATAC-seq datasets. The main tool the field used for several years was *t*-SNE. However, it has now been largely replaced by UMAP as field standard, as UMAP is much faster and less memory intensive, and, most importantly, it preserves global data structure and relationships between data points, unlike *t*-SNE.

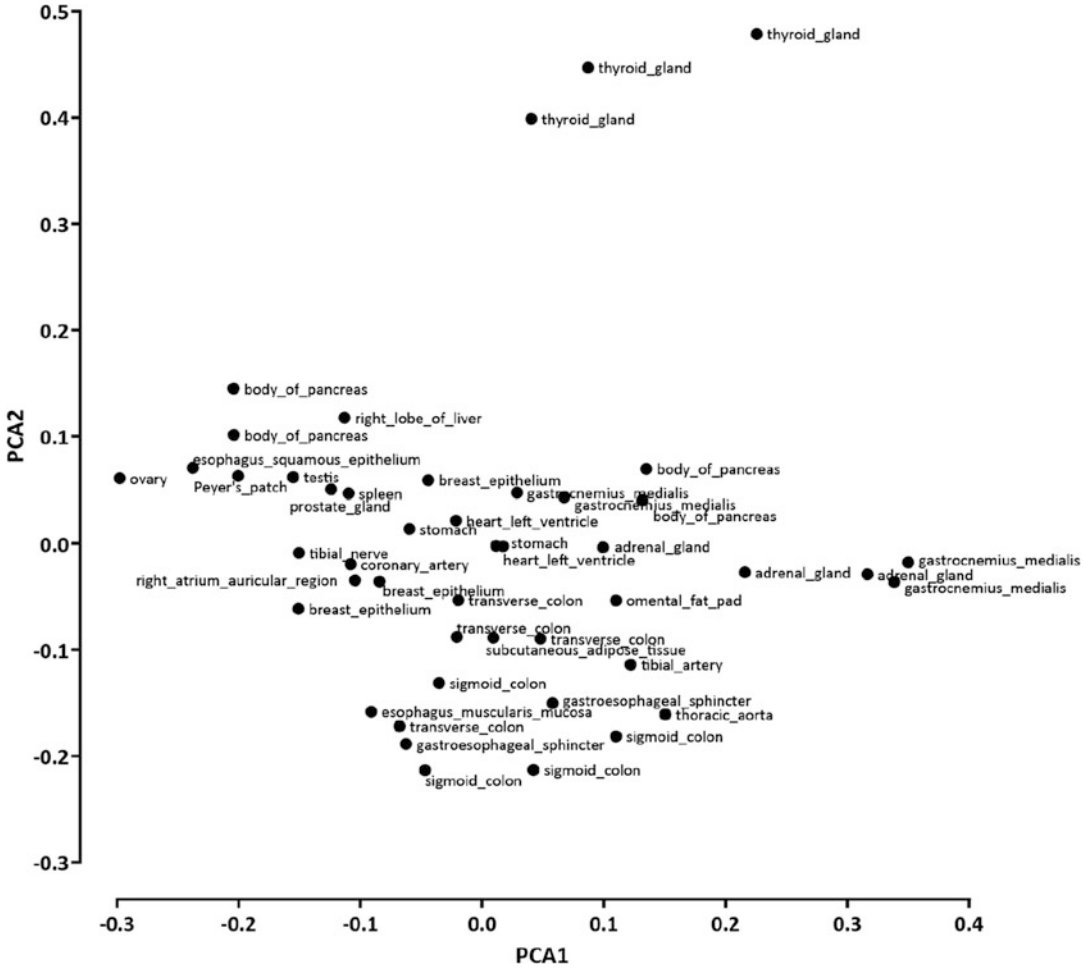


Fig. 11 First two PCA projections for human ENCODE ATAC-seq datasets

UMAP can be, in its simplest form, run from inside Python as follows:

First, UMAP and `numpy` are loaded:

```
import numpy as np
import umap
```

Second, a `numpy` array object containing the ATAC matrix is created.

Then UMAP is run:

```
reducer = umap.UMAP()
embedding = reducer.fit_transform(data)
```

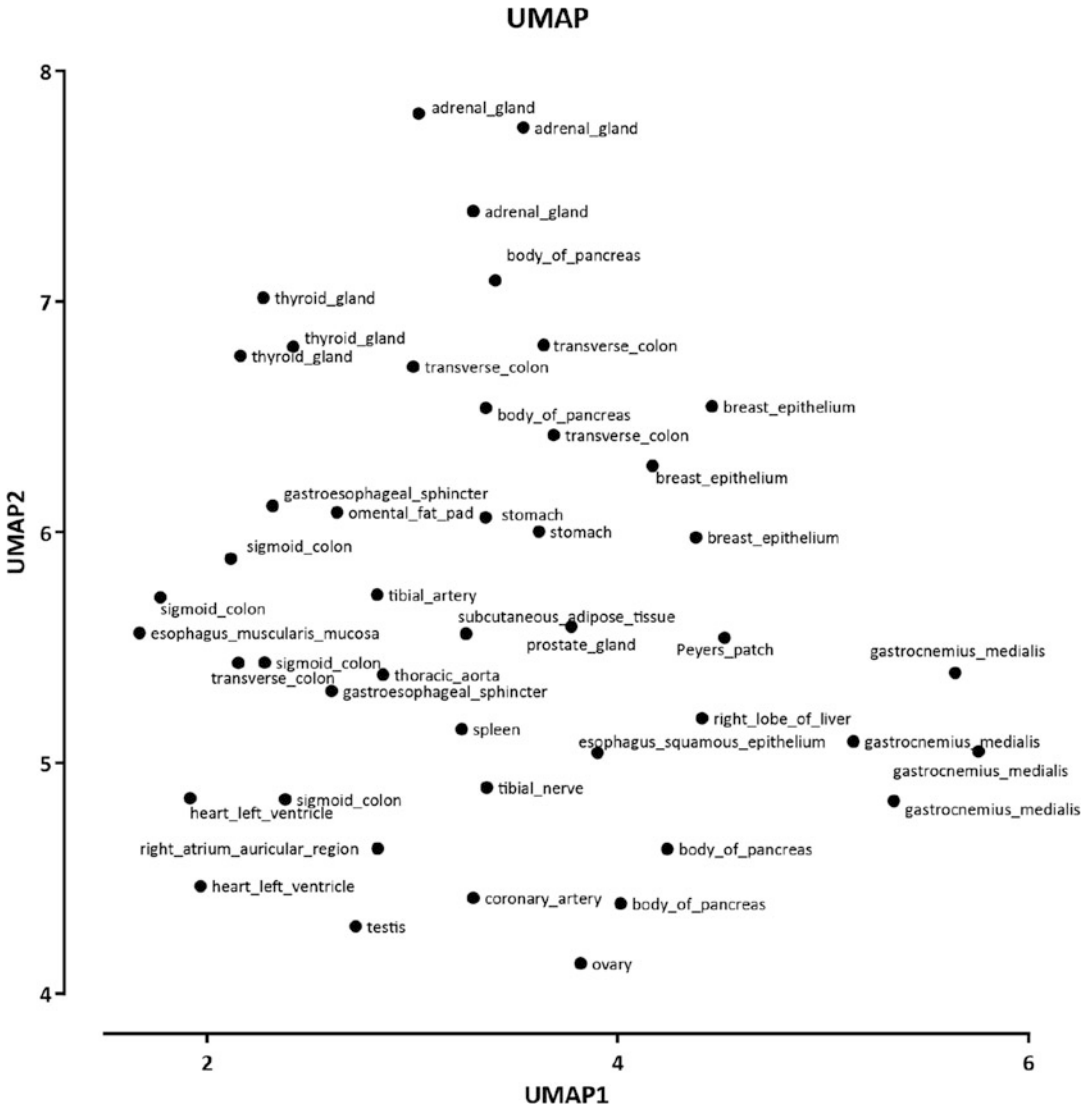


Fig. 12 UMAP projections for human ENCODE ATAC-seq datasets

The `embedding` object contains the projections along the first two UMAP dimensions. It should be noted that while default settings often work well, UMAP output depends on several hyper-parameters; for a more detailed discussion of these the reader is referred to the online UMAP documentation.

UMAP projections for ENCODE ATAC-seq datasets are shown in Fig. 12.

3.11 Clustering of Accessible Regions Across Conditions/Cell Types

Clustering of ATAC-seq peaks across cell lines, tissues, or times can identify sets of genomic regions exhibiting unique biologically relevant dynamic accessibility patterns. Subsequent analyses can then focus more deeply on such subsets of peaks, e.g. by analyzing their sequence properties.

It is not recommended to perform clustering on all peaks, as most of them typically show no differences between conditions. Filtering peaks and retaining the most variable ones are thus usually performed prior to clustering. One approach for doing this is to take peaks that are called as differential by DESeq2/edgeR between different conditions.

Before clustering the differential peaks, it is recommended to normalize the read counts, e.g. as $\log_2$(Reads/Counts Per Million), both for visualization purposes and in order to correct for differences in sequencing depth.

From within R:

```
library(viridis)
library(tidyverse)
library(tglkmeans)

res = read_csv("samples.condition1-vs-condition2.csv")
rpm_fixed = read_tsv("samples.RPM.table")

rpm_filter = rpm_fixed[order(res$padj),
                       4:dim(rpm_fixed)[2]][1:35000]
rpm_filter = log2(rpm_filter+1)
```

Next we cluster using TGL Kmeans. Note that the number of clusters K to be used has to be decided *a priori*, and usually there is no prior reliable guess of what it should be. One possibility is to determine K using the elbow method:[1]

```
km = TGL_kmeans(rpm_filter,12,"euclid",reorder_func =
              "hclust",id_column=FALSE)

image(t(as.matrix(rpm_filter[order(km$cluster),])),,xaxt="n",
           yaxt="n",col=viridis(12),zlim=c(0,8),main="TITLE")

cur_y = 0
tot_y = 1
for (i in 1:K) {
    cur_y = cur_y + km$size[i]/sum(km$size)
```

[1] https://www.r-bloggers.com/finding-optimal-number-of-clusters/.

```
    abline(a=cur_y, b=0, col="yellow",lwd=2)
    mtext(i,side=2,line=1,at=(cur_y/tot_y)-
    ((km$size[i]/sum(km$size))/2),adj=1,cex=2,las=2)
}
```

The results from the clustering above is shown in Fig. 13.

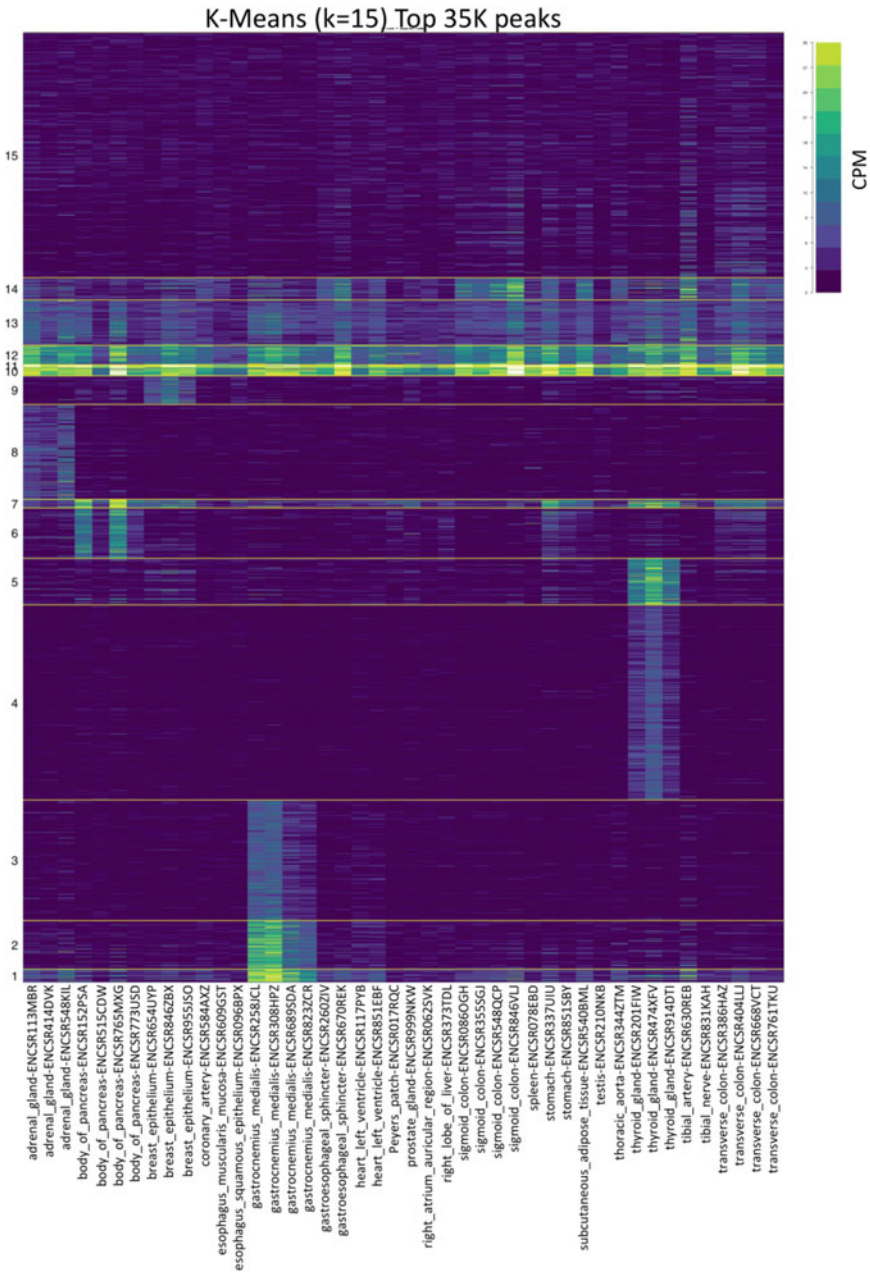


Fig. 13 K-means clustering analysis applied to human ENCODE ATAC-seq datasets. K-means clustering ($k = 15$) was carried out on the top 35,000 most variable peaks across the human ENCODE ATAC-seq datasets

3.12 Analyzing Variable Motif Accessibility Using chromVAR

Accessibility peaks are defined by the regulatory input of transcription factors, and therefore there is a relationship between the motif content of open chromatin regions and active regulatory factors in a given cell or cell type. This relationship can be captured by analyzing variable accessibility associated with transcription factor motifs. The chromVAR algorithm and R package were created to provide that functionality [32]. It works by taking as input a set of accessibility peaks and read counts for each peak across all samples in a dataset together with a collection of mapped TF motif instances. The chromVAR algorithm then identifies "accessibility deviations" for each TF and cell/cell type after correcting for a number of biases (such as variable tagmentation, GC content, mean accessibility, and others).

Within R:

```
library(chromVAR)
library(motifmatchr)
library(JASPAR2018)
library(BSgenome.Hsapiens.UCSC.hg38)
library(TFBSTools)

opts <- list()
opts["species"] <- "Homo sapiens"
opts["collection"] <- "CORE"
motifs <- TFBSTools::getMatrixSet(JASPAR2018::JASPAR2018,
opts)
if (!isTRUE(all.equal(TFBSTools::name(motifs), names(mo-
tifs))))
names(motifs) <- paste(names(motifs), TFBSTools::name(mo-
tifs),
                                                    sep = "_")

genome = "BSgenome.Hsapiens.UCSC.hg38"
tf_num = 100

rowRanges = makeGRangesFromDataFrame(rpm_fixed[,1:3])
counts = as.matrix(rpm_fixed[,4:dim(rpm_fixed)[2]])
colData = DataFrame(Tissue = names(rpm_fixed),
                            row.names=colnames(counts))

rse <- SummarizedExperiment(assays=SimpleList(counts=counts),
                        rowRanges=rowRanges, colData=colData)

counts_filtered <- addGCBias(rse, genome = genome)
motif_ix <- matchMotifs(motifs, counts_filtered, genome =
genome)

dev_TF <- computeDeviations(object = counts_filtered,
```

```
                                annotations = motif_ix)

bg <- getBackgroundPeaks(object = counts_filtered)

expected <- computeExpectations(counts_filtered)
variability_TF <- computeVariability(dev_TF)
variabilty_plot_TF <- plotVariability(variability_TF,
                                      use_plotly = FALSE,n=8)

pdf("variability_plot_TF.pdf")
print(variabilty_plot_TF)
dev.off()

sample_cor_TF <- getSampleCorrelation(dev_TF)
dist_TF <- as.matrix(as.dist(sample_cor_TF))
TF_colData <- colData(dev_TF)
TF_colNames <- TF_colData$Treatment
colnames(dist_TF) <- TF_colNames
rownames(dist_TF) <- TF_colNames

pheatmap(dist_TF, clustering_distance_rows =
     as.dist(1-sample_cor_TF),  clustering_distance_cols =
   as.dist(1-sample_cor_TF), filename = "Heatmap_TF_colnames.
pdf",
         height=10,width=10)

Highest_var_TF <- cbind(variability_TF, rownames(variabili-
ty_TF))
Highest_var_TF <- Highest_var_TF %>% group_by(name) %>%
                          dplyr::slice(which.max(variability))
Highest_var_TF <- Highest_var_TF[order(Highest_var_TF$varia-
bility,
                               decreasing = TRUE),]
Highest_var_TF <- Highest_var_TF[c(1:tf_num),]
Dev_Highest_var_TF <- assays(dev_TF)[[1]]
Dev_Highest_var_TF <- as.data.frame(Dev_Highest_var_TF)
Dev_Highest_var_TF <- cbind(rownames(variability_TF),
                                        variability_TF$name,
Dev_Highest_var_TF)
Dev_Highest_var_TF <- (subset(Dev_Highest_var_TF,
      Dev_Highest_var_TF$`rownames(variability_TF)` %in%
      Highest_var_TF$`rownames(variability_TF)`))
rownames(Dev_Highest_var_TF) <-
     Dev_Highest_var_TF$`variability_TF$name`
Dev_Highest_var_TF <- Dev_Highest_var_TF[,c(3:(dim(data)[2]
+2))]
pheatmap(Dev_Highest_var_TF, cluster_rows = T,
     cluster_cols = FALSE, scale = "row", filename =
     "Top100_TF_Heatmap.pdf",height=10,width=10)
```

The result is a map of transcription factor activity in each cell/cell type, as shown for ENCODE ATAC-seq datasets in Figs. 14 and 15.

3.13 Analyzing Transcription Factor Footprints

While not as strongly protective against enzymatic action as nucleosome occupancy, occupancy of DNA by other factors can preclude cleavage/modification too, resulting in protection "footprints." The physical association of individual transcription factor molecules with DNA lasts on the order of seconds for most factors (e.g. [51]), thus some skepticism regarding their ability to provide protection against enzymatic action is not unwarranted. Nevertheless, empirically such footprints are indeed observed [52–55], and their analysis is potentially tremendously powerful.

Two types of footprinting analysis can be carried out—globally and at the level of individual footprints. The former involves aggregate analysis across motif instances (often restricted to those within accessible regions of the genome), the latter aims at identifying individual footprints genome-wide.

Individual footprinting has been attempted by multiple studies using very deep DNase-seq data [52–55], and understandably, given its potential to map regulatory circuits, the computational problem of identifying such footprints has generated much interest by the bioinformatic community, with a large number of methods for how to optimally carry it out having been proposed [51, 56–68]. However, it is probably fair to say that the jury is still out regarding how robust footprinting is at the level of individual motif instances [69, 70].

First, very deeply sequenced datasets are necessary for individual motif footprints to become visible within accessible peaks. At least 300 million mapped reads have been typically used for this task in mammalian genomes. Second, DNase and other enzymes used to map open chromatin all have some non-negligible biases that need to be properly modeled if they are not to confound footprinting analysis.

TF footprints are also observed in ATAC-seq datasets (Fig. 16), but footprinting analysis in the ATAC-seq domain focuses primarily on global footprint profiles. As ATAC-seq libraries are usually generated starting from only 50,000 mammalian cells, they generally do not contain sufficient molecular complexity to support the depth of sequencing required to get to individual footprinting level, unless a large number of individual libraries for the same cell type are pooled together, which is rarely done.

Nevertheless, global footprinting profiles on their own are still highly informative about transcription factor activity across cell types.

These profiles are generating by calculating the average bias-corrected insertion (i.e. 5′ fragment ends) profiles around a set of

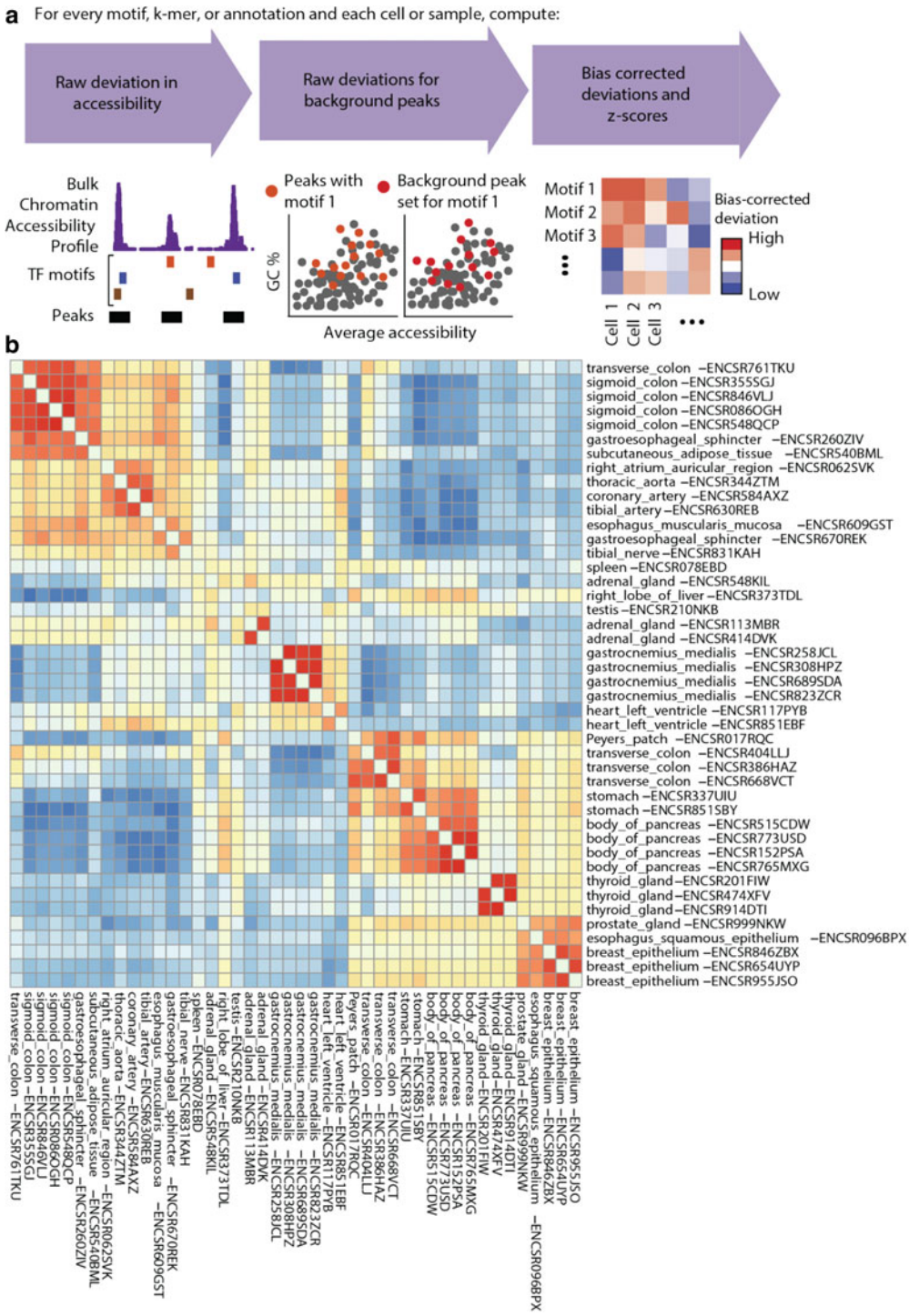


Fig. 14 Analysis of transcription factor activity variability using chromVar. (**a**) Overview of the chromVar algorithm (adapted from the original chromVar publication [32]. (**b**) Clustergram of ENCODE human ATAC-seq datasets based on transcription factor accessibility deviation scores estimated by chromVar

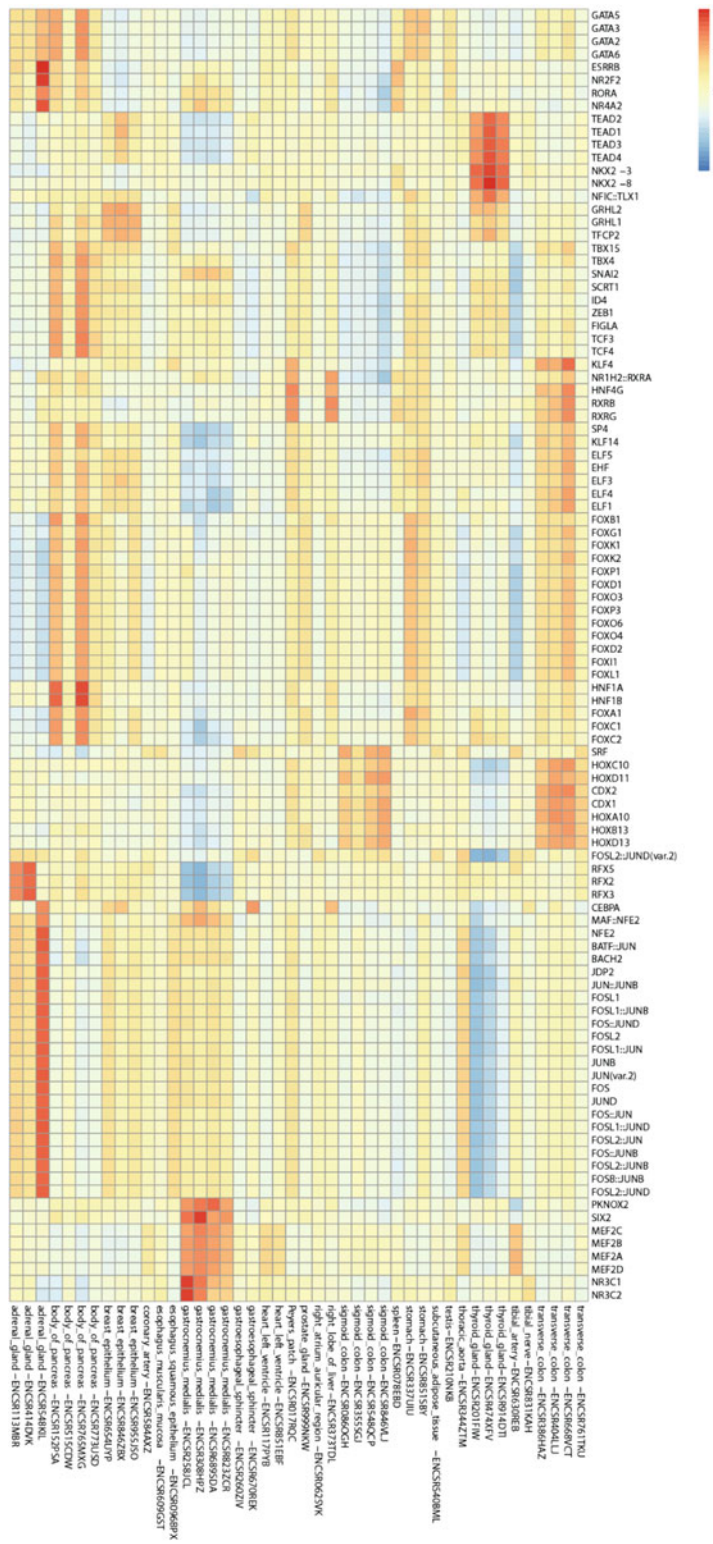


Fig. 15 Analysis of transcription factor activity variability using chromVar. Shown are transcription factor accessibility deviation scores across ENCODE human ATAC-seq datasets

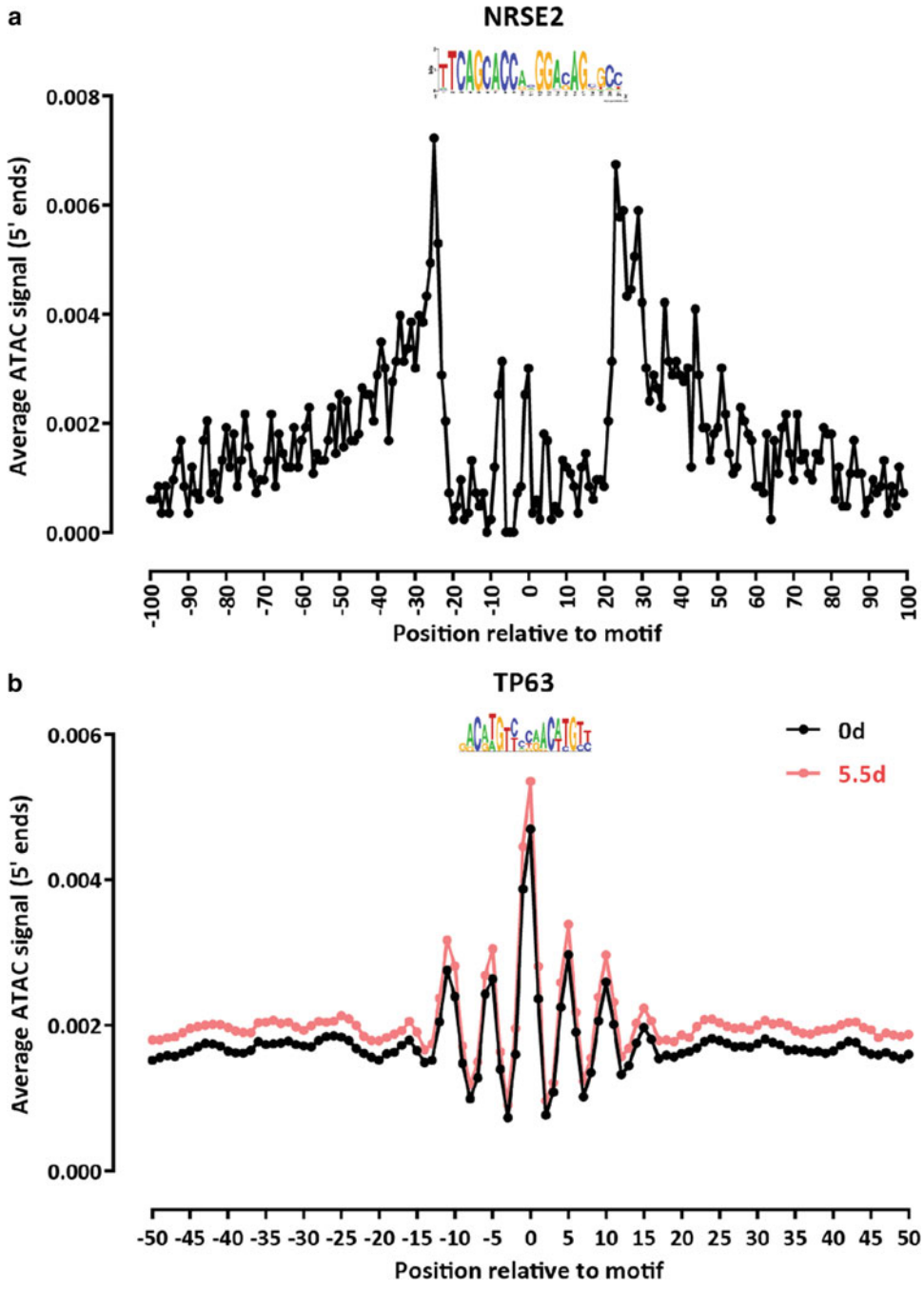


Fig. 16 Global footprinting of occupied NRSF/REST NRSE2 motifs [71] and of TP63 motifs in human keratinocytes (ENCODE accession ENCSR798IJQ). (**a**) NRSF/REST motifs were derived as previously described [71, 72], all instances of the NRSE2 motif identified genome-wife, and then intersected with publicly available ChIP-seq peaks (ENCODE accession ENCFF048JKT) to derive a set of occupied motifs. (**b**) TP63 motif definitions were obtained from the CIS-BP motif database [73] and mapped to the human genome using FIMO version 5.0.5 [74]

motif instances for each individual transcription factor after correction for the sequence bias of the Tn5 enzyme.

The transposase sequence bias is, in the simplest and most general procedure proposed [64], estimated from sequencing libraries generated by transposition of naked DNA. For a given value of k, usually $k=6$, the frequency F_i^{exp} of all k-mers in the mappable portion of genome is calculated, as is the observed frequency F_i^{obs}, corresponding to the k-mers centered on transposase insertion sites. The transposition propensity F_i^{obs}/F_i^{exp} is then calculated and used to adjust observed insertion counts in ATAC-seq datasets.

Examples of global ATAC-seq footprints for transcription factors are shown in Fig. 16. We note that the exact shape of the footprints depends greatly on the properties of the particular transcription factor. TFs with long motifs and tight and stable association with DNA (such as NRSF/REST in Fig. 16a) tend to exhibit deeper footprints than TFs that bind to shorter motif and/or occupy DNA more transiently.

3.14 V-plots

As ATAC-seq datasets contain fragments corresponding to subnucleosomal and nucleosomal fragments, they can be used to study nucleosome organization around certain genomic landmarks (such as TSSs, transcription factor binding sites, and others) based on the average fragment structure around such sites.

The V-plot [75] is the main tool for carrying out such analyses. V-plots are constructed from paired-end datasets by calculating the density of fragments on a map of the positions of sequencing fragment mid-points against the length of fragments, from the view point of a set of single-base pair genomic features.

Figure 17 shows such a V-plot centered on occupied (as measured by ChIP-seq) CTCF motifs. CTCF is known to be a strong nucleosome positioning factor [76], thus in this case it is not surprising to observe both mono- and dinucleosomal fragment density areas on the flanks relative to the CTCF sites, together with a high density of subnucleosomal fragments centered on the CTCF motif itself. In addition, the 10-bp periodicity characteristic to ATAC-seq datasets is also clearly visible as a persistent horizontal feature of the V-plot.

An example of running a custom-written script for generating V-plots is provided below:

```
python V-plot.py SAMPLE.2x36mers.unique.nochrM.dedup.bam
motifs.positions 0 1 500 500
SAMPLE.2x36mers.unique.nochrM.dedup.V-plot -stranded 2
```

Where the motifs file in this case is in the following format:

```
chr <tab> midPoint <tab> strand
```

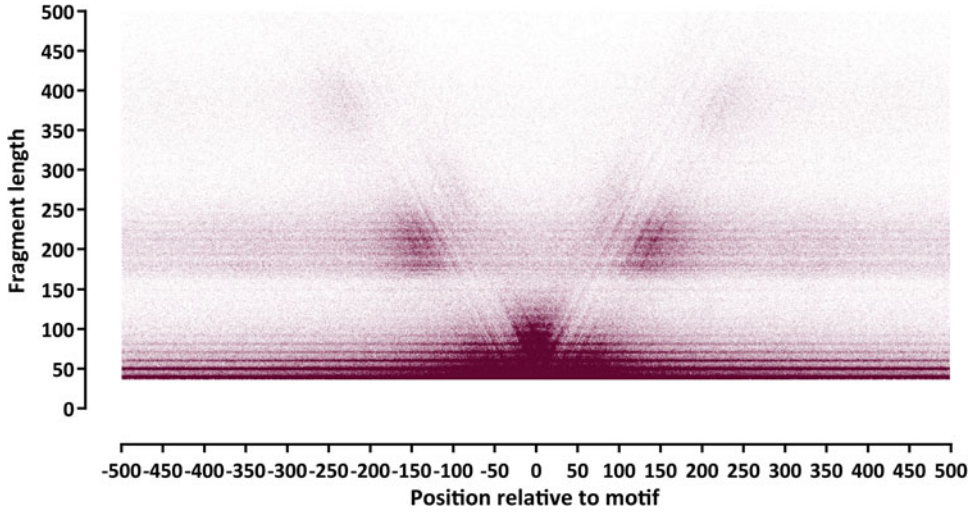


Fig. 17 ATAC-seq V-plot centered on occupied CTCF motifs. The ENCODE keratinocyte ATAC-seq dataset (0d time point) is used as an example

And the V-plot encompasses the ±500 bp neighborhood around the specified set of transcription factor motifs.

V-plots also make it possible to identify positioned nucleosomes within regulatory elements. As ATAC-seq strongly enriches for accessible chromatin, such an analysis can only be carried out in the immediate vicinity of open chromatin regions/*cis*-regulatory elements, but this is often highly informative given that cREs are where dynamics functionally relevant changes in chromatin states typically occur. The NucleoATAC algorithm [77] (based on identifying the positions in the genome in the immediate vicinity for open chromatin regions that maximize nucleosomal V-plot signatures, i.e. a V-plot in which a nucleosome-length fragment is observed centered on the V-plot midpoint) was developed to carry out this task (though we note that it is, as of the writing of this text, no longer maintained and no alternative algorithms are available).

Acknowledgements

The authors thank members of the Greenleaf and Kundaje labs for many helpful discussions. Z.S. is supported by EMBO Long-Term Fellowship EMBO ALTF 1119–2016 and by Human Frontier Science Program Long-Term Fellowship HFSP LT 000835/2017-L. G.K.M. was supported by the Stanford School of Medicine Dean's Fellowship.

References

1. Wu C (1980) The 5′ ends of *Drosophila* heat shock genes in chromatin are hypersensitive to DNase I. Nature 286(5776):854–860
2. Keene MA, Corces V, Lowenhaupt K et al (1981) DNase I hypersensitive sites in Drosophila chromatin occur at the 5′ ends of regions of transcription. Proc Natl Acad Sci USA 78:143–146
3. McGhee JD, Wood WI, Dolan M et al (1981) A 200 base pair region at the 5′ end of the chicken adult β-globin gene is accessible to nuclease digestion. Cell 27:45–55
4. Dorschner MO, Hawrylycz M, Humbert R et al (2004) High-throughput localization of functional elements by quantitative chromatin profiling. Nat Methods 1:219–225
5. Sabo PJ, Humbert R, Hawrylycz M et al (2004) Genome-wide identification of DNaseI hypersensitive sites using active chromatin sequence libraries. Proc Natl Acad Sci USA 101:4537–4542
6. Sabo PJ, Kuehn MS, Thurman R et al (2006) Genome-scale mapping of DNase I sensitivity in vivo using tiling DNA microarrays. Nat Methods 3:511–518
7. Crawford GE, Holt IE, Whittle J et al (2006) Genome-wide mapping of DNase hypersensitive sites using massively parallel signature sequencing (MPSS). Genome Res 16:123–131
8. Boyle AP, Davis S, Shulha HP et al (2008) High-resolution mapping and characterization of open chromatin across the genome. Cell 132 (2):311–322
9. Thurman RE, Rynes E, Humbert R et al (2012) The accessible chromatin landscape of the human genome. Nature 489 (7414):75–82.
10. Kelly TK, Liu Y, Lay FD et al (2012) Genome-wide mapping of nucleosome positioning and DNA methylation within individual DNA molecules. Genome Res 22(12):2497–2506
11. Krebs AR, Imanci D, Hoerner L, Gaidatzis D et al (2017) Genome-wide Single-Molecule Footprinting Reveals High RNA Polymerase II Turnover at Paused Promoters. Mol Cell 67(3):411–422.e4
12. Shipony Z, Marinov GK, Swaffer MP et al (2018) Long-range single-molecule mapping of chromatin accessibility in eukaryotes. bioRxiv 504662
13. Wang Y, Wang A, Liu Z et al (2019) Single-molecule long-read sequencing reveals the chromatin basis of gene expression. Genome Res 29(8):1329–1342
14. Aughey GN, Estacio Gomez A, Thomson J et al (2018) CATaDa reveals global remodelling of chromatin accessibility during stem cell differentiation in vivo. Elife 7:pii: e32341
15. Chereji RV, Eriksson PR, Ocampo J, Clark DJ (2019) DNA accessibility is not the primary determinant of chromatin-mediated gene regulation. bioRxiv 639971
16. Ponnaluri VKC, Zhang G, Estéve PO et al (2017) NicE-seq: high resolution open chromatin profiling. Genome Biol 18(1):122
17. Umeyama T, Ito T (2017) DMS-Seq for in vivo genome-wide mapping of protein-DNA interactions and nucleosome centers. Cell Rep 21(1):289–300
18. Timms RT, Tchasovnikarova IA, Lehner PJ (2019) Differential viral accessibility (DIVA) identifies alterations in chromatin architecture through large-scale mapping of lentiviral integration sites. Nat Protoc 14(1):153–170
19. Buenrostro JD, Giresi PG, Zaba LC et al (2013) Transposition of native chromatin for fast and sensitive epigenomic profiling of open chromatin, DNA-binding proteins and nucleosome position. Nat Methods 10:1213–1218
20. Buenrostro JD, Wu B, Litzenburger UM et al (2015) Single-cell chromatin accessibility reveals principles of regulatory variation. Nature 523(7561):486–490
21. Cusanovich DA, Daza R, Adey A et al (2015) Multiplex single cell profiling of chromatin accessibility by combinatorial cellular indexing. Science 348(6237):910–914
22. ENCODE Project Consortium (2012) An integrated encyclopedia of DNA elements in the human genome. Nature 489:57–74
23. Amemiya HM, Kundaje A, Boyle AP (2019) The ENCODE Blacklist: Identification of Problematic Regions of the Genome. Sci Rep 9(1):9354
24. Langmead B, Trapnell C, Pop M et al (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. Genome Biol 10:R25
25. Langmead B, Salzberg SL (2012) Fast gapped-read alignment with Bowtie 2. Nat Methods 9:357–359
26. Li H, Handsaker B, Wysoker A et al (2009) The Sequence Alignment/Map format and SAMtools. Bioinformatics 25:2078–2079
27. Feng J, Liu T, Qin B et al (2012) Identifying ChIP-seq enrichment using MACS. Nat Protoc 7:1728–1740

28. Li Q, Brown J, Huang H et al (2011) Measuring reproducibility of high-throughput experiments. Ann Appl Stat 5:1752–1779
29. Kuhn RM, Haussler D, Kent WJ (2013) The UCSC genome browser and associated tools. Brief Bioinform 14:144–161
30. Kent WJ, Zweig AS, Barber G et al (2010) BigWig and BigBed: enabling browsing of large distributed datasets. Bioinformatics 26:2204–2207
31. Love MI, Huber W, Anders S (2014) Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biol 15(12):550
32. Schep AN, Wu B, Buenrostro JD, Greenleaf WJ (2017) chromVAR: inferring transcription-factor-associated accessibility from single-cell epigenomic data. Nat Methods 14:975–978
33. Ramírez F, Ryan DP, Grüning B et al (2016) deepTools2: a next generation web server for deep-sequencing data analysis. Nucleic Acids Res 44(W1):W160–W165
34. Quinlan AR, Hall IM (2010) BEDTools: a flexible suite of utilities for comparing genomic features. Bioinformatics 26(6):841–842
35. Liao Y, Smyth GK, Shi W. (2014) featureCounts: an efficient general purpose program for assigning sequence reads to genomic features. Bioinformatics 30(7):923–930
36. Bolger AM, Lohse M, Usadel B (2014) Trimmomatic: a flexible trimmer for Illumina sequence data. Bioinformatics 30 (15):2114–2120
37. Martin M (2011) Cutadapt removes adapter sequences from high-throughput sequencing reads. EMBnet J 17(1):10–12
38. Li H, Durbin R (2009) Fast and accurate short read alignment with Burrows-Wheeler transform. Bioinformatics 25:1754–1760
39. Corces MR, Trevino AE, Hamilton EG et al (2017) An improved ATAC-seq protocol reduces background and enables interrogation of frozen tissues. Nat Methods 14:959–962
40. Hazkani-Covo E, Zeller RM, Martin W (2010) Molecular poltergeists: mitochondrial DNA copies (numts) in sequenced nuclear genomes. PLoS Genet 6(2):e1000834
41. Marinov GK, Wang YE, Chan D, Wold BJ (2014) Evidence for site-specific occupancy of the mitochondrial genome by nuclear transcription factors. PLoS One 9(1):e84713
42. Smith DR, Keeling PJ (2015) Mitochondrial and plastid genome architecture: reoccurring themes, but significant differences at the extremes. Proc Natl Acad Sci USA 112 (33):10177–10184
43. Landt SG, Marinov GK, Kundaje A et al (2012) ChIP-seq guidelines and practices of the ENCODE and modENCODE consortia. Genome Res 22(9):1813–1831
44. Daley T, Smith AD (2013) Predicting the molecular complexity of sequencing libraries. Nat Methods 10(4):325–327
45. Marinov GK, Kundaje A, Park PJ, Wold BJ (2014) Large-scale quality analysis of published ChIP-seq data. G3 (Bethesda) 4(2):209–223
46. Tarbell ED, Liu T (2019) HMMRATAC: a Hidden Markov ModeleR for ATAC-seq. Nucleic Acids Res pii: gkz533
47. McCarthy DJ, Chen Y, Smyth GK (2012) Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation. Nucleic Acids Res 40 (10):4288–4297
48. Ritchie ME, Phipson B, Wu D, Hu Y, Law CW, Shi W, Smyth GK (2015) Limma powers differential expression analyses for RNA-sequencing and microarray studies. Nucleic Acids Res 43(7):e47
49. van der Maaten LJP, Hinton GE (2008) Visualizing high-dimensional data using t-SNE. J Mach Learn Res 9:2579–2605
50. Becht E, McInnes L, Healy J et al (2018) Dimensionality reduction for visualizing single-cell data using UMAP. Nat Biotechnol 37:38–44
51. Li Z, Schulz MH, Look T et al (2019) Identification of transcription factor binding sites using ATAC-seq. Genome Biol 20(1):45
52. Hesselberth JR, Chen X, Zhang Z et al (2009) Global mapping of protein-DNA interactions in vivo by digital genomic footprinting. Nat Methods 6(4):283–289
53. Neph S, Stergachis AB, Reynolds A et al (2012) Circuitry and dynamics of human transcription factor regulatory networks. Cell 150:1274–1286
54. Neph S, Vierstra J, Stergachis AB et al (2012) An expansive human regulatory lexicon encoded in transcription factor footprints. Nature 489:83–90
55. Stergachis AB, Neph S, Reynolds A et al (2013) Developmental fate and cellular maturity encoded in human regulatory DNA landscapes. Cell 154:888–903
56. Pique-Regi R, Degner JF, Pai AA et al (2011) Accurate inference of transcription factor binding from DNA sequence and chromatin accessibility data. Genome Res 21(3):447–455
57. Cuellar-Partida G, Buske FA, McLeay RC et al (2012) Epigenetic priors for identifying active transcription factor binding sites. Bioinformatics 28(1):56–62

58. Piper J, Elze MC, Cauchy P et al (2013) Wellington: a novel method for the accurate identification of digital genomic footprints from DNase-seq data. Nucleic Acids Res 41(21): e201
59. Sherwood RI, Hashimoto T, O'Donnell CW et al (2014) Discovery of directional and nondirectional pioneer transcription factors by modeling DNase profile magnitude and shape. Nat Biotechnol 32(2):171–178
60. He HH, Meyer CA, Hu SS et al (2014) Refined DNase-seq protocol and data analysis reveals intrinsic bias in transcription factor footprint identification. Nat Methods 11:73–78
61. Sung MH, Guertin MJ, Baek S, Hager GL. 2014. DNase footprint signatures are dictated by factor dynamics and DNA sequence. Mol Cell 56(2):275–285
62. Gusmao EG, Dieterich C, Zenke M, Costa IG (2014) Detection of active transcription factor binding sites with the combination of DNase hypersensitivity and histone modifications. Bioinformatics 30(22):3143–3151
63. Raj A, Shim H, Gilad Y et al (2015) msCentipede: modeling heterogeneity across genomic sites and replicates improves accuracy in the inference of transcription factor binding. PLoS One 10(9):e0138030
64. Yardimci GG, Frank CL, Crawford GE, Ohler U (2015) Explicit DNase sequence bias modeling enables high-resolution transcription factor footprint detection. Nucleic Acids Res 42 (19):11865–11878
65. Gusmao EG, Allhoff M, Zenke M, Costa IG (2016) Analysis of computational footprinting methods for DNase sequencing experiments. Nat Methods 13(4):303–309
66. Quach B, Furey TS (2017) DeFCoM: analysis and modeling of transcription factor binding sites using a motif-centric genomic footprinter. Bioinformatics 33(7):956–963
67. Baek S, Goldstein I, Hager GL (2017) Bivariate genomic footprinting detects changes in transcription factor activity. Cell Rep 19 (8):1710–1722
68. Karabacak Calviello A, Hirsekorn A, Wurmus R et al (2019) Reproducible inference of transcription factor footprints in ATAC-seq and DNase-seq datasets using protocol-specific bias modeling. Genome Biol 20(1):42
69. Sung MH, Baek S, Hager GL (2016) Genome-wide footprinting: ready for prime time? Nat Methods 13(3):222–228
70. Vierstra J, Stamatoyannopoulos JA (2016) Genomic footprinting. Nat Methods 13 (3):213–221
71. Mortazavi A, Leeper Thompson EC, Garcia ST et al (2006) Comparative genomics modeling of the NRSF/REST repressor network: from single conserved sites to genome-wide repertoire. Genome Res 16(10):1208–1221
72. Johnson DS, Mortazavi A, Myers RM, Wold B (2007) Genome-wide mapping of in vivo protein-DNA interactions. Science 316 (5830):1497–1502
73. Weirauch MT, Yang A, Albu M et al (2014) Determination and inference of eukaryotic transcription factor sequence specificity. Cell 158:1431–1443
74. Grant CE, Bailey TL, Noble WS (2011) FIMO: scanning for occurrences of a given motif. Bioinformatics 27:1017–1018
75. Henikoff JG, Belsky JA, Krassovsky K et al (2011) Epigenome characterization at single base-pair resolution. Proc Natl Acad Sci USA 108:18318–18323
76. Fu Y, Sinha M, Peterson CL, Weng Z (2008) The insulator binding protein CTCF positions 20 nucleosomes around its binding sites across the human genome. PLoS Genet 4:e1000138
77. Schep AN, Buenrostro JD, Denny SK et al (2015) Structured nucleosome fingerprints enable high-resolution mapping of chromatin architecture within regulatory regions. Genome Res 25:1757–1770

Chapter 11

Genome-Wide Noninvasive Prenatal Diagnosis of SNPs and Indels

Tom Rabinowitz and Noam Shomron

Abstract

Noninvasive prenatal diagnosis (NIPD) is an emerging field, that enables testing for diseases in the fetus with no risk to the pregnancy, compared to invasive methods (e.g., amniocentesis). The procedure is based on the presence of fetal DNA within the mother's plasma cell-free DNA (cfDNA). Today, NIPD is performed for chromosomal abnormalities (e.g., Down syndrome) and some large deletions/duplications. It is also available for point mutations but is limited for one mutation or up to several genes simultaneously. Genome-wide detection of fetal point mutations was presented in a few studies, and the first software tool for this task, *Hoobari*, has recently become available. Here we describe the necessary steps in genome-wide noninvasive fetal genotyping, including examples using the *Hoobari* software. We discuss the various materials, software, computational infrastructure, and samples required for this analysis. Genome-wide analysis of point mutations in the fetus is not widely studied, albeit much space for algorithmic improvements exists. Here we suggest practical solutions for challenges along the process. Our work assists bioinformaticians in accessing NIPD data analysis and can eventually be utilized for other cfDNA-related fields.

Key words Noninvasive prenatal diagnosis, NIPD, Fetal, Cell-free DNA, cfDNA, cffDNA, Cell-free fetal DNA, Hoobari, Circulating tumor DNA, ctDNA, Liquid biopsy

1 Introduction

1.1 Noninvasive Prenatal Diagnosis

Diagnosis of diseases in the fetus during pregnancy, also known as prenatal diagnosis, is an approach to detect medical conditions in the earliest stage of life. Like any process of diagnosis, it typically begins with a history and physical examination of the mother. Essential diagnostic procedures during pregnancy frequently require the involvement of imaging (ultrasound) and genetic tests for congenital disorders. Attaining genetic material from the fetus is challenging; until recently, it could only be accomplished using invasive methods such as chorionic villus sampling (CVS) or amniocentesis. As such, these methods pose a risk of miscarriage and other pregnancy complications [1, 2] and thus only offered in specific time windows during pregnancy, and not before

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_11,

10–12 weeks of gestation [3]. In recent years, noninvasive prenatal diagnosis (NIPD) methods have become available. These methods usually rely on cell-free DNA (cfDNA) in the maternal plasma as. A small amount of this DNA, roughly 10% in the end of the first trimester, is fetal cfDNA (cffDNA) [4].

NIPD is now a rapidly growing field, accelerated by academia, national healthcare systems, and industry. The first available utilizations of NIPD were for chromosome-level genetic phenomena, including chromosomal aneuploidies such as down syndrome, or early-stage sex determination [5–8]. This remains the most prevalent use of NIPD. Recent studies show that NIPD can replace current screening tests not only in women that are in increased risk for Down syndrome but also in the wider population [9]. Several studies have also presented the success of NIPD in detecting microdeletions and duplications [10–13]. The last challenge of NIPD is the detection of point mutations, that is, single nucleotide polymorphisms (SNPs). The major obstacle with such mutations and variants is the deep coverage that is required in each genomic locus. The main approaches to this problem rely on either digital PCR, which is a highly accurate method for quantification [14], or targeted next generation sequencing (NGS) [15]; in some approaches, these are combined with haplotyping of the parents [16]. These approaches offer a reliable and accurate solution in cases where there are a few tested mutations or genes, hence they are already widely used in several healthcare systems [16].

1.2 Genome-Wide Analysis of Rare Diseases

Another field of research that has emerged approximately at the same time as NIPD, is NGS-based genome-wide analysis of rare diseases. During this process, either whole exome sequencing (WES) or whole genome sequencing (WGS) are first performed. The sequenced reads are then mapped to the human reference genome, and a large number of variants are detected in the process of variant calling. To assess whether these variants have a deleterious effect they undergo quality-control, whereby they are analyzed, interpreted and filtered based on various features. Usually, WES/WGS analyses are performed in families with a known medical condition or phenotype that follows Mendelian inheritance. The purpose of finding the causative mutation in rare diseases beyond diagnosis, could be for research, or for enabling in vitro fertilization (IVF) with preimplantation genetic diagnosis (PGD) in future pregnancies. Since its introduction, WES/WGS analysis has been responsible for the discovery of a large amount of disease-causing mutations.

WES/WGS can also be used for prenatal diagnosis, by sequencing the fetal DNA from the amniotic fluid or following CVS. Such sequencing enables the highest possible resolution—at the level of a single nucleotide—for examining the fetal DNA [17]. It was shown in several studies that prenatal WES can be a strong tool for

diagnosis of diseases in the fetus, even superior to traditional methods of genetic diagnosis [18–21].

1.3 Genome-Wide NIPD of Monogenic Disorders

The ultimate goal of NIPD is the detection of all genetic abnormalities in the fetus, including genome-wide detection of point mutations that cause monogenic diseases. In noninvasive fetal WGS, there are three possible types of mutations: maternal-, paternal- or double-inherited. Paternal-inherited mutations are the easiest to detect, as a paternal allele will be a foreign allele in the plasma. Maternal- and double-inherited are harder to predict, because the maternal allele is present in the plasma even when it was not inherited. Several attempts to achieve genome-wide detection of maternal alleles have been performed [22–24]. These proof-of-concept studies showed that it is possible to perform genome-wide NIPD of monogenic diseases in the fetus. Detection of maternal-inherited mutations was addressed using different approaches, based on either haplotyping of the mother, WGS with an ultra-deep coverage, or WES that enables an even deeper coverage. However, these attempts had various limitations. For instance, the tests were performed in very late stages of pregnancy, which are less clinically relevant. Furthermore, some mutation types were not evaluated, such as double-inherited mutations and small insertions-deletions (indels), and many cases of maternal-inherited mutations were skipped. In regards to haplotyping, high-throughput techniques for genome-wide haplotyping have limited use given their low availability and reduced resolution [16, 25, 26]. To date, no method has been able to suggest an accurate genome-wide solution for genome-wide detection of monogenic diseases in the fetus.

Recently, a different approach for NIPD of monogenic diseases was presented (Fig. 1). This approach differed in that it treated the analysis as a unique case of WES/WGS analysis [27]. Here, the cfDNA was analyzed using a dedicated variant-caller, named *Hoobari* [27], which incorporated information from the parental DNA and the cfDNA, including the size differences between fetal and maternal cfDNA fragments. This method was used for samples taken in the first trimester of pregnancy and showed to be highly accurate in noninvasively sequencing the fetal genome. It was further successfully applied over double-inherited mutations and indels for the first time. We also showed, as a proof of concept, that the accuracy can even be further improved using an additional optimization step based on a machine learning model. This model was created using previously analyzed and verified results of our analyses, to learn their error patterns and correct these errors in future analyses.

1.4 Chapter Outline

It has been generally argued that the required sequencing depth of coverage is the main obstacle in achieving noninvasive prenatal

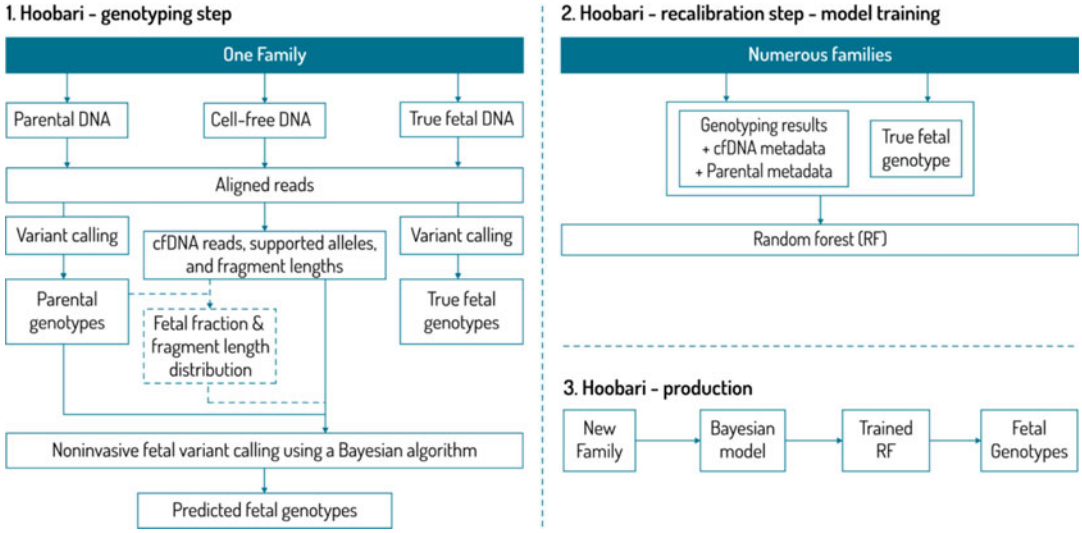


Fig. 1 Workflow for noninvasive fetal genotyping study. (1) Each family consists of four samples: two parental genomic DNA samples, maternal cfDNA and pure fetal DNA that is used as the gold standard. All samples are mapped, resulting in reads that are aligned to the human reference genome. The pure fetal sample and the parental samples are genotyped using a standard variant caller. The parental genotypes are utilized for calculating the fetal fraction and the fragment length distribution, which, together with the cfDNA in each position of interest, are used for genotyping of the fetus. (2) Using many families that have undergone noninvasive fetal genotyping, a machine learning model is trained. The features are metadata of the parental and cfDNA variants, that was not utilized in the Bayesian algorithm. The labels are taken from the genotyped true fetal sample. (3) The trained machine learning model is used as a step that refines the variant calls

WGS. Nevertheless, as was the case using *Hoobari*, there are still many potential bioinformatic improvements for NIPD of monogenic diseases, even without a deeper coverage. In general, technological limitations can be overcome with computational and algorithmic aspects of an analysis. Addressing the challenge of NIPD of monogenic diseases in terms of variant calling, as *Hoobari* does, may assist in further exploring this area. This paradigm means that a unique variant detection software and algorithm are used here for noninvasive fetal genotyping. The algorithm utilizes all the genetic information that is available during pregnancy, from both parental DNA and cfDNA. The analysis is executed through a straightforward bioinformatical pipeline. Standardization and modularity enable constant introduction of improvements to different parts of the process. This way, a generic infrastructure is formed, that enables bioinformaticians to explore and improve the field of genome wide NIPD of monogenic disorders.

Here we describe in detail the different steps in a noninvasive fetal genotyping experiment, mainly from a bioinformatical perspective along with a technological one. The relevant materials for steps prior to the computational analysis are briefly reviewed, that is, sample collection, library preparation, and sequencing

techniques. Bioinformatical tools that are used in the pipeline are listed. The main section then describes the methods that comprise the pipeline for noninvasive fetal genotyping. We attempt to explain every step, including algorithmic ones, in simple terms while bringing examples from *Hoobari*. A special emphasis is put on outlining considerations, tips, and tricks that are relevant in such studies; the most important ones are listed together at the end.

2 Materials

2.1 Preanalysis Materials

The maternal plasma is a unique sample with specific characteristics that should be taken into account when attempting to perform NIPD. There are various considerations for cfDNA collection, extraction, enrichment, library preparation, and sequencing. Many technologies aim to solve issues and challenges that arise during each step. Some of these considerations are discussed here in general, however for detailed explanations see other publications [28, 29].

First, the maternal blood is collected using several ethylenediaminetetraacetic acid (EDTA) tubes. In order to reach the amount of DNA that is required by the library preparation protocol, it is important to give attention to the plasma volume and its measured cfDNA concentration. The volume depends on the number of tubes collected, and these should be filled to the top, unlike in blood count. Second, the blood sample is separated to red blood cells, white blood cells, and plasma. This is achieved by centrifugation at 4 °C for 10 minutes at 1600 × *g*. The transparent layer, the plasma, is then separated and centrifuged again at 16,000 × *g* for 10 min at room temperature, to remove any residual cells. The first centrifugation is relatively slow, to prevent hemolysis and contaminating of the plasma with nuclear maternal DNA. Such DNA further dilutes the amount of fetal DNA within the plasma, and also has different characteristics from cfDNA, which might interfere with the analysis. In general, it is important that the maternal blood samples are carefully handled, and centrifugation performed immediately after blood collection.

Extraction of cfDNA from the plasma is performed using dedicated purification kits. Extra purification and cleaning steps might be required if the measured DNA concentration seems low, relative to the amount needed for the library preparation step. Enrichment of fetal-derived DNA is also possible through several protocols; this, however, results in loss of DNA. Therefore, it is yet to be tested in the context of genome-wide fetal NIPD of monogenic diseases. As with any kit, it is recommended to first use those that have already shown reliable performance, and to test them on one or two samples only. If possible, a more comprehensive comparison should be performed. It is also recommended to use samples from

previously published studies which are known to be technically sound and could be compared to.

Before sequencing, the purified cfDNA requires library preparation. In this context, it is important not to alter the natural fragmentation of the cfDNA. During library preparation, PCR amplification cycles are sometimes required, and if WES is performed, more cycles are performed. Even with "PCR-free" protocols for WGS, several cycles might be required in the settings of very deep sequencing and small amounts of DNA. PCR amplification introduces errors, higher rates of duplicate reads, and insert biases toward certain reads. Considerations regarding sequencing method, that is, WGS vs. WES, and the effect of the depth of coverage are further discussed below.

Finally, the purified cfDNA sample is sequenced. As with any NGS experiment, it is important to choose a platform that can handle the required coverage at a reasonable turnaround time. It is recommended to first sequence part of the prepared library, to check its quality, as well as the fetal DNA fraction and length distribution. If the results are satisfactory, the rest of the library can be sequenced to achieve the desired coverage.

The effect of these biological and technical steps on the results is crucial. However, there are no widely accepted guidelines for this part of the procedure. The field of NIPD is still in its infancy and, together with liquid biopsies in the context of cancer, holds a huge potential for clinical and research technologies. It is recommended to be constantly aware of new solutions that are invented, developed, and marketed.

2.2 Computational Tools

Various tools are required for the analysis of cfDNA. Some are general tools or software that are is regularly used for NGS analysis. *Hoobari* is the only software tool that is specific to the task of noninvasively genotyping a fetus, as a basis for genome wide NIPD of monogenic diseases. General tools can be replaced by others that fulfill a similar function, but *Hoobari* is the only one available for its task. *Hoobari* currently depends on Freebayes, but *Hoobari's* code is freely available and can be edited to work with other variant callers as well (Table 1).

3 Methods

3.1 Sample Selection

3.1.1 Family Trios

Theoretically, NIPD can rely solely on a blood sample from the mother. The blood can then be centrifuged, with the cfDNA extracted from the plasma and maternal genomic DNA extracted from the white blood cells layer, also known as the buffy coat.

Adding a paternal sample to the analysis has several advantages, the first and most obvious one being the improved sensitivity for detection of paternal-derived mutations. Second, it enables

Table 1
Computational tools used during genome-wide NIPD of monogenic diseases

Tool/resource	Function	References	Link
BWA-MEM	Read alignment	[30]	http://bio-bwa.sourceforge.net
Samblaster	Deduplication	[31]	https://github.com/GregoryFaust/samblaster
Samtools	BAM manipulation	[32]	http://www.htslib.org
Sambamba	BAM manipulation	[33]	https://lomereiter.github.io/sambamba
Freebayes	Variant calling	[34]	https://github.com/ekg/freebayes
Hoobari	Noninvasive fetal variant calling	[27]	https://github.com/nshomron/hoobari

accurate fetal DNA fraction calculation using genomic loci in which both parents are homozygous for different alleles, since fetal-derived DNA fragments can be identified in the maternal plasma in these loci. An accurate estimation of the fetal fraction improves the sensitivity of every aspect of the diagnosis, and can be utilized for other analyses, such as NIPD of chromosomal abnormalities and microdeletions/duplications. Third, the collection of fetal-derived DNA fragments is used for the calculation of the fragment-length distribution, which is used for an even more accurate mutation detection (characteristics other than the lengths can be utilized similarly, but such features are still under research). Finally, when both the maternal and paternal genotypes are available, it is possible to calculate the prior probability for the fetal genotype using Mendelian laws.

For the above reasons, currently the optimal approach to perform NIPD of monogenic diseases is using family trios, that is, maternal genomic DNA from the mother's buffy coat, paternal DNA from the father's blood or saliva, and fetal and maternal cfDNA from the mother's plasma.

3.1.2 *True Fetal Sample*

The last required sample is pure fetal DNA. This is required for research purposes, to enable a comparison of the predicted fetal genotypes with the gold standard. A fetal sample can also be relevant in clinical circumstances, as confirmation of a positive result when the noninvasive procedure is used as a screening test. Several sources for a true fetal sample are available; they usually include chorionic villus sampling (CVS), amniocentesis, and umbilical cord blood. These tests differ in their accuracy and risks. Placental mosaicism, for instance, can cause false positive results in NIPD, but also in CVS [35]. As well, the risk of miscarriage is generally higher in CVS than amniocentesis [1]. Amniocentesis is the most accurate test, but it is not available in early stages of pregnancy, making CVS the preferred test in this time.

Therefore, the accuracy assessment of an NIPD analysis depends on the selected source of the true fetal sample. A possible approach in such studies is to use the best available gold standard depending on the week of gestation in which the cfDNA sample was drawn. In early stages, for example, it would be CVS, which is performed at 10–12 weeks of gestation and in later stages, amniocentesis or a postnatal sample (e.g., umbilical cord blood) can be used as the true reference. However, since women that take part in NIPD studies are often recruited when they perform CVS or amniocentesis, any of the aforementioned sources can be used at any stage. Nonetheless, the differences between these sources should be taken into account and discussed accordingly when performing NIPD studies.

3.2 Biological and Technical Considerations

The accuracy of NIPD of monogenic diseases is affected by several parameters that are applicable for any NIPD procedure. These could be (1) biological concerns, such as the amount of the fetal DNA fraction, or (2) technical concerns, such as the depth of coverage. Some features of the analysis, such as the specific fragment length distribution, affect the accuracy on both the biological and technical aspects.

3.2.1 Fetal Fraction

The rate of fetal-derived DNA within the total amount of cfDNA in the maternal plasma is termed the "fetal fraction." The cfDNA levels in plasma are relatively low, and the fetal cfDNA levels are even lower, as they originate only from the placenta. Higher fetal fraction improves the accuracy of the prediction of the fetal genotype. The fetal fraction at 10–12 weeks of gestation is about 10% [36]. This is a major challenge in any NIPD analysis, since less material is available for sequencing.

Different biological factors affect the fetal fraction during pregnancy. First, the placenta itself grows throughout the pregnancy, as it provides nutrients to the growing fetus. As a result, the fetal fraction rises as the pregnancy progresses. Furthermore, the fetal fraction increases with fetal crown–rump length, smoking, pregnancy protein markers, and trisomy 21 karyotype, and decreases with increased maternal weight. Aside note fetal fraction varies among different ethnic groups [36].

The plasma sample can be enriched for fetal fragments, prior to sequencing, using several laboratory techniques that filter out longer fragments [28, 37, 38]. Such methods are not used in the analysis described here, as they might lower the sensitivity [39]. Moreover, these technical steps complicate the process, making it less approachable for researchers. Here, the fragment length information is utilized within the algorithm, such that filtering of fragments is avoided (see below). It is still possible to use an enriched sample, and this should not interfere with the analysis we describe here. Nevertheless, it would require careful execution and

possibly some specific adjustments for the high fetal fraction, since the algorithm was not tested with these settings.

As NIPD of monogenic disorders depends on the fetal fraction, an accurate fetal fraction calculation is crucial for optimal results. The fetal fraction can be calculated in numerous methods, differing by their biological basis and statistical methods, but all of them are based on the identification of fetal-specific fragments. Some available methods are based on Y chromosome fragments, while others are sex-independent but require the paternal genotypes [40]. The method used here is a straightforward calculation using the paternal information, without statistical corrections and technical or biological considerations. This performs well enough for the comparison between different factors in the algorithm, but eventually, in order to improve the overall accuracy, we recommend researchers to try to optimize the fetal fraction calculation, using other methods or even a combination of methods (*see* **Note 1**).

3.2.2 Fragment Length Distribution (Fig. 2)

The cfDNA in the plasma appears in fragments with varying lengths. The length distribution of fetal-derived reads is different from that of the maternal-derived reads. The method described here takes advantage of this phenomenon (further discussed below). The length distribution depends on technical and biological factors, and this can affect the consistency of the algorithm's accuracy. For example, as mentioned in Subheading 2, if the maternal blood samples are not carefully handled, genomic DNA can be released from blood cells and these can affect the distribution. PCR amplification and pull-down kits can be biased toward specific fragments rather than others, based on their lengths or other physical characteristics. On the other hand, the length differences do not seem to change throughout the pregnancy or vary among women. The length distributions are calculated empirically for each new family, but if the differences become too subtle due to technical reasons, the lengths might not assist as expected. For this reason, when using the length differences for NIPD of monogenic diseases, it is important to optimize the technical stages of sample collection, library preparation, and sequencing, and to assure that the length distributions are consistent among experiments.

3.2.3 Depth of Coverage

Any NGS-based DNA analysis depends on the depth of the coverage. The coverage at a certain position represents the number of times that the nucleotide was read. Deeper coverage means more evidence of a certain nucleotide in a given genomic locus, which results in better accuracy. An even deeper coverage than usual is required for NIPD of monogenic diseases, since the fetal fraction can be very low. If for example, 30× is the coverage of choice for WGS (i.e., an average of 30 coverage for each nucleotide), and the fetal fraction is 10%, the required depth in the case of NIPD will

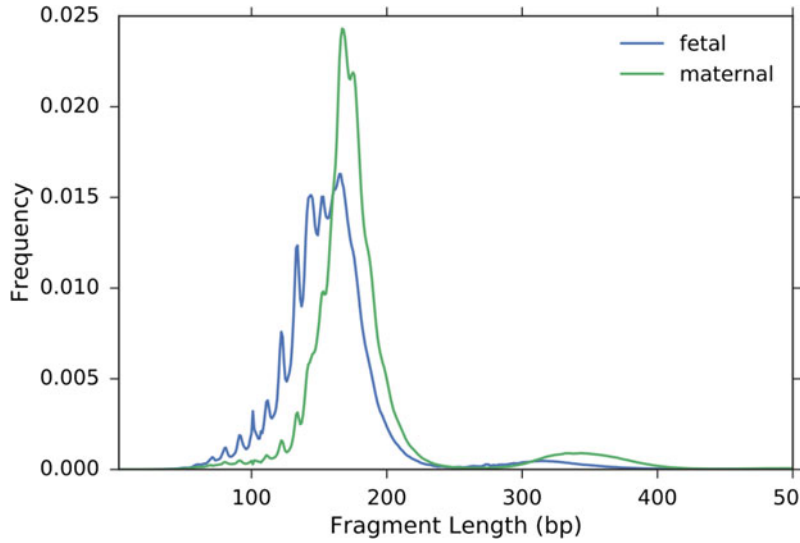


Fig. 2 The length distribution of fetal-derived fragments (blue) and maternal derived fragments (green). The fetal fragments are shorter and present a more distinguished degradation pattern

theoretically be at least 300×. A lower fetal fraction requires even deeper coverages but sequencing to such depth is currently costly. This makes the sequencing depth a key limitation in brining genome-wide NIPD of monogenic disorders to the market. There are several possible ways to deal with this hurdle. For example, only certain genomic regions could be sequenced, for example the coding regions, but this has several disadvantages, as discussed in Subheading 3.2.4. Another option is to try to utilize as much information as possible from the given fragments that cover each position. For example, in the method described here, the fragment lengths are used in order to achieve better accuracy, and other information can be utilized as well. Eventually, both approaches can be integrated to achieve the optimal results at an affordable cost.

3.2.4 Exome vs Genome Sequencing

Genome-wide NIPD of monogenic diseases can refer to WGS, WES, or even narrower NGS panels that represent genes of clinical relevance. This sequencing methods differ by the sequenced regions, and also by their protocols, common errors and biases. WGS is the most accurate method, even when referring only to coding regions. The required average coverage is approximately 30×, which is not as deep as when using WES. This is possible due to the fact that the coverage in WGS is a lot more uniform. As a result, WGS can be performed using PCR-free protocols, which do not require a PCR amplification step during library preparation. WGS has several limitations as well; it is currently costly and poses somewhat of a challenge in terms of computational infrastructure. Also, although more accurate, it involves sequencing of many parts

of the genome yet to be interpreted and understood. This results in many variants of unknown significance (VUS). For these reasons, in standard variant calling, WES analysis is still most common. Often, the coverage required for WES analysis is 100×, but compared with WGS, only ~2% of the genome is sequenced.

In the context of noninvasive fetal genotyping, the overall required coverage is much deeper than 30×. Therefore, it is sometimes necessary to run two to three PCR amplification cycles prior to sequencing. Even after this, a coverage of 300× might not suffice, pending on the fetal fraction. On the other hand, due to the mandatory extra steps of PCR amplification and the use of pull-down kits, WES and other NGS panels that are based on a similar technology increase the error rate, insert different sorts of bias and inflate the level of duplicate reads. These biases disrupt the accuracy of the fetal genotyping considerably and should be taken into consideration.

3.3 Computational Pipeline for Noninvasive Fetal Variant Calling

In this section, the required steps in the workflow of noninvasive fetal genotyping are reviewed in detail. Initially, sequence reads of all samples are mapped to reference genome, and variant calling is performed over the parental genomic DNA samples. Then, the cfDNA is scanned, to identify candidate positions and extract information about the reads that cover their genomic location. In the same step, cfDNA reads in positions where the parents are homozygous for different alleles are processed, to enable calculation of the fetal fraction and the fetal and maternal fragment length distributions. Finally, a Bayesian genotyping algorithm is applied on the candidate positions, resulting in a genotype at each none-reference parental position. To compare the results to the gold standard, the true fetal sample also undergoes mapping and variant calling.

3.3.1 Alignment, Deduplication, Indel Realignment

Sequencing the cohort of samples results in raw sequence read files, in FASTQ format. These raw reads are then mapped (aligned) to the reference human genome, resulting in BAM files, which contain aligned reads. There are several available mapping tools, and numerous studies that compare their abilities [41]. Here we used BWA-MEM, as mentioned in Subheading 3.

After mapping all of the samples to the reference genome, it is important to mark duplicate reads in all of them. Duplicate reads arise from the DNA amplification steps before sequencing the samples, as explained in Subheading 3.2.4. They affect the results of downstream analysis by distorting the read counts and allele counts at each candidate variant. This can affect the estimation of the fetal fraction, fragment length distribution, cause erroneous genotypes or affect the quality of the correctly called variants. Most of the issues caused by duplicate reads occur both in cfDNA analysis, which includes a mixed sample of two sources (mother and

fetus), and in the common settings of variant calling, in which the sample is from a single source. The plasma sample, however, usually requires extra amplification steps, due to the limited amount of DNA in it. Therefore, it is prone to high duplicate reads rate. Since the fetal genotypes are eventually predicted using this sample, marking duplicate reads in it is essential. It is pivotal to assure that every downstream step ignores duplicate reads. Most variant callers do so by default, but if an external code is implemented, this should be taken into account.

The software tool used here for marking duplicate reads is Samblaster. Like most algorithms, it is based on read pairs that share the same genomic coordination. Theoretically, duplicate reads can arise from two identical DNA molecules, which are not a result of amplification. DNA is usually being fragmented during the library preparation step, such that identical molecules are not likely, and they become negligible. However, cfDNA is already fragmented by maternal enzymes. Recently it has been shown that fetal and maternal fragments tend to originate from certain positions more than from others [26]. This means that at least some of the duplicate reads should not be ignored as this may cause errors. Given its probably minor effect, this issue is not dealt with in the analysis described here.

In addition to mapping and marking duplicate reads, the reads should also be sorted by coordinates and an index file should be created. These steps can be performed using tools such as Samtools, Sambamba, and others.

Additional steps can be applied over the mapped, deduplicated, sorted, and indexed files. These include realignment or local reassembly around known or candidate indels, as well as base quality recalibration. These steps require extensive computational resources, and today they are usually performed during variant calling by the variant calling algorithm itself. This change of approach, however, is based on experiments that took place in the settings of standard variant calling, not NIPD. When deep cfDNA samples and a different variant calling algorithm are used, such steps might still improve the accuracy.

Prior to and after each computational process described above, it is important to assess quality measures of the reads, such as depth of coverage, base and mapping qualities. The aforementioned steps are not easy to implement when such deep sequencing is used, due to limited computational infrastructure. A few solutions for this are available, for instance, some of the steps could be performed by an external sequencing service provider (*see* **Note 2**).

3.3.2 Step 1: Parental Variant Calling

After all the read alignment files are ready, it is possible to begin analyzing the variants and mutations in the data. The first step is to genotype the parents. This step is based on existing software for

variant calling (*see* **Note 3**). Here we use Freebayes, but other software could be used. The variant calling software tool requires the parental read alignment files and a path to the indexed reference genome file. The output is a variant call format (VCF) file, containing the parental genotypes. In order for a genotype to appear in the VCF, at least one of the parents should have at least one alternate allele in the given locus, that is, at least one parent should be either heterozygous or homozygous for the alternate allele. Loci in which both parents are homozygous for the reference allele can also appear if there were some evidence for a variant there, prior to the exact probability calculation by the variant caller. More than 97% of the variants are biallelic among the general population [42]. - Therefore, if an alternate allele appears in both parents, either in heterozygous or homozygous form, it will usually be the same one. Other cases, in which the mother and the father have different alternate alleles, require adjustment of the Bayesian algorithm used downstream.

At this stage, it is important to filter the parental variant calls, and keep only ones that were determined with high confidence. Filtering out can be based on depth of coverage, for example. It is also important to check the default settings of the variant caller, as some variant callers perform basic filtering by default.

3.3.3 Step 2: Plasma DNA Preprocessing

Once the parents are genotyped, it is possible to analyze the cfDNA with respect to the parental variants. Different combinations of parental genotypes are used in multiple ways during this step. Positions in which both parents are homozygous, but for different alleles, are utilized for the fetal fraction and length distribution calculation. In positions where at least one parent is heterozygous, an algorithm is needed in order to predict the fetal genotype. This algorithm requires the fetal fraction, the fragment length distribution, and the fragment lengths and supported alleles of each covering the candidate variants. This information is extracted and stored during the plasma DNA preprocessing step.

3.3.4 Step 2: Part A—Calculating the Fetal Fraction and Length Distribution

In order to calculate the fetal fraction and the fragment length distribution, it is required to explore the cfDNA sample in positions where both parents are homozygous for different alleles. Such positions enable the extraction of fetal-derived reads. For example, if the mother is homozygous for the reference allele and the father is homozygous for the alternate allele, the fetus is a mandatory heterozygous. Therefore, if a cfDNA read in such position presents the alternate allele, it cannot be a maternal read, and can be tagged as a fetal-derived read. All the fetal-derived reads are saved to a database, along with their fragment lengths. The other reads represent the allele that is shared by the fetus and the mother; these reads are also saved to a database, to enable the fetal fraction

calculation. The count of fetal reads represents only one fetal allele per position; to calculate the fetal fraction, the count of fetal reads is multiplied by two, and divided by total number of reads, both shared and fetal.

To understand which allele is supported by each cfDNA read that covers the relevant positions, it is required that the reads will be carefully aligned using methods as local reassembly or realignment around indels. However, as discussed in Subheading 3.3.1, such methods can be computationally intensive and, for this reason amongst others, modern variant callers work in a manner that does not require such processing. The read-level information can be extracted from the variant caller, but this is challenging, since by default, variant callers return only variant-level information. For example, in a given position, the user would be able to see what the genotype is, its probability, and even some statistics about the variant calling process in that position, such as the alternate allele read counts, the read depth, etc. However, it would be impossible to tell which read exactly supported each allele.

One way to achieve read-level information from the variant caller, is by analyzing its debugging output, which contains more information about all its calculations. To do so, the variant caller is first applied over the cfDNA sample in debugging mode. The variant caller also receives the parental genotypes as input. When the algorithm runs, read-level information is printed per analyzed position, and streamed to a small program that analyzes it. This program saves the reads at any position of interest, along with information about the genomic coordinate, the allele they support, whether they are fetal or shared, and their fragment length.

The fetal and shared reads that were collected for the fetal fraction calculation are also used for calculating the fragment length distributions. In paired-end sequencing, each DNA fragment is sequenced from both ends, returning a pair of forward and reverse strand reads. This enables measuring the length of the original fragment. This information is calculated during the mapping process and is saved in the resulting BAM file, where it is termed the template length. Fetal and maternal fragments are known to differ in their length distributions, and fetal fragments are generally shorter. In other words, the fetal fraction should be higher for short fragments, and lower for long fragments. For example, if the fetal fraction is 10%, the fetal fraction for fragment the length of 140 bp would be 15%, and in 166 bp long fragments it would be 5%. It is possible to utilize these differences for a more accurate noninvasive fetal genotyping. To do so, the fetal fraction is stratified per fragment length. To have enough fragments per length, a larger bin size can be chosen. A table of fetal fraction values per fragment length is saved and used later during the fetal variant calling, as explained in Subheading 3.3.6.

In summary, a variant caller is run in debugging mode, and the read-level output is streamed to a separate database. This information is then used for calculating the total fetal fraction and a fetal fraction value per fragment length.

3.3.5 Step 2: Part B—Identifying Potential Mutation Loci in the cfDNA

While running the variant caller and analyzing its read-level output, it is also important to save read-level information about positions in which variants are found. As mentioned, the variant caller looks for fetal variants only in positions where there are variants in the parental DNA. For each relevant position, the information about each read is saved to a database. It is important to note that de novo mutations are not explored in such analysis, since in such cases the parental genotype will not present a variant. It may be possible to perform the fetal variant calling already at this stage, instead of saving read-level information to a database. However, the noninvasive fetal variant calling requires two passes over the genome, as the first one is used for calculating the fetal fraction and length distribution. Theoretically, when cfDNA will be regularly extracted, and processing and sequencing will be performed many times, technical variability will become negligible. It will then be possible to use multiple methods to calculate the fetal fraction, and the first run over the cfDNA sample could be skipped.

Aside from the per variant read-level information that is saved in the database, a VCF file containing information about the fetal variant candidates is saved as well. The genotypes that appear in this file are meaningless, as they are based on an algorithm that ignores the cfDNA settings and assumes that the input sample represents one source of DNA. Other information in this file is still useful for the final step of the analysis, and, for this reason, it is kept.

3.3.6 Step 3: Bayesian Algorithm for Variant Calling

The previous steps were performed in order to gather information that is required for the noninvasive fetal genotyping: the fetal fraction, the length distribution, parental genotypes, and read-level information in candidate positions, that is, the supported alleles and fragment lengths. Once all this information is collected, it is possible to genotype the fetus using a variant caller. Regular variant callers, however, are not suited for this task, since the cfDNA has unique characteristics: (1) it is a mixture of two sources that share half of their genome; (2) the source of interest, that is, the fetal DNA, is found in a small amount compared with the maternal DNA, which acts as background noise; and (3) the fetal and maternal fragment length distributions differ. Therefore, a fetal genotyping algorithm is used, which can genotype the fetus using a cfDNA sample.

The noninvasive fetal genotyping algorithm or cfDNA-based fetal variant caller, termed *Hoobari*, is based on a Bayesian algorithm. In this algorithm, the prior probability is multiplied by the

likelihood for each possible fetal genotype and to calculate the posterior probability, each such product is divided by the sum of all the products. The genotype with the highest posterior probability is called as the fetal genotype. The algorithm is further explained below, and different considerations are discussed.

The first calculation in the algorithm is that of the prior probability, that is, what is the probability of each possible fetal genotype prior to the assessment of the cfDNA. In standard variant callers, the prior probability for finding a variant in a position is set 0.001 in all positions, a value which corresponds to the number of variants normally found in a human genome. In most variant callers it is possible to use other prior probabilities, for instance, by using population-based allele frequencies; however, this is not a common practice. In *Hoobari*, the prior probabilities are based on the parental genotypes. As mentioned, most variants are biallelic, meaning that there are two possible alleles in the population, while other alleles are much less abundant. Therefore, in most positions, the parents will not present different alternate alleles. This simplifies the problem, since the fetal genotype in a given position can be one of three options: homozygous for the reference allele, heterozygous, or homozygous for the alternate allele. The probability of each genotype is calculated using simple Mendelian inheritance laws. For example, if both parents are heterozygous, then the fetus has a probability of 0.25 to be homozygous, and 0.5 to be heterozygous. Multiallelic positions are currently not dealt with by *Hoobari*; adding further support is a simple algorithmic adjustment but requires adequate implementation (*see* **Note 4**).

Aside from the prior probabilities, the likelihood of each possible fetal genotype is calculated. The likelihood function is based on the cfDNA reads in every position, with every read serving as evidence that presents different degrees of support for each genotype. The likelihood function represents the chance of seeing a certain collection of evidence, if the fetus had a given genotype. In contrast with the prior probability, here the fetal genotype is not the question, but the assumption. Based on this assumption and given the maternal genotype at the position, the likelihood quantifies the chance for seeing a read with a certain allele, which arrives from a fragment with a certain length. The fragment length is used for choosing an adjusted fetal fraction, as explained above. This fetal fraction can be seen as the general probability that the read is fetal, regardless of the allele supported by it. The chance of the read being maternal is the complement of that probability. As an example, in a position where the mother is heterozygous, the likelihood is calculated for a certain read. The read supports the alternate allele, and the fetal fraction that corresponds to its fragment length is 17%. Assuming that the fetus is homozygous for the reference, the likelihood for such read would be 0.17×0, or simply 0, if it is a

fetal read, and, similarly, 0.83×0.5 if it is a maternal read. The overall likelihood of this read is the sum of these two calculations. In this example, if we assume that the fetal DNA is heterozygous, it becomes more likely to see a read that supports the alternate allele, and indeed, the sum becomes higher. The likelihood is calculated for each possible fetal genotype, for each read that covers the position. Eventually, the likelihood of a given fetal genotype in a certain position is the product of the likelihoods that were calculated for this genotype in each read.

In the last stage, the posterior probabilities are calculated. Up to this point, the probability for a certain genotype is the product of the prior probability and the likelihood, that is, the joint probability. To calculate the posterior probability of a certain genotype, its joint probability is divided by the sum of joint probabilities of all the possible genotypes. In effect, this calculation normalizes the probabilities to values between 0 and 1, and the sum of all the genotype probabilities per variant is 1. The posterior probabilities are not necessary for determining the genotype, as it can already be determined using the largest joint probability. The posterior probabilities are mainly calculated for reasons of easier reading of the results.

A few remarks about the algorithm should be taken into consideration. First, the calculation is a based on a naïve-Bayesian algorithm, meaning that the reads covering each position are considered as independent and as such, a simplification of the true relationship between reads. Second, to calculate the fetal genotype in a given position, a deep coverage is required, as explained above. As a result, a large number of reads is used in the calculation of the likelihood. It is important to note that while the prior probabilities are important and improve the accuracy, overall, it is the likelihood which is the core of the algorithm.

Running the noninvasive fetal variant calling algorithm of *Hoobari* is straightforward. The software tool requires the VCF files of the parents and the cfDNA. The location of the databases mentioned in previous steps are automatically found by *Hoobari* but can be manually supplied. This results in a VCF file with the predicted genotypes of the fetus.

In research setting, the next step would be to compare these results to the true fetal genotypes from the CVS or amniocentesis, as part of an accuracy assessment. It is also important to understand the effect of different features of each variant on the sensitivity, specificity and accuracy of the results. For example, it is possible to explore that relationship between the depth and the accuracy, and see which depth enables sufficient accuracy. Not only can features of the cfDNA affect results, but so too parental features. Just like in regular variant calling, the fetal genotyping process results in a VCF file containing all the predicted variants. From this point, the results can be conveniently processed using existing tools for VCF

manipulation. Many of these tools, however, iterate over all the variants and print the results to a new file. They would better suit a serial pipeline but not a test environment that requires many trials and analyses of the results. Moreover, merging the true fetal, cfDNA and parental VCF files to one file while maintaining a file with a valid format can become an issue. A preferred method to work with all these sources of information is to create a database of variants and query it each time.

3.3.7 Machine Learning-Based Variant Recalibration

Variant calling usually results in thousands and even millions of variants. A common practice is to continue filtering these variants based on different information related to each variant. For example, the variant list can be filtered by the depth, such that positions where the depth of coverage is too low are excluded. Other information can also be used, such as the allelic ratio, that is, the ratio between the counts of each allele in the position, the genotype probability, and the distance from the second-best genotype probability in the position. Traditionally, filters were serially applied, using trial and error to learn the optimal order and threshold values. However, when a lot of information is used for filtering, dependence between different filters cannot be ignored. This is especially important in noninvasive fetal variant calling, in which there is more information per variant, such as the parental information. For example, the allelic ratio of a heterozygous call in the parents can be affected by technical factors; these factors can similarly affect the cfDNA and cause errors in the fetal genotyping algorithm, which heavily relies on the allelic ratio.

Machine learning is a common method for modeling the interdependent relationship between filters. Generally, the idea behind this method is to look at previous analyses, along with their gold standard confirmation results, and learn which combination of values lead the algorithm to be more accurate. Instead of separate filters, the information of each variant is used as a set of different features of that variant. Numerous machine learning algorithms can find the relationship between the set of values in all the features to the gold standard result. The algorithm that is applied here is called random forest; this algorithm creates multiple decision trees and averages them. After the random forest model is trained, it can be applied in future analyses. As more analyses will be performed and verified using invasive techniques, data will accumulate, the model could be trained using diverse data and its generalizability will improve.

3.3.8 Machine Learning Model Training

As explained above, creating a database of variants is a convenient practice when analyzing the accuracy of a noninvasive fetal genotyping experiment. Such database is also useful for training a machine learning model. Training can be performed with different

packages available for many programming languages. Variants from different family analyses should be merged to one database which serves as the training set. The training set should be further split to training and validation sets. The validation set is used for testing of the trained model and adjustments of different parameters and methods, if needed. After this process is over, and the results over the training set are satisfactory, the model should be tested on a test set, that is, a family that is not a part of the training set.

When training, it is possible to stratify the training set to create models with different purposes. For example, a model for variants in which both parents are heterozygous might be more accurate if trained only on such variants, but a model for indels might gain accuracy by training over SNPs. Another part of training which requires attention is feature selection. Each variant has many features related to it, yet some might have a larger effect over the results than others. Features that have a small effect on the outcome should not be used, mainly because they lead to large models that are harder to train and apply (*see* **Note 5**).

4 Notes

Since fetal-derived DNA in maternal plasma was first discovered, it has become an important noninvasive genetic tool in prenatal diagnostics. Other developments that occurred since then, such as the sequencing of the human genome and the introduction of NGS, have also contributed to the rapid growth of NIPD. Current technology is adequate for using cfDNA-based test as screening for certain chromosomal disorders and for detection of point mutations in the level of up to several genes. However, noninvasive WGS of the fetus still requires a very deep coverage. As sequencing technologies continue to improve and their cost continues to drop, this problem might be resolved. Regardless, the algorithms that are used to analyze the sequenced reads can still be explored and improved. Here we outlined the most recent method for noninvasive fetal WGS, which deploys a dedicated algorithm, software and pipeline for noninvasive fetal genotyping. The genotyping method can be used for smaller regions of the genome as well, and the same ideas can assist in the analysis of related fields, such as circulating tumor DNA analysis. To help future bioinformaticians that are interested in this field, several recurrent themes are emphasized throughout this chapter, as follows:

1. Many parts of the analysis described here can still be further optimized. For instance, a comparison between different fetal fraction calculation methods, different mapping and realignment methods, and of course, various options in the technical steps prior to the bioinformatical analysis.

2. Working with such deep sequencing files create computational challenges. For example, a BAM file of the plasma DNA, sequenced to a coverage of 300× using WGS, can range in the size from 0.5–1 Terabytes. To work with such large data, it is important to use streams and parallelization. Most of the manipulations described here can be streamed from one tool to another, and most tools can be parallelized. Many external sequencing service providers also suggest an automated workflow for several computational processing steps, which is already optimized to run quickly on their infrastructure, which is usually strong; it is possible to use this option as well.
3. Working with existing tools for generic parts of an analysis is common practice in bioinformatics. If a certain software tool or package is already widely used and there is much experience with it, there is no reason not to use it inside a larger pipeline, even if it is not exactly used as intended. Pipelines should be modular, such that specific steps can be replaced by ever improving algorithms and external tools.
4. The solutions brought here exclude several cases, such as mutations in sex chromosomes, multiallelic variants, and others. It is recommended to first explore the most general and common problem; in this case it is biallelic variants in autosomes. Only then the model can be generalized to support each and every scenario.
5. Many models and methods are available for machine learning. Training set and feature selection, as well as adjustment of different model parameters, are part of the art of creating a good machine learning model. These techniques are discussed in greater detail in many articles, guides, courses, and books. We recommend users further explore these areas, as we could not review them to a full extent here.

References

1. Akolekar R, Beta J, Picciarelli G et al (2015) Procedure-related risk of miscarriage following amniocentesis and chorionic villus sampling: a systematic review and meta-analysis. Ultrasound Obstet Gynecol 45:16–26. https://doi.org/10.1002/uog.14636
2. Tabor A, Alfirevic Z (2010) Update on procedure-related risks for prenatal diagnosis techniques. FDT 27:1–7. https://doi.org/10.1159/000271995
3. Zelig CM, Knutzen DM, Ennen CS et al (2016) Chorionic villus sampling, early amniocentesis, and termination of pregnancy without diagnostic testing: comparison of fetal risk following positive non-invasive prenatal testing. J Obstet Gynaecol Can 38:441–445.e2. https://doi.org/10.1016/j.jogc.2016.03.006
4. Lo YM, Corbetta N, Chamberlain PF et al (1997) Presence of fetal DNA in maternal plasma and serum. Lancet 350:485–487. https://doi.org/10.1016/S0140-6736(97)02174-0
5. Chiu RWK, Chan KCA, Gao Y et al (2008) Noninvasive prenatal diagnosis of fetal chromosomal aneuploidy by massively parallel genomic sequencing of DNA in maternal plasma. Proc Natl Acad Sci U S A 105:20458–20463. https://doi.org/10.1073/pnas.0810641105

6. Lo YMD, Lun FMF, Chan KCA et al (2007) Digital PCR for the molecular detection of fetal chromosomal aneuploidy. Proc Natl Acad Sci U S A 104:13116–13121. https://doi.org/10.1073/pnas.0705765104
7. Papageorgiou EA, Patsalis PC (2012) Non-invasive prenatal diagnosis of aneuploidies: new technologies and clinical applications. Genome Med 4:46. https://doi.org/10.1186/gm345
8. Fan HC, Blumenfeld YJ, Chitkara U et al (2008) Noninvasive diagnosis of fetal aneuploidy by shotgun sequencing DNA from maternal blood. Proc Natl Acad Sci U S A 105:16266–16271. https://doi.org/10.1073/pnas.0808319105
9. van der Meij KRM, Sistermans EA, Macville MVE et al (2019) TRIDENT-2: national implementation of genome-wide non-invasive prenatal testing as a first-tier screening test in the Netherlands. Am J Hum Genet 105:1091–1101. https://doi.org/10.1016/j.ajhg.2019.10.005
10. Chu T, Yeniterzi S, Rajkovic A et al (2014) High resolution non-invasive detection of a fetal microdeletion using the GCREM algorithm. Prenat Diagn 34:469–477. https://doi.org/10.1002/pd.4331
11. Jensen TJ, Dzakula Z, Deciu C et al (2012) Detection of microdeletion 22q11.2 in a fetus by next-generation sequencing of maternal plasma. Clin Chem 58:1148–1151. https://doi.org/10.1373/clinchem.2011.180794
12. Peters D, Chu T, Yatsenko SA et al (2011) Noninvasive prenatal diagnosis of a fetal microdeletion syndrome. N Engl J Med 365:1847–1848. https://doi.org/10.1056/NEJMc1106975
13. Neofytou MC, Tsangaras K, Kypri E et al (2017) Targeted capture enrichment assay for non-invasive prenatal testing of large and small size sub-chromosomal deletions and duplications. PLoS One 12:e0171319. https://doi.org/10.1371/journal.pone.0171319
14. Perlado S, Bustamante-Aragonés A, Donas M et al (2016) Fetal genotyping in maternal blood by digital PCR: towards NIPD of monogenic disorders independently of parental origin. PLoS One 11:e0153258. https://doi.org/10.1371/journal.pone.0153258
15. Lam K-WG, Jiang P, Liao GJW et al (2012) Noninvasive prenatal diagnosis of monogenic diseases by targeted massively parallel sequencing of maternal plasma: application to β-thalassemia. Clin Chem 58:1467–1475. https://doi.org/10.1373/clinchem.2012.189589
16. Jenkins LA, Deans ZC, Lewis C, Allen S (2017) Delivering an accredited non-invasive prenatal diagnosis service for monogenic disorders, and recommendations for best practice. Prenat Diagn. https://doi.org/10.1002/pd.5197
17. Hayward J, Chitty LS (2018) Beyond screening for chromosomal abnormalities: advances in non-invasive diagnosis of single gene disorders and fetal exome sequencing. Semin Fetal Neonatal Med 23(2):94–101. https://doi.org/10.1016/j.siny.2017.12.002
18. Drury S, Williams H, Trump N et al (2015) Exome sequencing for prenatal diagnosis of fetuses with sonographic abnormalities. Prenat Diagn 35:1010–1017. https://doi.org/10.1002/pd.4675
19. Best S, Wou K, Vora N et al (2018) Promises, pitfalls and practicalities of prenatal whole exome sequencing. Prenat Diagn 38:10–19. https://doi.org/10.1002/pd.5102
20. Stark Z, Tan TY, Chong B et al (2016) A prospective evaluation of whole-exome sequencing as a first-tier molecular test in infants with suspected monogenic disorders. Genet Med 18:1090. https://doi.org/10.1038/gim.2016.1
21. Mackie FL, Carss KJ, Hillman SC et al (2014) Exome sequencing in fetuses with structural malformations. J Clin Med 3:747–762. https://doi.org/10.3390/jcm3030747
22. Lo YMD, Chan KCA, Sun H et al (2010) Maternal plasma DNA sequencing reveals the genome-wide genetic and mutational profile of the fetus. Sci Transl Med 2:61ra91. https://doi.org/10.1126/scitranslmed.3001720
23. Fan HC, Gu W, Wang J et al (2012) Non-invasive prenatal measurement of the fetal genome. Nature 487:320–324. https://doi.org/10.1038/nature11251
24. Kitzman JO, Snyder MW, Ventura M et al (2012) Noninvasive whole-genome sequencing of a human fetus. Sci Transl Med 4:137ra76. https://doi.org/10.1126/scitranslmed.3004323
25. Snyder MW, Adey A, Kitzman JO, Shendure J (2015) Haplotype-resolved genome sequencing: experimental methods and applications. Nat Rev Genet 16:344–358. https://doi.org/10.1038/nrg3903
26. Chan KCA, Jiang P, Sun K et al (2016) Second generation noninvasive fetal genome analysis reveals de novo mutations, single-base parental inheritance, and preferred DNA ends. Proc Natl Acad Sci U S A 113(50):E8159–E8168. https://doi.org/10.1073/pnas.1615800113

27. Rabinowitz T, Polsky A, Golan D et al (2019) Bayesian-based noninvasive prenatal diagnosis of single-gene disorders. Genome Res https://doi.org/10.1101/gr.235796.118
28. Sillence K (2016) Cell-free fetal DNA (cffDNA) enrichment for non-invasive prenatal testing (NIPT): a comparison of molecular techniques
29. Bianchi DW, Chiu RWK (2018) Sequencing of circulating cell-free DNA during pregnancy. N Engl J Med 379:464–473. https://doi.org/10.1056/NEJMra1705345
30. Li H, Durbin R (2009) Fast and accurate short read alignment with Burrows–Wheeler transform. Bioinformatics 25:1754–1760. https://doi.org/10.1093/bioinformatics/btp324
31. Faust GG, Hall IM (2014) SAMBLASTER: fast duplicate marking and structural variant read extraction. Bioinformatics 30:2503–2505. https://doi.org/10.1093/bioinformatics/btu314
32. Li H, Handsaker B, Wysoker A et al (2009) The Sequence Alignment/Map format and SAMtools. Bioinformatics 25:2078–2079. https://doi.org/10.1093/bioinformatics/btp352
33. Tarasov A, Vilella AJ, Cuppen E et al (2015) Sambamba: fast processing of NGS alignment formats. Bioinformatics 31:2032–2034. https://doi.org/10.1093/bioinformatics/btv098
34. Garrison E (2012) freebayes: Bayesian haplotype-based genetic polymorphism discovery and genotyping
35. Mardy A, Wapner RJ (2016) Confined placental mosaicism and its impact on confirmation of NIPT results. Am J Med Genet C Semin Med Genet 172:118–122. https://doi.org/10.1002/ajmg.c.31505
36. Ashoor G, Syngelaki A, Poon LCY et al (2013) Fetal fraction in maternal plasma cell-free DNA at 11–13 weeks' gestation: relation to maternal and fetal characteristics. Ultrasound Obstet Gynecol 41:26–32. https://doi.org/10.1002/uog.12331
37. Hu P, Liang D, Chen Y et al (2019) An enrichment method to increase cell-free fetal DNA fraction and significantly reduce false negatives and test failures for non-invasive prenatal screening: a feasibility study. J Transl Med 17:124. https://doi.org/10.1186/s12967-019-1871-x
38. Jorgez CJ, Bischoff FZ (2009) Improving enrichment of circulating fetal DNA for genetic testing: size fractionation followed by whole gene amplification. FDT 25:314–319. https://doi.org/10.1159/000235877
39. Webb A, Madgett T, Miran T et al (2012) Non invasive prenatal diagnosis of aneuploidy: next generation sequencing or fetal DNA enrichment? Balkan J Med Genet 15:17–26. https://doi.org/10.2478/v10034-012-0013-z
40. Peng XL, Jiang P (2017) Bioinformatics approaches for fetal DNA fraction estimation in noninvasive prenatal testing. Int J Mol Sci 18:453. https://doi.org/10.3390/ijms18020453
41. Shang J, Zhu F, Vongsangnak W et al (2014) Evaluation and comparison of multiple aligners for next-generation sequencing data analysis. Biomed Res Int 2014(11, supplement):309650. https://www.hindawi.com/journals/bmri/2014/309650/abs/. Accessed 8 Jan 2020
42. Campbell IM, Gambin T, Jhangiani SN et al (2016) Multiallelic positions in the Human Genome: challenges for genetic analyses. Hum Mutat 37:231–234. https://doi.org/10.1002/humu.22944

Chapter 12

Genome-Wide Noninvasive Prenatal Diagnosis of De Novo Mutations

Ravit Peretz-Machluf, Tom Rabinowitz, and Noam Shomron

Abstract

Noninvasive prenatal diagnosis (NIPD) has become a common, safe, and effective procedure for detection of inherited diseases early in pregnancy. It is based on the analysis of fetal cell-free DNA (cffDNA) derived from the placenta, circulating in the maternal plasma. De novo mutations, although rare, cause a considerable number of dominant genetic disorders. Due to the sparse representation of fetal-derived sequences in the blood, the challenge of detecting low frequency fetal de novo mutations becomes preponderant. Hence, this detection type requires deep genome-wide sequencing of cffDNA from maternal plasma and a unique analysis approach. Here we suggest and discuss a method for identifying de novo mutations based on whole genome sequencing (WGS) of cell-free DNA (cfDNA) from maternal plasma samples. Our method consists of an augmented pipeline for analysis of de novo mutation candidates. It begins with an enhanced noninvasive fetal variant calling step, followed by a candidate de novo mutation filtration, and then finally, a supervised machine learning approach is utilized for reduction of false positive rates. Overall, this study provides a basis for genome-wide de novo mutation analysis in NIPD procedures, which could be used in any procedure where rare de novo mutations should be carefully picked out of a sea of data.

Key words De novo mutations, Machine learning, Noninvasive prenatal diagnosis, NIPD, Fetal, Cell-free DNA, cfDNA, Cell-free fetal DNA, cffDNA, Hoobari

1 Introduction

Germline de novo mutations are extremely rare: their rate estimate in the human genome is 1.18×10^{-8} [1]. Nevertheless, they play a substantial role in common human diseases [2] and therefore become crucial for a comprehensive and thorough prenatal genetic diagnosis. The low rate of de novo mutations causes the specificity in the identification process to be low, resulting in a high rate of false positives. The main challenge is to identify the actual de novo mutations out of many candidates found, by filtering out false

Ravit Peretz Machluf and Tom Rabinowitz contributed equally to this work.

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_12,

positive results without omitting any deleterious mutations, thus maintaining both high sensitivity and high specificity.

Prenatal whole exome sequencing (WES) or whole genome sequencing (WGS) gradually becomes possible in noninvasive techniques using maternal plasma DNA; however, these sequencing methods are still considered common in invasive tests, such as amniocentesis or chorionic villus sampling (CVS). Pure fetal DNA sequencing using WES/WGS ensures high enough resolution, at the level of a single nucleotide [3]. Unfortunately, these invasive tests impose risk of pregnancy loss [4, 5] and delay the diagnosis conclusions to a far later phase in the pregnancy, since they are not available in its early stages.

1.1 Noninvasive Prenatal Diagnosis

The rapid development and popularity of noninvasive prenatal diagnosis (NIPD) in the last decade have paved the way for early detection of fetal variants. Typically, it includes analysis of cell-free DNA (cfDNA) and cell-free fetal DNA (cffDNA) in maternal plasma, hence exposing the mother and fetus to no risk in comparison to invasive techniques [4, 5]. At first, NIPD was used to identify chromosomal abnormalities [6, 7] and sex determination [8, 9]. Later on it has become feasible to identify specific monogenic disorders via NIPD such as β-thalassemia and achondroplasia [10, 11]. More recently, new progress has been made, enabling genome-wide detection of monogenic disorders of both paternal and maternal origin through noninvasive fetal WGS [12].

1.2 NIPD of De Novo Mutations

NGS-based de novo mutation identification poses a great challenge. Their low rate makes it difficult to differentiate a true de novo mutation from a false one, due to sequence or alignment error. This challenge intensifies when the testing is done noninvasively, as de novo point mutations require deep coverage in each genomic locus. Few methods to detect fetal de novo mutations noninvasively are available, but they are aimed at specific loci, genes or small gene panels. One attempt to identify de novo mutations in a genome-wide sequencing of maternal plasma DNA analysis is made by filtering potential de novo mutations. The process includes increasingly stringent filters based on predefined criteria such as the depth of coverage and variant calling quality score [13, 14]. This method, however, has limitations; mainly the fact that the filters were applied independently, without utilizing the co-dependency between them. Consequently, as the method's specificity increases, true de novo mutations are filtered out resulting in lower sensitivity. More recently, a new noninvasive prenatal screening panel was introduced; this panel was designed for the detection of de novo or paternally inherited disease-causing variants in 30 genes associated with frequent human dominant monogenic disorders. A unique molecular indexing (UMI) based method was used along with a deduplication algorithm to reduce both

sequencing and PCR errors [15]. This approach is aimed at the identification of frequent, pre-defined fetal variants using prior knowledge on potentially affected genes or diseases. In this method, the identification of de novo mutations is limited to pre-defined, handcrafted regions in the genome and by design may miss out on pathogenic variants without prior dispositions in the process.

1.3 NIPD of De Novo Mutations Using Machine Learning

Former attempts to identify de novo mutations used consecutive filtering criteria and were found not effective enough in keeping high specificity along with high sensitivity. These techniques resulted in either missing true de novo mutations, that is, lower sensitivity, or having a large number of false positives, that is, low specificity. As de novo mutations are highly associated with human diseases, maximized sensitivity is extremely important; lowering the specificity, however, introduces a complex classification problem: how to differentiate many false variants from true de novo mutations, and how to properly classify de novo mutations in the context of noninvasive prenatal diagnosis. In the past few years, machine learning has become a powerful platform in genomics. It enables modeling of vast amount and complex dataset, including data from WGS, for the purpose of identifying genetic variants [16]. Supervised machine learning algorithms are becoming more accessible. These algorithms, given a set of known positive variants in advance, are used to train a model that is able to classify variants as real versus artifact [17]. De novo mutation categorization can benefit greatly from a machine learning classification model as it can utilize many of the attributes of these variants, while considering their complex relationship with one another. This method can improve the accuracy of de novo mutation identification, eliminating false, irrelevant, non de novo variants. However, to harness such powerful platform in the context of noninvasive de novo mutation detection, three main obstacles should first be addressed: (1) the extremely imbalanced ratio of positives to negatives, that is, true de novo mutations to false positives (~74:100 K); (2) the limited data available of genotyped plasma-based trios, resulting in very few true positive mutations in total; and most importantly, (3) the unique nature of the fetal genotyping using cfDNA in maternal plasma, which compared to genotyping of invasively acquired fetal cells, results in lower predicting confidence.

Previous work had been performed with machine learning models to identify de novo mutations, yet not in the clinical context of prenatal diagnosis. For instance, one approach was based on a gradient-boosting algorithm for filtering de novo mutations in standard family trio WGS [18]. Due to easy accessibility to vast data of genotyped family trios, this work was able to overcome the first two obstacles mentioned above, yet it did not aim at a noninvasive prenatal solution. Another utilization of machine learning

models in the context of cfDNA have been performed in the field of circulating tumor DNA (ctDNA), specifically for the detection of early stage colorectal cancer [19]. Although ctDNA fragments hold similar characteristics to the fetal fragments analyzed in NIPD, this model cannot be implemented for cfDNA in a straightforward manner. Eventually, progress made within the field of liquid biopsies can potentially be relevant for NIPD and vice versa.

1.4 Summary

NIPD has come a long way in the past decade and so did de novo mutation identification. Yet effectively diagnosing de novo mutations using NIPD techniques is still considered a major challenge, especially when attempting to provide high sensitivity and specificity without limiting the analysis to a pre-defined panel of genes.

Here we describe a new pipeline for genome-wide prenatal de novo point mutation detection. In this pipeline (Fig. 1), the raw sequences are first aligned to the reference genome. Second, an adjusted version of Hoobari, a Bayesian-based tool for noninvasive fetal variant calling [12] is used in order to obtain a list of de novo mutation candidates. These candidates are fetal single nucleotide variant (SNV) positions that are suspected to be de novo mutations. Third, the list of de novo mutation candidates is filtered based on the corresponding parental genotypes, and only de novo candidates that are absent in the parents are kept. A supervised machine learning model is trained based on the filtered fetal de novo candidate list and the true de novo mutations. The true de novo mutations are based on true fetal genotypes, which are achieved through an invasive test followed by a trio analysis. Finally, on a different family, the trained model is tested, that is, it is used for further reduction of false positive rates, aiming to address both specificity and sensitivity issues of previous methods. Even though the pipeline presented here is a prototype, limited due to data size and imbalance issues, we present the concept of machine learning-based prenatal de novo mutation identification, which is both noninvasive and genome-wide. This will enable building a generic and robust model given large scale data of maternal plasma-based trios for future use.

2 Materials

2.1 Pre-analysis Materials

The materials required for de novo mutation detection pipeline are parental samples, pure fetal sample and the maternal plasma. The pipeline presented here is composed of various steps, the last of them being a machine learning classifier model, distinguishing true fetal de novo mutations from false candidates. A machine learning approach was chosen for its ability to consider various interdependent features as opposed to a manual classification method. Yet, to be able to harness a machine learning classifier for this task, a

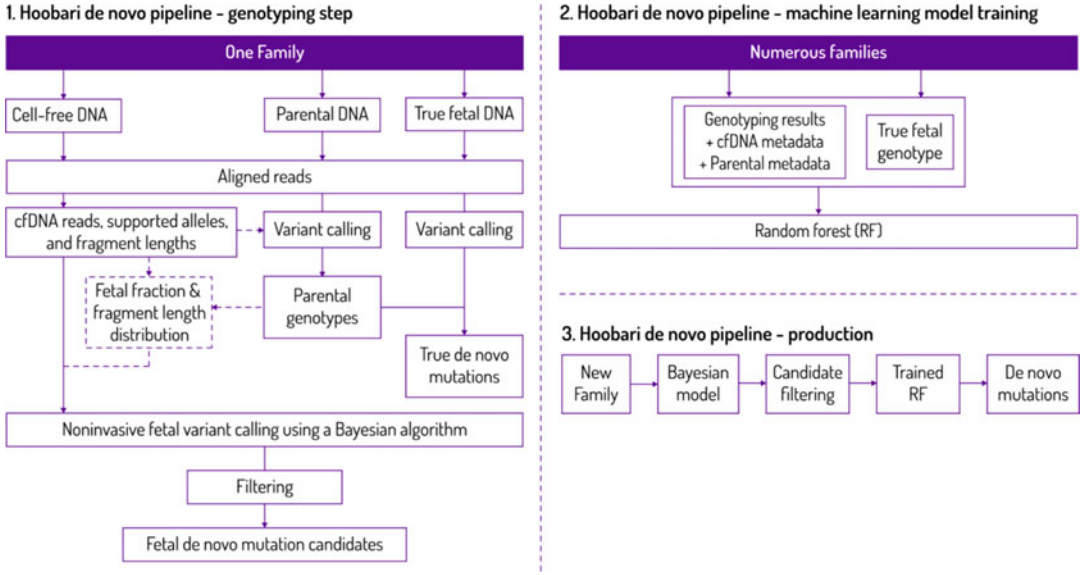


Fig. 1 Workflow for noninvasive fetal de novo mutation genotyping experiment. (1) Each family consists of four samples: two parental genomic DNA samples, maternal cfDNA and pure fetal DNA that is used as the gold standard. All samples are mapped, resulting in reads that are aligned to the human reference genome. First, the pure fetal sample is genotyped by a variant caller. The genotypes are then filtered to form the true de novo mutations list. Second, the cfDNA is genotyped using a regular variant caller. The parents are then genotyped in the same positions that presented a possible variant in the cfDNA. In addition, the parental genotypes are utilized for calculating the fetal fraction and the fragment length distribution, which, together with the cfDNA in each position of interest, are used for noninvasive genotyping of the fetus. Finally, fetal variants with parental homozygous genotype to the reference genome are filtered as de novo mutations candidates (2) Using families that have already undergone noninvasive fetal genotyping, a machine learning model is trained. The features used are metadata of the parental and cfDNA variants, that were not used by the Bayesian algorithm. The labels are taken from the true de novo mutation list previously created. (3) The trained machine learning model is eventually used as an integral part of genome-wide de novo mutations prediction, that accurately filters the called variants

unified, high quality sequencing of these samples is in order. In our research, the different samples were sequenced using the same sequencing technique, Illumina based NGS for whole genome (WGS). Sequencing bias and errors are highly affected by biological and physical aspects of the DNA, like fragments GC content, sequenced reads length and others. These sequence parameters are later processed by variant calls and translated to key machine learning classification features. In order for them to be used effectively by the model, these features should be similar in structure across different samples and families. Nonetheless, the work done here can be generalized to other sequencing techniques. This section will review pre-analysis steps generally, and elaborate on the specific considerations that should be made for de novo mutation detection.

Out of the samples processed, the maternal plasma is the one holding unique characteristics and therefore requires special treatment. The exceptional handling of plasma derived cfDNA, its collection, extraction, enrichment, library preparation and sequencing, has direct implications on the NIPD analysis. Yet it may be even more challenging for de novo mutation analysis. During this procedure, a blood sample is first collected from the pregnant mother using a number of Ethylene-diamine-tetra-acetic acid (EDTA) tubes. The required number of tubes collected might be even higher for de novo mutations analysis, such that a valid coverage of both mother and fetal DNA is achieved. Second, the sample should be centrifuged to separate white blood cells and plasma and in order to remove any residual cells. The first cycle aims to prevent blood cells from hemolysis that contaminates the plasma. Unless done carefully and as close as possible to the blood being drawn, the cfDNA characteristics may shift and compromise the accuracy of the analysis. Third, the cfDNA extraction from the plasma is done via dedicated purification kits. For de novo mutation analysis, extra purification and cleaning steps are required in order to minimize the chance for error. A possible additional step is the actual (physical) enrichment of fetal-derived DNA from the plasma. This step leads to loss of DNA material and hence to lower sensitivity, therefore, we have not used it in our approach. Fourth, prior to its sequencing, the purified cfDNA requires library preparation. In this context, the main concern would be modifying the natural fragmentation of the cfDNA in the process. PCR amplification cycles are required for very deep sequencing and small amounts of DNA, even with PCR-free WGS protocols. These cycles lead to errors, higher rates of duplicated reads and overall bias to certain reads. This is particularly challenging in the context of fetal de novo mutations in cfDNA, since their relatively low representation in the plasma exposes them to these kinds of errors (as explained in Subheading 3.2.4). Finally, the purified cfDNA sample is sequenced. It is important to choose a platform that can reach the very deep coverage required for de novo mutations analysis.

We find it appropriate to mention that despite the approach we propose here, the field of NIPD of monogenic diseases and specifically detection of de novo mutations is in its infancy. There are no widely accepted standards for maternal plasma analyses nor those that meet the quality restrictions. Such well needed guidelines can be achieved also through clinical and basic research in similar fields such as ctDNA in liquid biopsies for somatic de novo mutation detection.

2.2 Computational Tools

There are several tools available for the analysis of cfDNA. Some of them are general NGS analysis tools that have been utilized for this purpose (Table 1). Hoobari is in fact the only software tool specific for comprehensive noninvasively genotyping a fetus. Hoobari's

Table 1
Computational tools used during genome-wide NIPD of point de novo mutations

Tool/ resource	Function	References	Link
BWA-MEM	Read alignment	[30]	http://bio-bwa.sourceforge.net
Samblaster	Deduplication	[31]	https://github.com/GregoryFaust/samblaster
Samtools	BAM manipulation	[32]	http://www.htslib.org
Sambamba	BAM manipulation	[33]	https://lomereiter.github.io/sambamba
Freebayes	Variant calling	[34]	https://github.com/ekg/freebayes
Hoobari	Noninvasive fetal variant calling	[12]	https://github.com/nshomron/hoobari
Vcftools	VCF manipulation	[35]	https://vcftools.github.io/index.html
Scikit-learn	Python machine learning toolkit	[36]	https://scikit-learn.org/

modular platform was utilized to form a prototype pipeline, named Hoobari-denovo, for genome wide noninvasive de novo mutations detection. Currently, Hoobari's variant calling implementation is based on Freebayes as its variant caller. This can easily change to other variant callers as Hoobari's code is open sourced and available on GitHub.

In addition to Hoobari's standard steps, a classification machine learning model was used. There are many comprehensive freely available machine learning toolkits. Scikit-learn, which is used here, is a robust and very popular Python library for classification, regression, clustering and other machine learning problem types; yet it can be easily replaced by other tools.

3 Methods

3.1 Sample Selection

3.1.1 Family Trios

De novo mutations are basically genetic alterations resulting from a variant in a germ cell of one of the parents, or one that happens during early embryogenesis. Thus, as opposed to a basic NIPD analysis process, de novo mutation NIPD cannot rely solely on a blood sample from the pregnant mother. In addition to a sample from the mother, which contributes both cfDNA extracted from the plasma and maternal genomic DNA extracted from white blood cells, a paternal sample is crucial. In general, using the paternal DNA sample is beneficial to any fetal genotyping process. It improves the sensitivity to paternal derived mutations; provides key genomic loci where the parental genotypes are homozygous

for different alleles, which improves accuracy of fetal fraction calculation; and, given the maternal genotype, it enables prior probability calculation for the fetal genotype using Mendelian laws. Put simply, fetal noninvasive genotyping can be performed without a paternal sample, but it will be less accurate. The diagnosis of fetal de novo mutations, on the other hand, must include the paternal sample as mandatory input. Aside from the abovementioned straightforward advantages, the access to the paternal DNA sample enables us to differentiate between an inherited paternal mutation to a de novo one. In both scenarios, the representation of the alternate allele in the cfDNA is relatively small, as it does not exist in the maternal genome. In most cases, the fetal variant is a paternal inherited one; thus, the prior knowledge of the paternal genotype enables to filter many positions out. Predicting the fetal genotype in these positions is relatively straightforward. The remaining positions are those that are interesting in the de novo analysis context; such positions present a variant in the cfDNA, which is absent from both maternal and paternal genotypes. These fetal genetic alterations can be explained in one of two scenarios. They can either be (1) a potential de novo mutation or, (2) an error originating in one of the pre-analysis steps. To distinguish between these two, a more sophisticated approach is superimposed. This step consists of the machine learning algorithm described here. Hence, performing NIPD of de novo mutations requires using family trios that are composed of both maternal and paternal genomic DNA samples, and cfDNA extracted from the maternal plasma.

3.1.2 *True Fetal Sample*

The sequenced pure fetal DNA sample is used as the gold standard for the noninvasive fetal genotyping. This sample, taken invasively, undergo a standard family trio analysis along with the parental samples. Parent–offspring trio variant calling is a standard and well-studied analysis paradigm which introduced several available methods for trio genotyping. Yet detection of de novo mutations remains one of the main challenges in such analyses. Therefore, in order to retrieve true fetal de novo mutations from the pure fetal sample, an additional analysis is often required. Several methods addressing this challenge were suggested in the past few years [18, 20].

The Hoobari-denovo pipeline comprises a generic separated step for de novo mutation detection in standard trios. Here, the trio is composed of the pure fetal sample along with the parental DNA samples, and the process withholds several steps: (1) variant calling of both fetus and parents; (2) extracting positions that present a variant in the fetus but not in the parents; (3) applying a technical quality control step to eliminate noise of sequencing errors using variant quality features, such as the depth of coverage and the variant calling quality score. As neither ours or others' approaches [18, 20] have been fully standardized for de novo

mutation detection in parent–offspring trios, we recommend trying optimization procedures for true de novo mutation identification, perhaps even by combining several methods (*see* **Note 1**).

The pure fetal DNA sample is acquired invasively for experimental purposes only. It can be extracted during pregnancy via chorionic villus sampling (CVS) or amniocentesis, or during labor from the umbilical cord blood. It is important to note that each invasive technique introduces different risks and have an impact on the sample analysis accuracy. Comparing CVS to amniocentesis, CVS holds a higher risk for miscarriage than amniocentesis [4] and its analysis is more prone to placental mosaicism-related errors. Yet it is available in earlier stages of the pregnancy, while amniocentesis is performed at the beginning of the second trimester. Nonetheless, the differences between the tests are relatively minor, and either of them can be used as a gold standard. In most NIPD studies, women are recruited when they go through an invasive test, so researchers can add the type of the test to their inclusion or exclusion criteria. Generally speaking, researchers should consider that some of the errors that arise in an NIPD analysis can originate from the invasive test itself and not from the NIPD algorithm.

3.2 Biological and Technical Considerations

3.2.1 Fetal Fraction

During pregnancy, the maternal plasma cfDNA contains fragments of both maternal and fetal origin. The fetal fraction is the estimated rate of fetal-derived fragments in the cfDNA. This fraction increases as the pregnancy progresses and the placenta grows. Upon 10–12 weeks of gestation the fetal fraction is usually around 10% [21]. The fetal fraction can be manually enriched through various laboratory enrichment methods prior to sequencing, utilizing the difference in fragment lengths between the fetus and mother [22–24]. These methods may lower the sensitivity of the analysis [25] and therefore, are not included in Hoobari-denovo pipeline. Instead, the fragment length distribution is calculated and used by Hoobari's Bayesian algorithm (as described below). The suggested approach in this guide for NIPD of de novo mutations is based on Hoobari, a noninvasive variant calling for monogenic disorders [12], which in turn depends on parameters such as the fetal fraction; therefore, accurately calculating the fetal fraction leads to improved accuracy of the analysis. Several methods are available for fetal fraction calculation. These methods apply biological filtering and statistical methods to distinguish fetal fragments from maternal ones. Most methods use either paternal-specific alleles that can be found in the maternal plasma, or fragments arriving from the Y chromosome [26]. Hoobari's calculation is based on paternal-specific alleles but does not include corrections based on biological characteristics nor statistical ones. This approach can be further optimized to improve the analysis accuracy (*see* **Note 2**).

3.2.2 Fragment Length Distribution

DNA fragments circulating in the maternal plasma can be of maternal or fetal origin. Fetal-derived fragments hold a different length distribution than maternal-derived ones, and they are generally shorter. This phenomenon is utilized in fetal genotyping as it assists classifying the origin of cfDNA fragments. The length distribution also depends on technical factors such as the handling of the maternal blood sample. Uncareful handling can change the natural distribution of the fragments due to genomic data release from maternal blood cells. However, as the length differences remain unchanged during pregnancy, as well as among women, it is a vital tool in NIPD analysis. Thus, meeting the optimized guidelines for maternal blood sample handling is highly important.

3.2.3 De Novo Mutations Rate

De novo mutations are extremely rare; their rate estimate in the human genome is 1.18×10^{-8}, which corresponds to ~74 novel SNVs per genome per generation [1]. The overall rate of de novo germline mutations may be higher in individuals with genetic disease than in those without [2]. The rate is higher in males compared to females, and correlates with paternal age [27]. This can be explained by the greater number of germline cell divisions in men compared with women, such that more replication errors occur [28]. Although rare, de novo mutations are responsible for many human genetic diseases. Therefore, any comprehensive DNA analysis should include de novo mutations diagnosis, especially a prenatal one.

Detecting de novo mutations in the genome in an unbiased manner, that is, without prior information about the presence of such alterations, poses a great challenge also in a standard parent–offspring trio analysis. Sequencing artifacts, although compromising any DNA analysis, dramatically affect the ability to detect de novo mutations. Sequencing artifacts in the offspring are essentially false variants that do not appear in the parental genome, hence mistakenly detected as de novo mutations. Similarly, these artifacts can lead to the opposite error: a false negative variant call in one of the parents, i.e. missed parental variant position by the variant caller, cause offspring inherited heterozygous position to be mistakenly detected as a de novo mutation. The first error is much more prone in NIPD de novo mutation detection as a result of low fetal fraction in the maternal blood. Insertion-Deletions (Indels) and copy number variants (CNV) de novo mutations are even harder to detect than SNVs using WGS [2] and will not be discussed in this chapter (*see* **Note 3**).

3.2.4 Depth of Coverage

The depth of coverage affects the quality of sequencing; the deeper the coverage, the higher the confidence in the content of the genomic locus, and the analysis becomes more accurate. The depth of coverage is important in any NGS-based analysis, but it becomes crucial when handling de novo mutations as their

biological nature makes them similar to sequencing and alignment errors.

Analyzing fetal WGS for de novo mutations noninvasively requires an even deeper coverage. The fetal DNA representation is much lower and depends on the fetal fraction in the cfDNA. For instance, a 10% fetal fraction will require a 300× depth of coverage of the cfDNA to yield 30 fragments covering a fetal position. This deep coverage is essential for two main reasons: (1) to minimize the effect of the abovementioned errors and lead to less candidate positions, which lowers the false positive rate. In addition, as opposed to inherited variants, prior knowledge regarding the location of de novo mutations is not contributed by the parental genotype. Thus, it is much harder to distinguish fetal true de novo mutations from non-variant positions, which are the majority of positions in the genome. (2) A deep coverage provides more evidence on the genomic loci under investigation and hence, it can also assist in reducing the overlooked de novo mutations, lowering the false negative rate. However, reaching such deep coverage with WGS is still costly, and it demands extensive computational resources for downstream analysis. Working around these constraints can be carried out by limiting the analysis to specific regions of the genome. This approach is problematic in the context of genome-wide de novo mutation detection (as described in Subheading 3.2.5).

3.2.5 Genome Wide Sequencing

Genome wide NIPD for de novo mutation detection can be achieved by either WGS or dedicated NGS panels covering genes of clinical relevance. WGS provides comprehensive information for the entire genome; thus, it is less prone to missing new mutations. One of the main advantages of WGS is that it enables a relatively low depth of coverage, while maintaining high accuracy results. This feature enables using WGS in PCR-free protocols which in turn eliminate PCR amplification inherent errors. These capabilities, however, are limited; the genome-wide de novo mutations detection requires an extra deep coverage (as mentioned above). In some cases, reaching such coverage still requires several PCR amplification cycles prior to sequencing, after library preparation. And yet this is far less error prone than the alternative of using WES or NGS based panels. Other limitations of WGS should also be considered: aside from its relatively high cost, WGS introduces more variants of unknown significance than other techniques. Including these variants withhold significant clinical and ethical implications that should be taken into consideration.

3.3 Computational Pipeline for Noninvasive Fetal Variant Calling

Here we review the required steps in the Hoobari-denovo pipeline. As a first step, sequenced reads of all four samples are aligned to the reference genome. The true fetal de novo sample undergoes a separate process for detection of true de novo mutations, later used as the gold standard during the training of a machine learning

model. Second, variant calling of the cfDNA from the maternal plasma sample is performed. This results with a list of genomic loci that present alternate alleles; the parental genomic DNA samples are then genotyped in these loci. Maintaining this order of variant calling is important to ensure that all fetal variant positions are included, especially ones that are absent in the parents and may potentially be de novo mutations. Third, the cfDNA is scanned to identify de novo candidate positions and to retrieve genomic information regarding the reads of positions in question. In this step, cfDNA reads in positions where both parents are homozygous for the reference allele are kept as fetal de novo mutation candidates. In addition, for fetal genotyping purposes, fetal fraction and both fetal and maternal DNA fragments length distribution are calculated. Hoobari's Bayesian genotyping algorithm is then applied on the suspected fetal variant positions, resulting with the fetal genotypes and their probabilities at each variant position. Hoobari's genotyping is a necessary prerequisite step for noninvasive detection of fetal de novo mutations. It allows exploration of the fetal genotype in relation to the parental equivalent positions as two separated entities and not as the mixed origin DNA source originally given to the algorithm as cfDNA. Finally, the list of fetal de novo candidates undergoes additional filtering by a supervised classification machine learning model. This model is pretrained using the results of different families that have undergone the same pipeline. The gold standard fetal genotypes of those families, i.e. the true fetal de novo mutations that are extracted from the pure fetal DNA, are used as the model labels.

3.3.1 Alignment, Deduplication, Indel Realignment

Sequencing whole genome samples provides FASTQ format sequence read files. These files contain raw data of the different reads. These reads undergo a mapping process to the reference genome generating BAM files with aligned reads. This step can be performed by many existing alignment tools. Their compared abilities are elaborated over several studies [29]. In this pipeline, BWA-MEM is used. The next step is to address duplicate reads. Duplicate reads are the result of DNA amplification steps prior to the sample sequencing. They can have a devastating effect on de novo mutations detection due to distortion of read and allele counts at suspected variant positions, leading to both missed de novo positions or wrongful ones. NIPD is especially prone to these issues as amplification steps are usually required for plasma sample analysis due to its limited amount of DNA and due to relatively low depth of coverage. In this pipeline, the software tool chosen for deduplication is Samblaster. Most variant callers are aware of duplication markings and ignore those reads, Hoobari included. Yet in case of in-house code implementation, this capability should be manually added. After aligning and marking duplicate reads, the next step would be to sort and index the BAM file by coordination.

There are many tools available for this task, in particular Samtools and Sambamba used here. Performing additional steps like realignment, local assembly near Indels and base quality recalibration is important for de novo mutation detection. These steps improve accuracy and lower the chances for sequencing-related errors. However, they demand extensive computational resources especially when handling deep sequenced DNA. Although these steps are currently not included in the pipeline presented here, they can and should be integrated (*see* **Note 4**).

3.3.2 Plasma Variant Calling

As the first step of variant analysis, the aligned and deduplicated cfDNA is genotyped. Instead of using the parental positions as a baseline for reviewing the fetal variant positions like Hoobari originally does, a different approach is required. In the de novo case, the cfDNA is first genotyped, without relying on the parental variant positions. Only then, the parents are genotyped in positions that presented an alternate allele in the cfDNA. This ensures that no variant positions in the fetus are missed out only because the parents corresponding positions show no variant, that is, they are homozygous for the reference allele. These exact loci that might be overlooked are the ones considered de novo candidates, as they represent fetal heterozygosity without a Mendelian inheritance support. This part can be performed using Hoobari with manual extra steps, but it is also possible to edit Hoobari's code, to create an automated Hoobari-denovo pipeline (*see* **Note 5**).

The variant caller chosen for this step is Freebayes (*see* Subheading 2.2), but it can be easily replaced by others (*see* **Note 6**). The variant caller requires the cfDNA BAM file and the reference genome file used for reads mapping as input (Command Example 1). This results in a variant call format (VCF) file, containing the potential variants in the cfDNA. Positions in which there is at least one alternate allele will appear in this file. This step results in cfDNA genotype based on the plasma variant calls. To improve the accuracy of this output, the plasma variant calls should be further filtered based on quality control attributes such as the variant quality score and depth of coverage. In some variant calling pipelines, this quality dependent filtering is based on configurable thresholds.

3.3.3 Parental Variant Calling

In the previous step, all potential cfDNA variants were saved in one list. In this step, the parental aligned DNA samples are genotyped; this is performed using the same variant caller. Usually, the parental variant calling returns all positions in which at least one parent is either heterozygous or homozygous for the alternate allele. But, since this pipeline aims to detect potential de novo mutations in the fetus, it is preferred to also genotype positions that are suspected fetal variants based on the plasma. Such variants are absent in the parents, i.e. both parents are homozygous for the reference allele.

In order to achieve that, the cfDNA VCF file generated in the previous step is used as an input to the variant caller along with the parental samples (Command Example 2). Similar to previous steps, this step can be embedded in the Hoobari-denovo pipeline by editing Hoobari's code.

3.3.4 Plasma DNA Preprocessing

The following step helps distinguish between maternal and fetal DNA fragments using the parental genotypes. It includes calculations of both fetal fraction and fragments length distribution. Accurate calculations are key step for optimal outcome in fetal genotyping, which Hoobari-denovo pipeline depends on. These calculations are based on parental genotypes in various positions, which help tracking fragments that are most probably of fetal origin. Positions where both parents are homozygous, each for a different allele, deem the fetal genotype at this position to be heterozygous by Mendelian laws. Hence, cfDNA fragments that present the allele that is absent in the mother can be considered fetal. The fetal fraction calculation is repeated for groups of fragments with similar length, to enable utilizing the fragment length distribution in later stages. The information that is learned in this stage from specific positions is used for the genotyping of the rest of the positions.

3.3.5 Bayesian Algorithm for Variant Calling and Filtering of De Novo Candidates

In previous steps, cfDNA fragments' information was gathered to enable noninvasive fetal genotyping, which is a prerequisite for fetal de novo mutation detection. The cfDNA, as opposed to DNA from a single source, is a mixture of maternal and fetal genome. The maternal to fetal DNA fragments ratio is much favorable to the mother. Hence, only a small amount of fetal DNA is present in the cfDNA. In addition, maternal fragments are generally longer than fetal ones. These distinctive characteristics are not addressed by a regular variant caller. However, Hoobari's Bayesian algorithm for variant calling consists of a dedicated algorithm, which aims to genotype the fetus using a cfDNA sample. The input to this algorithm is the parental genotypes and read-level metadata at fetal candidate positions, as well as the fetal fraction and fragment length distribution. Usually, the parental genotypes are used by Hoobari to calculate the prior probability for each possible fetal genotype. In the case of de novo mutations, the loci of interest are those where both parents are homozygous for the reference allele. In practice, this means that the prior probability is less relevant in de novo analysis, and the prediction is based mainly on the likelihood function, which is based on the cfDNA.

Once the fetal genotyping prediction is complete, the first de novo mutations filtering can occur (Fig. 2). If this step is added internally to Hoobari, running Hoobari-denovo becomes very similar to running Hoobari itself. The input includes the parents and the cfDNA VCF files. The output is a VCF file composed of only the fetal de novo mutation candidates, rather than the entire

predicted fetal genome as Hoobari would provide (Command Example 3). This filter limits the number of fetal variants from several millions to ~100 K fetal de novo mutation candidates. This step aims to provide maximal sensitivity, as far as the depth of coverage enables. However, the specificity is still low after this step, as more than 99.5% of the candidates are false positives, and hence, an additional reduction step is in order.

Variant Processing and Accuracy Assessment

So far, Hoobari's adjusted code provided a list of fetal de novo mutation candidates aiming at maximal sensitivity. High sensitivity goal comes at the cost of low specificity which will be handled in the next steps. In research settings, prior analysis of the pure fetal sample acquired invasively is required to generate the true fetal de novo mutations list. Validating Hoobari-denovo's candidates results contain the true fetal de novo mutations, provides an estimation of the variant caller's sensitivity. Overlooked true de novo mutations can be caused by several reasons, for example insufficient cfDNA depth of coverage and reference genome alignment issues. This step can serve as a good observation checkpoint for both cfDNA and parental features effect on the results. These annotations can be translated to future considerations that would improve the sensitivity of the pipeline.

3.3.6 Machine Learning-Based Variant Recalibration

The previous step reduced the total fetal variants from a few millions to around 100 K de novo mutation candidates, yet the number of true de novo mutation in a human genome is considerably lower; as explained above, it can be estimated as ~74. The large number of potential false positives demands an additional reduction step. A straightforward approach is to use consecutive filtering criteria based on different variant caller features such as depth and variant quality score, i.e. to exclude positions with low coverage quality score. However, these variant features or filter criteria may have dependencies between them. An example most relevant to noninvasive fetal variant calling would be the effect of parental variant information on the fetal genotyping process: technical and biological factors that affect the allelic ratio in heterozygous calls in the parents can have a similar effect in the cfDNA, leading to errors in the fetal genotyping algorithm. This is where machine learning methods can make a great difference. Classification algorithms can model these features while considering their complex relationship with one another. When ingested with the gold standard results, numerous supervised classifiers may improve the accuracy and specificity of the results in comparison to standard techniques. In the context of fetal de novo mutation detection, these algorithms can dramatically reduce the number of false de novo variants.

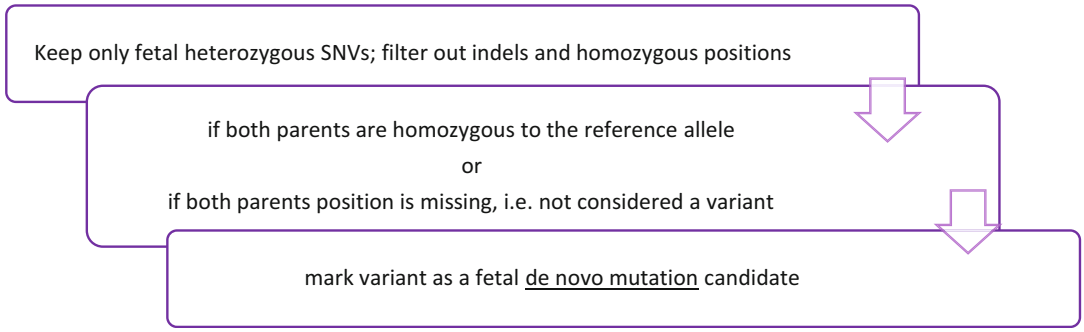


Fig. 2 Each cfDNA entry in the VCF file is looked at. Indels and homozygous positions are filtered out, leaving only entries that represent heterozygous SNVs to the alternate allele. In order to be marked as a fetal de novo mutation candidate, one of two conditions must apply: (1) both parents genotype in the given position is homozygous for the reference allele, hence the alternate allele in the fetus is a de novo change, or (2) the parents genotype in the given position is missing from the variant caller's output as it was not considered a variant, i.e. it assumed the parental genotype is homozygous for the reference allele. The second scenario where the fetal is heterozygous and the parents are homozygous for the alternate allele is less relevant, as the de novo fetal change represents the reference, common allele, which is less likely to be of any clinical significance

Model Training

Supervised classification model training requires raw data and labels marking the true de novo mutations. In this case, the gold standard true fetal de novo mutations extracted from the pure fetal DNA are marked positive, whereas all the other de novo candidates that were eventually found to be false positives are marked negative. At this stage, a model still cannot be straightforwardly trained, due to the nature of de novo mutations; their rare occurrence leads to few dozens of positives per family, at best, and creates a very imbalanced ratio of positives to negatives. To overcome the small number of positives, the model should be fed with as many genome-wide cfDNA analysis results, increasing the amount and diversity of noninvasively detected true fetal de novo mutations. This requirement by itself is a challenge (*see* **Note 7**). Even by doing so, the imbalanced ratio would remain, as each analysis provides only few dozens of true de novo mutations and ~100 K false candidates. To work around this issue, the dataset can undergo additional filtering to further reduce the number of negatives and hopefully make it more balanced. This can be achieved using various features acquired from the variant caller and from Hoobari, for example the genotyping posterior probability. Another option is to randomly sample a smaller number of entries from the negative entries, for example a 1000, maintaining a valid ratio between positives to negatives. Performing many iterations of random 1000 negative entries sampled, enables later model averaging resulting in a single, balanced, and more accurate model.

In general, the model training process is composed of two steps: training and validation, and hence, the training set should

be split to two subsets accordingly. Since de novo mutations are sparse, it is important to verify that they are proportionally split between the training and the validation sets. The validation is essential for evaluating the model and gives room for parameter calibration, and addition of model adjustments and tweaks (*see* **Note 8**). To prevent overfitting, changes to the model and its parameters are only possible during this step, before applying it on the actual test set. Once the model is optimized, it is tested over a new dataset composed of another family that is not present in the training or the validation sets. The test indicates the actual model accuracy. Technically, the model training can be performed with different available programming packages, for example Scikit-learn for Python (as described in Subheading 2.2).

In the pipeline described here, a random forest algorithm is used for classification. The building block for this algorithm is a decision tree, which is basically a sequence of questions, filters and thresholds of the data. The algorithm constructs multiple decision trees and averages them to one model. The trained model is the last step in Hoobari-denovo pipeline, enabling de novo mutation detection for genome-wide NIPD. The diversity and generalizability of the model increases as more genome wide NIPD data assembles are ingested to the model after invasively validated. Such diverse, generic model can then be used in future cfDNA analyses, effectively speculating the probability of each fetal de novo candidate to be a true de novo mutation. These probabilities eventually enable pregnant women and their physicians to understand the chance that a mutation was indeed identified, and whether they should validate it using a subsequent invasive test.

4 Notes

Noninvasive prenatal testing has developed tremendously in the last decade. At the beginning, it was mostly studied in the context of screening of chromosome level disorders, such as trisomy and sex-chromosome abnormalities. Later on, it was utilized for detection of point mutations of paternal and maternal origin, and even de novo mutations. This requires a high resolution, that is available thanks to the introduction of NGS, providing a significantly cheaper, quicker, and more accurate form of human genome sequencing. However, genome-wide detection of fetal de novo mutations still requires a relatively deep coverage. As sequencing technologies keep on improving and provide more accurate results, while their cost becomes lower, the coverage needed for fetal de novo mutation may soon be more accessible.

Here we reviewed the methods used for genome-wide noninvasive prenatal detection of de novo mutations and suggested a novel pipeline. This pipeline is based on a recently presented

noninvasive fetal genotyping tool, named Hoobari, together with a machine learning-based supervised classification model. Adjustments made to Hoobari for the sake of the de novo pipeline were easily achieved thanks to the modular nature of this tool. This pipeline can also handle specific regions in the genome, and its principles can be implemented in related research fields, for example ctDNA analysis in liquid biopsies.

Throughout this guide, few key points in Hoobari-denovo pipeline are emphasized leaving room for future bioinformaticians to improve and optimize, including the following:

1. Family trio de novo mutation detection has been long researched. There are several pipelines containing various parameters, implementing different approaches as described in Subheading 3.1.2, yet none of them had become the field standard. This should be taken into consideration when trying to generate a true fetal de novo mutations list out of the pure fetal DNA. It is recommended to choose a well-trusted pipeline based on experience, but also to test different pipelines and possibly combine them to one assembly method.
2. There are parts of the pipeline described here that can undergo additional tests and optimization. Few examples would be fetal fraction calculation, as well as mapping and realignment techniques, and some of the technical steps prior to the de novo mutation analysis.
3. NGS-based analysis is currently more reliable in detecting SNVs than it would detect indels and copy number variants (CNVs). For this reason and others, this pipeline addresses point de novo mutations only. We encourage researchers to further expand the scope of this pipeline and to provide a more comprehensive solution for the genome-wide noninvasive detection of prenatal de novo mutations.
4. WGS in general and under NIPD's deep sequencing requirements especially, generates large amounts of data at the form of BAM files. Storing, reading and manipulating such large BAM files can be costly in terms of computational resources. Some of the best practices for handling these large files are utilizing advanced techniques like streaming and parallelization. Many of the available bioinformatics tools support these methods and enable smooth transition of large amounts of data between one another, yet it is an important aspect to consider when selecting one.
5. Hoobari-denovo pipeline withholds several steps dedicated for de novo mutation detection. These steps can either be performed manually using Hoobari or be easily integrated into Hoobari's code which is freely available. Integrating this logic into Hoobari's code may provide a smoother and more robust

de novo pipeline that can be further packaged as standalone tool.

6. The Hoobari-denovo pipeline was designed modularly so that its steps can be replaced if need be by ever improving algorithms and external tools. Some steps in the pipeline are more generic than others and can be easily achieved with other existing tools. For instance, there are other variant callers that can replace Freebayes.
7. Studying the detection process of de novo mutations is limited by availability of genetic data from family trios. The limitation is even greater when attempting to address NIPD based trios, that is, trios that include the parents, the maternal plasma sample and a pure fetal sample. Building a trained, diverse and generalized classifying model for de novo mutation detection requires a large number of such trios. To achieve this, a large collaboration would be required, in an effort to establish a dedicated database containing NIPD-based trios.
8. Optimizing a machine learning model is an important step in any model training. It starts with data selection: selecting the most representative but not biased training and validation sets, and further includes features selection, algorithm tuning, averaging models, and so forth. This guide sets the foundation to building a supervised classifier model for de novo mutations detection in NIPD analysis yet does not elaborate on such optimization techniques. Readers are encouraged to explore this powerful platform and get more familiar with its optimization best practices currently available.

References

1. Conrad DF, Keebler JEM, DePristo MA et al (2011) Variation in genome-wide mutation rates within and between human families. Nat Genet 43:712–714. https://doi.org/10.1038/ng.862
2. Veltman JA, Brunner HG (2012) De novo mutations in human genetic disease. Nat Rev Genet 13:565–575. https://doi.org/10.1038/nrg3241
3. Hayward J, Chitty LS (2018) Beyond screening for chromosomal abnormalities: advances in non-invasive diagnosis of single gene disorders and fetal exome sequencing. Semin Fetal Neonatal Med 23:94–101. https://doi.org/10.1016/j.siny.2017.12.002
4. Akolekar R, Beta J, Picciarelli G et al (2015) Procedure-related risk of miscarriage following amniocentesis and chorionic villus sampling: a systematic review and meta-analysis. Ultrasound Obstet Gynecol 45:16–26. https://doi.org/10.1002/uog.14636
5. Tabor A, Alfirevic Z (2010) Update on procedure-related risks for prenatal diagnosis techniques. Fetal Diagn Ther 27:1–7. https://doi.org/10.1159/000271995
6. Lo YMD, Lun FMF, Chan KCA et al (2007) Digital PCR for the molecular detection of fetal chromosomal aneuploidy. Proc Natl Acad Sci 104:13116. https://doi.org/10.1073/pnas.0705765104
7. Fan HC, Blumenfeld YJ, Chitkara U et al (2008) Noninvasive diagnosis of fetal aneuploidy by shotgun sequencing DNA from maternal blood. Proc Natl Acad Sci 105:16266. https://doi.org/10.1073/pnas.0808319105
8. Hill M, Compton C, Lewis C et al (2012) Determination of foetal sex in pregnancies at risk of haemophilia: a qualitative study

exploring the clinical practices and attitudes of health professionals in the United Kingdom. Haemophilia 18:575–583. https://doi.org/10.1111/j.1365-2516.2011.02653.x

9. Lewis C, Hill M, Skirton H, Chitty LS (2012) Non-invasive prenatal diagnosis for fetal sex determination: benefits and disadvantages from the service users' perspective. Eur J Hum Genet 20:1127–1133. https://doi.org/10.1038/ejhg.2012.50
10. Lam K-WG, Jiang P, Liao GJW et al (2012) Noninvasive prenatal diagnosis of monogenic diseases by targeted massively parallel sequencing of maternal plasma: application to β-thalassemia. Clin Chem 58:1467–1475. https://doi.org/10.1373/clinchem.2012.189589
11. Chitty LS, Mason S, Barrett AN et al (2015) Non-invasive prenatal diagnosis of achondroplasia and thanatophoric dysplasia: next-generation sequencing allows for a safer, more accurate, and comprehensive approach. Prenat Diagn 35:656–662. https://doi.org/10.1002/pd.4583
12. Rabinowitz T, Polsky A, Golan D et al (2019) Bayesian-based noninvasive prenatal diagnosis of single-gene disorders. Genome Res 29:428–438. https://doi.org/10.1101/gr.235796.118
13. Kitzman JO, Snyder MW, Ventura M et al (2012) Non-invasive whole genome sequencing of a human fetus. Sci Transl Med 4:137ra76. https://doi.org/10.1126/scitranslmed.3004323
14. Chan KCA, Jiang P, Sun K et al (2016) Second generation noninvasive fetal genome analysis reveals de novo mutations, single-base parental inheritance, and preferred DNA ends. Proc Natl Acad Sci 113:E8159–E8168. https://doi.org/10.1073/pnas.1615800113
15. Zhang J, Li J, Saucier JB et al (2019) Non-invasive prenatal sequencing for multiple Mendelian monogenic disorders using circulating cell-free fetal DNA. Nat Med 25:439–447. https://doi.org/10.1038/s41591-018-0334-x
16. DePristo MA, Banks E, Poplin RE et al (2011) A framework for variation discovery and genotyping using next-generation DNA sequencing data. Nat Genet 43:491–498. https://doi.org/10.1038/ng.806
17. O'Fallon BD, Wooderchak-Donahue W, Crockett DK (2013) A support vector machine for identification of single-nucleotide polymorphisms from next-generation sequencing data. Bioinformatics 29:1361–1366. https://doi.org/10.1093/bioinformatics/btt172
18. Yadong Wang YL (2014) A gradient-boosting approach for filtering de novo mutations in parent–offspring trios. Bioinformatics 30 (13):1830–1836. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4071207/. Accessed 4 Mar 2019
19. Wan N, Weinberg D, Liu T-Y et al (2018) Machine learning enables detection of early-stage colorectal cancer by whole-genome sequencing of plasma cell-free DNA. Cancer Biol
20. Ramu A, Noordam MJ, Schwartz RS et al (2013) DeNovoGear: de novo indel and point mutation discovery and phasing. Nat Methods 10:985–987. https://doi.org/10.1038/nmeth.2611
21. Ashoor G, Syngelaki A, Poon LCY et al (2013) Fetal fraction in maternal plasma cell-free DNA at 11–13 weeks' gestation: relation to maternal and fetal characteristics. Ultrasound Obstet Gynecol 41:26–32. https://doi.org/10.1002/uog.12331
22. Hu P, Liang D, Chen Y et al (2019) An enrichment method to increase cell-free fetal DNA fraction and significantly reduce false negatives and test failures for non-invasive prenatal screening: a feasibility study. J Transl Med 17:124. https://doi.org/10.1186/s12967-019-1871-x
23. Sillence K (2016) Cell-free fetal DNA (cffDNA) enrichment for non-invasive prenatal testing (NIPT): a comparison of molecular techniques
24. Jorgez CJ, Bischoff FZ (2009) Improving enrichment of circulating fetal DNA for genetic testing: size fractionation followed by whole gene amplification. Fetal Diagn Ther 25:314–319. https://doi.org/10.1159/000235877
25. Webb A, Madgett T, Miran T et al (2012) Non invasive prenatal diagnosis of aneuploidy: next generation sequencing or fetal DNA enrichment? Balk J Med Genet BJMG 15:17–26. https://doi.org/10.2478/v10034-012-0013-z
26. Peng XL, Jiang P (2017) Bioinformatics approaches for fetal DNA fraction estimation in noninvasive prenatal testing. Int J Mol Sci 18:453. https://doi.org/10.3390/ijms18020453
27. Kong A, Frigge ML, Masson G et al (2012) Rate of de novo mutations and the importance of father's age to disease risk. Nature 488:471–475. https://doi.org/10.1038/nature11396
28. Arnheim N, Calabrese P (2009) Understanding what determines the frequency and pattern

of human germline mutations. Nat Rev Genet 10:478–488. https://doi.org/10.1038/nrg2529

29. Shang J, Zhu F, Vongsangnak W et al (2014) Evaluation and comparison of multiple aligners for next-generation sequencing data analysis. Biomed Res Int 2014:1–16. https://www.hindawi.com/journals/bmri/2014/309650/abs/. Accessed 8 Jan 2020
30. Li H, Durbin R (2009) Fast and accurate short read alignment with Burrows–Wheeler transform. Bioinformatics 25:1754–1760. https://doi.org/10.1093/bioinformatics/btp324
31. Faust GG, Hall IM (2014) SAMBLASTER: fast duplicate marking and structural variant read extraction. Bioinformatics 30:2503–2505. https://doi.org/10.1093/bioinformatics/btu314
32. Li H, Handsaker B, Wysoker A et al (2009) The Sequence Alignment/Map format and SAMtools. Bioinformatics 25:2078–2079. https://doi.org/10.1093/bioinformatics/btp352
33. Tarasov A, Vilella AJ, Cuppen E et al (2015) Sambamba: fast processing of NGS alignment formats. Bioinformatics 31:2032–2034. https://doi.org/10.1093/bioinformatics/btv098
34. Garrison E (2012) freebayes: Bayesian haplotype-based genetic polymorphism discovery and genotyping
35. Danecek P, Auton A, Abecasis G et al (2011) The variant call format and VCFtools. Bioinformatics 27:2156–2158. https://doi.org/10.1093/bioinformatics/btr330
36. Buitinck L, Louppe G, Blondel M, et al (2013) API design for machine learning software: experiences from the scikit-learn project. ArXiv13090238 Cs

Chapter 13

Accurate Imputation of Untyped Variants from Deep Sequencing Data

Davoud Torkamaneh and François Belzile

Abstract

The quality, statistical power, and resolution of genome-wide association studies (GWAS) are largely dependent on the comprehensiveness of genotypic data. Over the last few years, despite the constant decrease in the price of sequencing, whole-genome sequencing (WGS) of association panels comprising a large number of samples remains cost-prohibitive. Therefore, most GWAS populations are still genotyped using low-coverage genotyping methods resulting in incomplete datasets. Imputation of untyped variants is a powerful method to maximize the number of SNPs identified in study samples, it increases the power and resolution of GWAS and allows to integrate genotyping datasets obtained from various sources. Here, we describe the key concepts underlying imputation of untyped variants, including the architecture of reference panels, and review some of the associated challenges and how these can be addressed. We also discuss the need and available methods to rigorously assess the accuracy of imputed data prior to their use in any genetic study.

Key words GWAS, Genotyping, Reference panel, Genotype imputation, Imputation accuracy, Imputation, Untyped variants, Deep sequencing, NGS data analysis

1 Motivation

Genome-wide association studies (GWAS) have revolutionized plant and animal research by allowing scientists to pinpoint genomic regions (and sometimes genes) controlling traits of interest. In the last decade, GWAS studies have led to new discoveries about genes and pathways involved in complex traits, have provided substantial novel biological insights, have led to the development of diagnostic kits and biomarkers, and has facilitated basic research in plant and animal genetics and genomics [1]. GWAS relies on direct or indirect associations between common genetic variants (commonly defined as variants whose minor allele frequency (MAF) $\geq$ 1%) at different locations in the genome (Fig. 1). Genetic variants are broadly classified into two major categories: nucleotide and structural variants. Nucleotide variants are defined as

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_13,

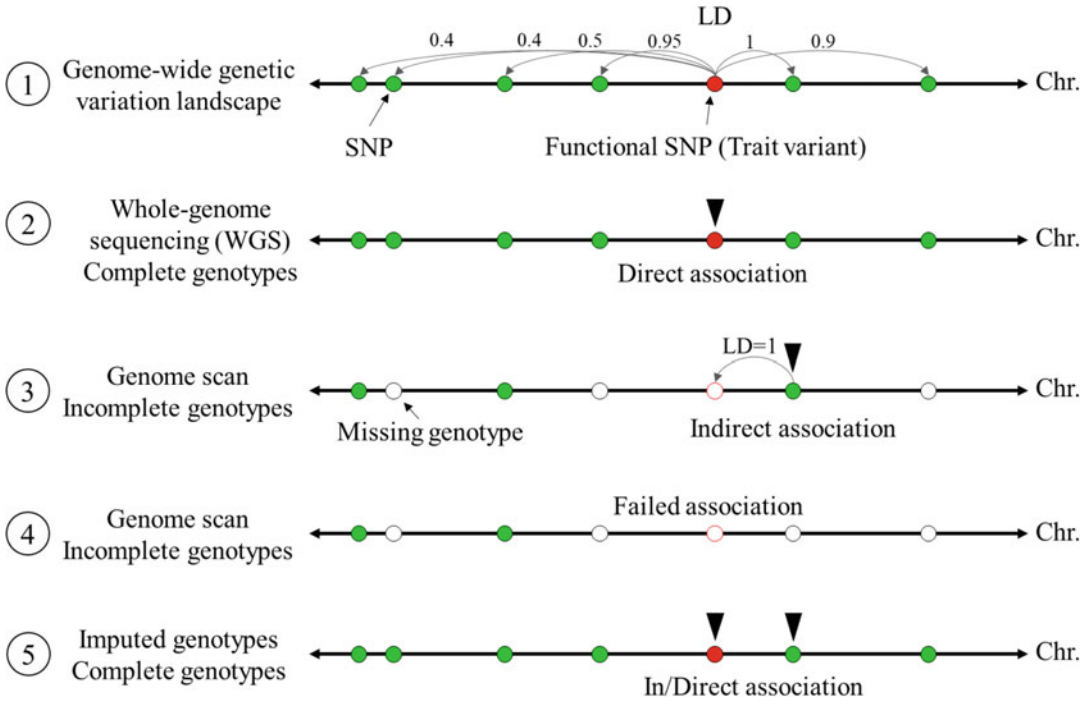


Fig. 1 Schematic representation of direct and indirect association detected through GWAS analysis based on LD and the impact of missing genotypes on the power of detection of an association

encompassing single or multiple nucleotide variants (SNPs, MNPs) and small insertions/deletions (indels), all of which generally encompass less than 50 bp, whereas structural variants (SVs) represent larger rearrangements of various types (deletions, insertions, inversions, translocations, duplications, and copy number variations [CNVs]) [1]. The statistical power and resolution of GWAS studies to detect trait variants (functional SNPs or causal genetic variants) is positively correlated with the number of available genetic variants across the genome, the pattern of linkage disequilibrium (long- and short-distance LD) and LD decay (gradual reduction in LD based on physical distance between loci as a result of recombination over time) [2].

With the advent of next-generation sequencing (NGS)-based genotyping methods, discovery of genetic variants (commonly single-nucleotide polymorphisms (SNPs)) has become much more accessible, affordable and has therefore been adopted for GWAS studies in large numbers and on large sets of individuals [3]. NGS-based genotyping methods encompass four main categories. *Whole-genome sequencing (WGS)* is the sequencing of the entire genome of individuals of the same species to a low (1–10×), medium (10–30×) or high (≥30×) depth of coverage. WGS allows for the discovery and genotyping of both nucleotide and structural variants) [4]. *Low-pass (skim) sequencing* is typically used to

sequence the entire genome, albeit incompletely, of large sets of individuals with an ultra-low coverage ($\leq 1\times$)) [5]. *SNP arrays* are high-density oligo arrays containing up to several million probes of short length, which allow for the genotyping of hundreds of thousands of "selected" SNPs across the genome, in a single reaction, on a large set of individuals [6]. Typically, WGS or sequencing of the transcribed portion of the genome (exome) of a set of representative samples is used to discover the SNP loci that will be interrogated using the resulting array. *Reduced-representation sequencing* methods aim to identify polymorphisms within a relatively constant subset of genomic regions for which there is not necessarily a priori knowledge regarding polymorphism. The subset of the genome that is characterized through sequencing can either be chosen by the experimenter, in the case of sequence capture and targeted amplicon sequencing (TAS) or represent a somewhat random sample as is the case in genotyping-by-sequencing (GBS) [3]. Despite the significant reductions in cost of whole-genome sequencing (WGS) experienced over the last few years, most GWAS populations are still genotyped using low-pass (skim) sequencing (e.g., $0.1–1\times$), SNP arrays or reduced-representation sequencing approaches [7]. The resulting catalogues of variants derived from these methods are inevitably incomplete. Therefore, researchers need to impute untyped or missing variants using a reference panel (known as a HapMap; a comprehensive catalog of common genetic variants within a population) before performing GWAS to maximize the number of SNPs identified in study samples [8].

In general, there are two main reasons to impute untyped variants (ones that were not directly genotyped in the study sample): (1) almost all large-scale genotyping efforts are based on genome scans and are highly likely to produce incomplete data and (2) the imputation of untyped variants may help fine map functional (causal) variants [9]. The results of GWAS studies using incomplete data can be biased and association could be missed due to the pattern of LD around the trait variants. Therefore, for a perfect or near-perfect coverage of the genome, imputation of untyped variants is required [10]. Here, we describe the key concepts underlying imputation of untyped variants, including the architecture of reference panels, tools and approaches for imputation, and the statistical methods for estimation of accuracy and sensitivity of imputation of untyped variants.

2 Characteristics of a Reference Panel

Genotype imputation is a well-established statistical technique where a reference panel of individuals genotyped at a high-density set of SNPs is used to infer untyped variants (unobserved genotypes) within study samples genotyped at a subset of these sites

[11]. The concept of genotype imputation is an extension of the haplotype-phasing algorithm where the model exploits known contiguous LD patterns and haplotype frequencies for a set of alleles lying on specific genomic regions of a reference panel and compares to study samples to find perfect or near-perfect matches [12]. Matched haplotype pattern(s) will be used to estimate unobserved genotypes in the study samples and based on the haplotype overlap, a score (probability of match) will be assigned to each imputed variant [13].

Nowadays, genetic variation resources (e.g., HapMaps) have been produced for many species capturing species-wide genetic diversity. These publicly available resources provide powerful reference panels for the imputation of untyped variants. However, in this context, two important questions arise. (1) How to estimate the quality of a reference panel for high-quality imputation of untyped variants [14] and (2) how to choose the best reference panel from different panels [15]. To answer these questions, here, we describe different characteristics of a high-quality reference panel based on extensiveness, population structure, and the inclusion of structural variants.

2.1 Extensiveness

Previous studies have documented that the accuracy of imputation of untyped variants increases as the size of a reference panel increases [13]. Assessing the extensiveness of a reference panel only on the basis of the number of individuals within the panel is a naïve generalization. In principle, the extensiveness of a reference panel should be defined by the comprehensiveness of nucleotide and haplotype diversity. By this definition, an extensive reference panel exhaustively captures common genetic variants and at least half of rare variants (MAF $< 1\%$) [16]. One would assume, in this fashion, that the number of SNPs has plateaued and that adding new individuals in the reference panel will not increase (significantly) the number of SNPs. On the other hand, the extent of LD (pairwise LD using both r^2 and D′) can provide a measure of haplotype diversity within a reference panel. Based on LD extent (long-range LD (LD decay $\geq$10 kb) and short-range LD (LD decay $<$10 kb)), haplotype-based tag SNPs can be detected [17]. A tag SNP is a representative SNP that is in very high LD with a larger group of SNPs in a region of the genome and thus contributes to capturing haplotypes. The saturation of the number of tag SNPs can be used as a sign of the comprehensiveness of haplotype diversity and consequently of the extensiveness of a reference panel. As a conclusion, in an extensive reference panel adding new individuals will not increase the number of SNPs or tag SNPs in a significant fashion.

2.2 Population Structure

The underlying assumption of an imputation algorithm is that the reference panel and the study samples are derived from the same broader population. In this respect, the assumption is that both populations share a similar pattern of LD, similar distribution of haplotypes, and similar population structure [18]. However, in the real world, different populations have experienced distinct demographic histories with different rates of migration and admixture events. Consequently, the presence of distinctive population structure within the study samples and the reference panel may result in differences in LD patterns and decreased accuracy of imputation [15]. Therefore, a careful consideration is required for the use of reference panels in the design of strategies for imputation of untyped variants. In general, reference panels are divided into two categories of (1) all-in-one reference panel (a large set of individuals representative of a species with a balanced representation of genetic/haplotype diversity and population structure) and (2) restricted reference panel (a limited reference panel representative of genetic/haplotype diversity and population structure of a specific population or geographical region) [19]. As a result, for a high-quality imputation of untyped variants, an appropriate reference panel based on study samples population structure should be used.

2.3 Structural Variants

Structural variants (SVs) represent an important class of genetic variation. Although these are typically much fewer in number than SNPs, their much greater size results in them accounting for a greater number of variable nucleotides than SNPs. Unfortunately, the methods for their discovery, genotyping, and imputation have lagged behind those for SNPs [20]. SVs affect LD pattern and complexify the characterization of haplotypes [21]. As genotype imputation algorithms are based on LD pattern and haplotype structure, any alteration of these elements will dramatically affect the accuracy of imputation. Moreover, SVs (e.g., large deletions) are most often wrongly imputed as untyped variants [22]. To date, SVs have only been integrated in the human reference panel which then enabled their correct imputation [23]. An extended reference panel including SVs will revolutionize the concept of imputation of untyped variants and will allow for more accurate imputation of SVs based on their SNP genotype status. This approach is still in its infancy and requires further improvement, at first, in the accurate and precise discovery and genotyping of SVs, efficient integration of SVs in a reference panel, accurate phasing of genotyped SVs, and efficient computational framework.

Table 1
List of most common genotype imputation tools

Tool	Type	Model[a]	Source	Citations[b]	First release	References
IMPUTE2	2-step	HMM	http://mathgen.stats.ox.ac.uk/impute/impute_v2.html	592	2011	[26]
Beagle5	1-step	HMM	http://faculty.washington.edu/browning/beagle/beagle.html	1400	2007	[25]
minimac4	1-step	HMM	https://genome.sph.umich.edu/wiki/Minimac4	439	2016	[16]
MaCH	2-step	MaCH	http://csg.sph.umich.edu/abecasis/MACH/download/	1400	2010	[27]
fastPHASE	1-step	HMM	http://scheet.org/software.html	1200	2006	[24]

[a]HMM: Hidden Markov model, MaCH: Markov chain haplotyping
[b]Citation count by November 2019

3 Methods for Imputing Untyped Variants

To date, to achieve accurate imputation of untyped variants, several phasing and imputation tools have been developed. Current state-of-the-art untyped variant imputation tools (Table 1) have been developed based on a hidden Markov model (HMM) and expectation–maximization (EM) algorithms [24]. The HMMs have a simple probabilistic structure which made them extremely attractive for missing-data imputation tools. A simple probabilistic structure results in a relatively parsimonious model and facilitates computation of large sets of genotypic data. In an HMM, an underlying hidden state (the haplotype phase and the true genotypes from a reference panel) generates the observed data (observed unphased genotypes including errors and missing data) [25]. In this model, specialized algorithms are used to find the most likely hidden state paths (e.g., Viterbi algorithm), to compute posterior probabilities of hidden states (e.g., Baum forward–backward algorithm), and to fit model parameters by maximizing the likelihood (e.g., Baum–Welch algorithm) [13]. Based on the requirement for a prephasing step (inferring haplotype structure from information derived from the study sample, reference panel, and recombination rate), tools for imputation of untyped variants are divided into two categories: one-step tools (e.g., Beagle) and two-step tools (e.g., IMPUTE2). Some of the two-step tools are limited only to model organisms due to the prerequisite of recombination rates (recombination map) and mutation rates for prephasing step.

Even though all these tools exploit differently an HMM to impute untyped variants from a reference panel, they follow a similar set of steps: (1) fit the model (estimating full data with haplotype phase and all genotypes to estimate the other parameters of the model, such as recombination fractions), (2) assign a particular value (probability) to a missing genotype which is dependent on the probabilities of its haplotypes belonging to particular clusters (overlapping sliding windows), and the frequencies of the observed genotypes (allele count) within those clusters (dependent on identity-by-descent (IBD) and identity-by-state (IBS), (3) statistical assessment of the precision of the prediction based on different iterations of the algorithm, and, finally, (4) selecting the greatest probability as an estimate for the untyped variants.

4 Estimating the Accuracy of Imputation of Untyped Variants

The accuracy of imputation of untyped variants refers to the percentage of untyped variants that are correctly imputed [17]. Typically, not all markers can be accurately imputed. Multiple factors can affect imputation accuracy. (1) *Extensiveness of reference panel.* A large and comprehensive reference panel provides a larger set of haplotypes to match against and provides a higher level of genetic similarity with study samples, which generally increases imputation accuracy [28, 29] (2) *Marker density of study samples.* A study population with a higher marker density increases the number of sites to match with. An increased number of matched sites improves the probability of finding shared haplotype segments [16]. (3) *Allele frequency of untyped variant in the reference panel.* In general, imputation of common variants is easier than rare variants. Establishment of haplotype background for rare variants is harder, since they have only been observed a few times in the reference panel [16]. (4) *Accuracy of haplotypes in both reference panel and study samples.* Imputation of untyped variants depends on the precision of shared haplotype segments in the reference panel and study samples. Errors in haplotypes result in breaking up of shared haplotype segments and decreased accuracy of imputation [30].

Therefore, the assessment of the accuracy of imputation of untyped variants is a crucial step before using imputed data in any genetic study. To date, several different approaches have been proposed to help identify well-imputed variants. In general, the accuracy of imputation of untyped variants can be assessed through five different approaches: (1) the overall average of statistical probabilities generated by imputation tools; (2) in silico masking (setting to "missing" a small proportion of the called genotypes) and attempting to impute them; (3) genotyping the reference

genome individual in the study population, imputing and comparing the resulting imputed genotypes to the reference; (4) leave-one-out design; and (5) the whole-genome sequencing of some of study samples.

4.1 Statistical Probabilities

The average probability (from 0 to 100%) that an imputed genotype call is correct, based on probabilistic values generated by imputation tools, can be used as a measure of imputation accuracy [16]. Unfortunately, this simple value is meaningless when variants with different allele frequencies are compared. It is possible to achieve 90% accuracy for a rare variant (allele frequency of <5%) by simply assigning the most common genotype to every individual. The second measure is based on Hardy–Weinberg equilibrium and calculated by comparing the variance in a set of imputed allele counts to theoretical expectations (typically, inaccurately imputed variants show less variability than expected based on allele frequency) [31]. This method is typically expressed as an r^2 coefficient and tries to capture the correlation between imputed and true genotypes. Despite the fact that such types of measures resting on a statistical framework are attractive, they are not recommended as a real and final measure of imputation accuracy.

4.2 Mask, Reimpute, and Compare

It is possible to mask (randomly remove) a subset of the variants (hundreds to thousands) from available genotype data (e.g., several samples from a reference panel) and to run imputation of untyped variants on this panel with masked variants. It is then possible to compare the result of imputation at such masked genotypes with the actual genotype observed prior to masking to assess imputation accuracy [32]. Despite being a strong and reliable method that is widely used in literature, it requires careful planning and implementation. Additionally, in many cases, the selection of variants was biased by selecting common variants [33]. Furthermore, masking, reimputing, and comparing is a time-consuming process and requires attention.

4.3 Genotyping the Individual from Which the Reference Genome Was Obtained

The accuracy of imputation of untyped variants can be estimated as the degree of concordance between the genotypes obtained for the individual from which the reference genome was derived. Typically, one would expect the called and imputed genotypes at all variable positions (in the population) to be identical to those indicated in the reference. Any observed differences will reflect the error rate in the original genotype calling and imputation [34]. This is an efficient, cost-effective, and robust approach that can be used to assess the accuracy of SNP calls from either the entire catalogue of SNPs (imputed + unimputed variants), the unimputed genotypes alone (determining the accuracy of genotyping), or only the imputed genotypes.

4.4 Leave-One-Out

In some studies, there is an overlap between study samples and individuals of the reference panel. In this scenario, the imputation accuracy of untyped variants can be estimated using a leave-one-out design. In each leave-one-out round, imputation can be performed after removing the genotypic data from the reference panel for one of the shared individuals (present in both the study sample and reference panel) [35]. Following imputation on the study sample, it is possible to assess the degree of concordance between the imputed genotypes in the individual that had been left out of the reference panel and the known genotypes.

4.5 Whole-Genome Sequencing

The most expensive approach is to perform whole-genome sequencing of at least one sample from study samples. Then, the imputation accuracy of untyped variants will be measured as the percentage of correct genotypes, and the correlation between the true genotypes and imputed dosages.

5 Conclusion

Over the past decade, imputation of missing and untyped variants has become a key step in the analysis of massively parallel NGS data, enabling researchers to analyze large samples and dissect the genetic basis of complex traits. Imputation of untyped variants allows to accurately evaluate the evidence for association of genetic markers that are not directly genotyped, increases the power of GWAS, increases the resolution in fine-mapping studies and allows one to combine data derived from different genotyping platforms for meta-analysis. Despite being an essential and powerful method, imputation is not without challenges and, accordingly, the accuracy of the result must be measured before using imputed data in any genetic study. In the light of experience gained over the past few years, the imputation of untyped variants in future studies is expected to be more efficient and accurate and to become an integral part in day-to-day genetic studies than ever before.

Acknowledgments

This work was supported by the SoyaGen grant (www.soyagen.ca) awarded to F. Belzile and funded by Génome Québec, Genome Canada, the government of Canada, the Ministère de l'Économie, Science et Innovation du Québec, Semences Prograin Inc., Syngenta Canada Inc., Sevita Genetics, Coop Fédérée, Grain Farmers of Ontario, Saskatchewan Pulse Growers, Manitoba Pulse & Soybean Growers, the Canadian Field Crop Research Alliance, and Producteurs de grains du Québec.

References

1. Visscher PM, Wray NR, Zhang Q et al (2017) 10 years of GWAS discovery: biology, function, and translation. Am J Hum Genet 101 (1):5–22. https://doi.org/10.1016/j.ajhg.2017.06.005
2. Bush WS, Moore JH (2012) Chapter 11: genome-wide association studies. PLoS Comput Biol 8(12):e1002822. https://doi.org/10.1371/journal.pcbi.1002822
3. Torkamaneh D, Boyle B, Belzile F (2018) Efficient genome-wide genotyping strategies and data integration in crop plants. Theor Appl Genet 131:499–511. https://doi.org/10.1007/s00122-018-3056-z
4. Huang X, Feng Q, Qian Q et al (2009) High-throughput genotyping by whole-genome resequencing. Genome Res 19 (6):1068–1076. https://doi.org/10.1101/gr.089516.108
5. Golicz AA, Bayer PE, Edwards D (2015) Skim-based genotyping by sequencing. Methods Mol Biol 1245:257–270
6. Rasheed A, Hao Y, Xia X et al (2017) Crop breeding chips and genotyping platforms: progress, challenges, and perspectives. Mol Plant 10:1047–1064. https://doi.org/10.1016/j.molp.2017.06.008
7. Tam V, Patel N, Turcotte M et al (2019) Benefits and limitations of genome-wide association studies. Nat Rev Genet 20(8):467–484
8. Halperin E, Stephan DA (2009) SNP imputation in association studies. Nature Biotechnol 4:349–351
9. Wang Z, Chatterjee N (2017) Increasing mapping precision of genome-wide association studies: to genotype and impute, sequence, or both? Genome Biol 18(1):118. https://doi.org/10.1186/s13059-017-1255-6
10. Guinot F, Szafranski M, Ambroise C et al (2018) Learning the optimal scale for GWAS through hierarchical SNP aggregation. BMC Bioinformatics 19:459. https://doi.org/10.1186/s12859-018-2475-9
11. Naj AC (2019) Genotype imputation in genome-wide association studies. Nat Rev Curr Protoc Hum Genet 102(1):e84. https://doi.org/10.1002/cphg.84
12. Li Y, Willer C, Sanna S et al (2009) Genotype imputation. Annu Rev Genomics Hum Genet 10:387–406. https://doi.org/10.1146/annurev.genom.9.081307.164242
13. Browning BL, Zhou Y, Browning SR (2018) A one-penny imputed genome from next-generation reference panels. Am J Hum Genet 103(3):338–348. https://doi.org/10.1016/j.ajhg.2018.07.015
14. Vergara C, Parker MM, Franco L et al (2018) Genotype imputation performance of three reference panels using African ancestry individuals. Hum Genet 137(4):281–292. https://doi.org/10.1007/s00439-018-1881-4
15. Zhang P, Zhan X, Rosenberg NA et al (2013) Genotype imputation reference panel selection using maximal phylogenetic diversity. Genetics 195(2):319–330. https://doi.org/10.1534/genetics.113.154591
16. Das S, Abecasis GR, Browning BL (2018) Genotype imputation from large reference panels. Annu Rev Genomics Hum Genet 19:73–96
17. Marchini J, Howie B (2010) Genotype imputation for genome-wide association studies. Nat Rev Genet 11:499–511
18. Bai WY, Zhu XW, Cong PK et al (2019) Genotype imputation and reference panel: a systematic evaluation on haplotype size and diversity. Brief Bioinform bbz108. https://doi.org/10.1093/bib/bbz108
19. Howie B, Marchini J, Stephens M (2011) Genotype imputation with thousands of genomes. G3 1:457–470
20. Ho SS, Urban AE, Mills RE (2019) Structural variation in the sequencing era. Nat Rev Genet. https://doi.org/10.1038/s41576-019-0180-9
21. Conrad DF, Hurles ME (2007) The population genetics of structural variation. Nat Genet 39(7 Suppl):S30–S36. https://doi.org/10.1038/ng2042
22. Tardivel A, Torkamaneh D, Lemay MA et al (2019) A systematic gene-centric approach to define haplotypes and identify alleles on the basis of dense single nucleotide polymorphism datasets. Plant Genome 12:180061. https://doi.org/10.3835/plantgenome2019.01.0061
23. Hehir-Kwa JY, Marschall T, Kloosterman, et al. (2016) A high-quality human reference panel reveals the complexity and distribution of genomic structural variants. Nat Commun 7:12989. https://doi.org/10.1038/ncomms12989
24. Scheet P, Stephens M (2006) A fast and flexible statistical model for large-scale population genotype data: applications to inferring missing genotypes and haplotypic phase. Am J Hum Genet 78(4):629–644. https://doi.org/10.1086/502802

25. Browning SR, Browning BL (2007) Rapid and accurate haplotype phasing and missing-data inference for whole-genome association studies by use of localized haplotype clustering. Am J Hum Genet 81(5):1084–1097. https://doi.org/10.1086/521987
26. Howie BN, Donnelly P, Marchini J (2009) A flexible and accurate genotype imputation method for the next generation of genome-wide association studies. PLoS Genet 5(6): e1000529
27. Li Y, Willer CJ, Ding J et al (2010) MaCH: using sequence and genotype data to estimate haplotypes and unobserved genotypes. Genet Epidemiol 34(8):816–834. https://doi.org/10.1002/gepi.20533
28. Loh PR, Danecek P, Palamara PF et al (2016) Reference-based phasing using the Haplotype Reference Consortium panel. Nat Genet 48 (11):1443–1448. https://doi.org/10.1038/ng.3679
29. Zhang B, Zhi D, Zhang K et al (2011) Practical consideration of genotype imputation: sample size, window size, reference choice, and untyped rate. Stat Interface 4(3):339–352. https://doi.org/10.4310/sii.2011.v4.n3.a8
30. Loh PR, Palamara PF, Price AL (2016) Fast and accurate long-range phasing in a UK Biobank cohort. Nat Genet 48:811–816
31. Palmer C, Pe'er I (2016) Bias characterization in probabilistic genotype data and improved signal detection with multiple imputation. PLoS Genet 12(6):e1006091. https://doi.org/10.1371/journal.pgen.1006091
32. Roshyara NR, Kirsten H, Horn K et al (2014) Impact of pre-imputation SNP-filtering on genotype imputation results. BMC Genet 15:88. https://doi.org/10.1186/s12863-014-0088-5
33. Ramnarine S, Zhang J, Chen LS et al (2015) When does choice of accuracy measure alter imputation accuracy assessments? PLoS One 10(10):e0137601
34. Abed A, Belzile F (2019) Comparing single-SNP, multi-SNP, and haplotype-based approaches in association studies for major traits in Barley. Plant Genome. https://doi.org/10.3835/plantgenome2019.05.0036
35. Torkamaneh D, Laroche J, Tardivel A et al (2018) Comprehensive description of genome-wide nucleotide and structural variation in short-season soya bean. Plant Biotechnol J 16 (3):749–759. https://doi.org/10.1111/pbi.12825

Chapter 14

Multiregion Sequence Analysis to Predict Intratumor Heterogeneity and Clonal Evolution

Soyeon Ahn and Haiyan Huang

Abstract

Multiregion sequencing can advance our understanding of the intratumor heterogeneity and the clonal evolution. Here, we introduced multiple aspects of multiregion sequencing and its analysis, including the study design and sampling strategy, current understanding of the tumor evolution model, and a protocol for multiregion sequencing analysis of DNA-sequencing data.

Key words Multiregional sequencing, Heterogeneity, Clonal evolution, Clone ordering, Phylogenetic tree

1 Introduction

Cancer evolves through the process of tumor formation, during which cells acquire somatic mutations over time through clonal evolution [1–3]. The clonal status of actional driver events and timing of the mutation process are the main selective pressures in the evolutionary trajectory. This dynamic evolutionary process is important for understanding the progression of cancer over time. Intratumor heterogeneity, or the presence of heterogeneous groups of cells (or *clones*) with different rare somatic mutations in a single tumor, also results from cancer evolution. Comprehensive analysis of tumors revealed that intratumor heterogeneity affects therapeutic resistance and treatment failure, as well as long-term clinical outcomes. Thus, understanding genetic heterogeneity in a single tumor is crucial in the prediction, prevention, and treatment of metastasis.

Longitudinal tumor samples are commonly required to investigate genetic heterogeneity and clonal evolution over time. Alternatively, multiregion or spatially separated samples obtained from a single patient can also be used to determine the evolutionary dynamics of cancer. When only multiregion samples are available,

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_14,

a fundamental step in studying the evolutionary trajectory of cancer is to construct a phylogenetic tree showing how somatic mutations and clonal expansion progress.

Recent developments in single-cell genome sequencing technology have greatly and directly influenced the understanding of tumor evolution. However, single-cell data has a high level of noise and the current methods remain prone to errors [4, 5]. In contrast, although multiregion bulk genome sequencing lacks the fine resolution at the single cell level, the bulk data is less noisy [5]. In addition, by sequencing multiple regions of a tumor over time, rich information on the complexity of the tumor tissue composition can be obtained, facilitating the study of intra-tumor heterogeneity and branched evolution [6]. In general, multiregion bulk sequencing has useful applicability in terms of its implication in clinics [7, 8]. There are also efforts that perform both multiregion bulk and single cell sequencing in order to improve the power and stability of results [9, 10].

In this article, we mainly focus on multiregion sequencing, and provide practical and technical details for constructing a tumor phylogenetic tree based on multiregion genome sequencing data.

2 Materials

2.1 Study Design

To design a multiregion sequencing study, sample availability and cost constraint are the first factors to consider. Other design factors include tissue preparation, tumor purity, sequencing platforms, sequencing depth, number of patients, number of samples per patient, as well as many clinical features [11]. We particularly discuss tissue preparation and sequencing strategies below, since they directly affect the data generation.

2.1.1 Tissue Preparation

Tumor and normal samples from the same patient are typically required to detect somatic mutations in cancer genome analysis. For normal samples, DNA extracted from white blood cells is usually used as control DNA to derive the normal genetic background. For tumor samples, DNA from fresh frozen tumor tissue is usually preferred. However, very often DNA extracted from formalin-fixed, paraffin-embedded (FFPE) samples could be the only available source from patients with solid tumors. In this situation careful consideration needs to be given to the DNA extraction methods as well as the selection of tissues or tumor regions to be sequenced, since FFPE generally contains poor quality DNA [12]. We refer to a previous study that offered a detailed description of protocols to collect tissues from multiple regions to investigate spatial heterogeneity [13].

2.1.2 Sequencing Strategy

Whole exome sequencing (WES) is generally more favorable compared to targeted sequencing to reveal unknown genetic variants [14]. However, in studies with preneoplasia or preinvasive lesions where tumor purity is expected to be low, sequencing depth will be a critical factor to affect the reliability of detected variants. A previous study demonstrated that WES with average depth of 300× failed to detect the variants that would be otherwise detectable by deep sequencing with average depth of 800× [15]. To mitigate the sequencing depth issue with WES, one might consider a two-stage design. At the first stage, WES can be used to scan a wide spectrum of mutations. At the second stage, targeted sequencing with high depth of coverage can be used to validate promising results from the first-stage WES analysis [16, 17]. In addition, copy number alterations (CNAs) are often recognized as a key feature associated with tumor evolution, because they may provide prognostic information [18–20]. Ignoring CNAs if they exist would affect the estimates of true cellular prevalence values (*see* **Note 1** for more details). When CNAs are expected, WES or SNP arrays would be cheaper alternatives but whole genome sequencing is the preferred platform for identification of CNAs due to its high resolution [21].

2.2 Data

To demonstrate the analysis pipeline, we will use a relatively small dataset of four samples. Detailed information about this illustrative dataset can be found in **Note 2**. In brief, each sample is a mixture of four known normal tissues with varying mixing proportions. Genotype information of low-frequency somatic point mutations (i.e., single nucleotide variants (SNVs)) was obtained by targeted sequencing. This dataset can be regarded as a mixture of four populations of diploid clones which do not have common mutations. Because samples are from normal tissues, copy number is two for every data point.

3 Methods

In this section, we review several key concepts and methodology components in constructing a tumor phylogenetic tree using low-frequency somatic mutation data (i.e., SNVs) sequenced from multiple tissues or tumor regions.

3.1 Overview on TUMOR Phylogenetic Tree Construction

A tumor phylogenetic tree is a canonical structure that describes tumor phylogeny. The leaves of the tree correspond to clones (mixture of cells) and the edges reflect sequences of somatic mutations [22, 23]. More specifically, the root hypothetically represents a normal cell (or a single founder cell) and nodes below are the root-cell's subclones. Edge length quantifies the difference of mutation profiles between a node and its immediate descendant

node. Therefore, a reconstructed tumor phylogenetic tree from cross-sectional data, together with the tumor's inferred clonal composition (i.e., the proportions of different clones/subclones), would offer a comprehensive illustration of the heterogeneity between tumor samples, and furthermore, help reveal a temporal clonal composition of tumors with samples collected longitudinally.

3.2 Evolution Model

Intratumor heterogeneity and subclonal alterations are important features of tumor evolution. As we have discussed above, most cancers evolve from a founder cell through clonal expansion leading to a population of heterogenous cancer cells. That is, clonal evolution in cancer is responsible for intratumor cellular diversity. It is widely accepted that intratumor heterogeneity is the result of evolutionary selection pressures and that subclones are selected because they exhibit the best fitness under the microenvironmental conditions [24]. Several models of evolution, including linear, branching, neutral, and punctuated evolution have been suggested. Linear evolution is the accumulation of sequential clonal succession [25]. In branching evolution, divergent subclones arise independently [26]. Neutral evolution is distinctively defined by the absence of selection, whereas the punctuated model is characterized by punctuated bursts of subclonal alterations during early cancer evolution [25]. It is believed that different tumor types follow different models of evolution. For the models mentioned above, since each relies on its own specific assumptions on timing of mutations and selection of clones, selecting an appropriate tumor evolution model is important for optimizing patient treatment [27–29]. For example, if a tumor evolution model concerns the evolutionary trajectory of treatment-resistant subclones, then adapting noninvasive screening like circulating tumor DNA would enable tracking the relevant mutations contained in the subclone [30]. Another example is the combined clonal therapies of ABL1 inhibitors, which is based on the branching process model to treat chronic myeloid leukemia patients [31]. The idea is to combine different targeted drugs with different profiles of resistance mutations at a varied timing might prevent the emergence of resistant subclones [30]. We will illustrate a computational construction of subclonal architectures, which would offer insights into a model of tumor evolution.

3.3 Sample Tree and Clone Tree

Accumulating evidence has suggested that tumor evolution affects the clinical course of the disease. For example, some studies based on multiregion sequencing have shown that genetic heterogeneity is correlated with a poor treatment response in cervical or ovarian cancer and that subclonal driver mutations are potential risk factors for rapid disease progression in chronic lymphocytic leukemia [32] and myelodysplasia [33]. Bulk tumor tissues contain a mixture of clones, and so, to construct a tumor phylogenetic tree, two analyses

need to be first done: a deconvolution process to identify the clonal composition of a tumor; and the determination of the clones' mutation profiles. We note that naïvely drawing a multiregion sample tree cannot correctly reflect the tumor's evolutionary trajectory, since that tree would mainly show the similarities between multiple regions [34]. In contrast, a clone tree, together with the inferred clonal composition of the tumor, offers a tumor phylogenetic representation and so it is suitable for characterizing tumor development. In general, a clone tree illustrating temporal ordering of mutations is considered a tumor phylogenetic tree.

3.4 Constructing a Tumor Evolutionary Tree

Through multiregion sequencing, SNVs and CNAs in multiregion samples can be obtained. SNVs and CNAs provide the basic information to construct a tumor evolutionary tree. SNVs can be summarized either as binary data (whether the variants are present or absent) or continuous data (VAF or copy-number-adjusted VAF when the assumption of diploidy is no longer valid). As VAF can be much affected by the copy number of the region where the point mutation locates, incorporating copy number information is useful. Note that reliable estimation of copy number changes remains difficult unless the whole genome sequencing or SNP array is conducted.

The heterogeneity of subclones can be quantified using SNVs and CNVs [3, 35]. For example, one may define the similarity between subclones by the number of common SNVs. When the depth of sequencing coverage is high, VAF values would be informative. The subpopulations of clones can be identified by a clustering analysis based on VAF values, and the temporal order of clones can be further inferred by phasing VAF similarity. There are two popular approaches to construct a tumor evolutionary tree from multiregion samples: clustering analysis based on the mutation events (*see* **Notes 3** and **4**) and the inference of the tree structure (*see* **Notes 6** and **7**). The first type of clustering approach aims to define subpopulations of clones based on mutation events such as SNVs. For example, PyClone implements a hierarchical model that infers cellular prevalence of variants and then produces clusters of clones/subclones [36] (*see* **Notes 3** and **4**). This clustering approach is intuitive, but it requires an additional step to identify subclones and decide clone ordering (phasing) to jointly infer the tree structures (*see* **Note 5**). The second type of approach is based on a simultaneous inference of phylogenetic structure as well as clonal composition. This type of approaches is more challenging since it involves extensive search spaces (*see* **Notes 6** and **7**). In addition, such approaches usually utilize mathematical models that require certain assumptions, such as (1) no mutation occurs twice in the course of cancer evolution and (2) no mutation is ever lost (no back mutations).

3.5 Software

In this article, we mainly illustrate the use of the following five popular phylogenetic tree construction software programs: PyClone (*see* **Note 3**), Clomial (*see* **Note 4**), LiCHeE (*see* **Note 5**), CITUP (*see* **Note 6**), and ClonEvol (*see* **Note 7**). A workflow for constructing clonal evolutionary tree from multiregion samples using these five software programs is shown in Fig. 1.

PyClone is based on a Bayesian clustering method that uses a Markov chain Monte Carlo framework to estimate the cellular prevalence distribution (*see* **Note 1**) [36]. Clomial decomposes the combination of total counts and variant allele counts matrix into a $Z \times P$ matrix, where Z is a clone genotype and P is the clone proportion determined by an expectation maximization method [37]. Unlike PyClone and Clomial, LICHeE and CITUP simultaneously perform matrix deconvolution and phylogenetic tree construction [38, 39]. LICHeE implements VAF value-based clustering and evolutionary constraint network construction, while CITUP enumerates all rooted trees and identifies an optimal genotype and phylogenetic relationship by exact quadratic integer programming or using an iterative heuristic method. In contrast, ClonEvol requires cluster information generated from other programs to perform clonal ordering and visualization [40].

In summary of input data, PyClone and Clomial require read counts of normal samples and tumor samples; LiCheE and CITUP require a matrix format for VAF values (*see* **Note 1**); Clonevol requires precalculated cluster information of variants and the CCF which we took from PyClone. The performance of the programs was evaluated using simulated datasets [39, 40]. These results, however, should be interpreted with caution because the true status of tumor growth is not directly observable and different types of cancers would require different tumor growth models. In Table 1, we provided the main features of the programs. We refer to [3] for comprehensive review on software tools for tumor phylogenetic tree construction.

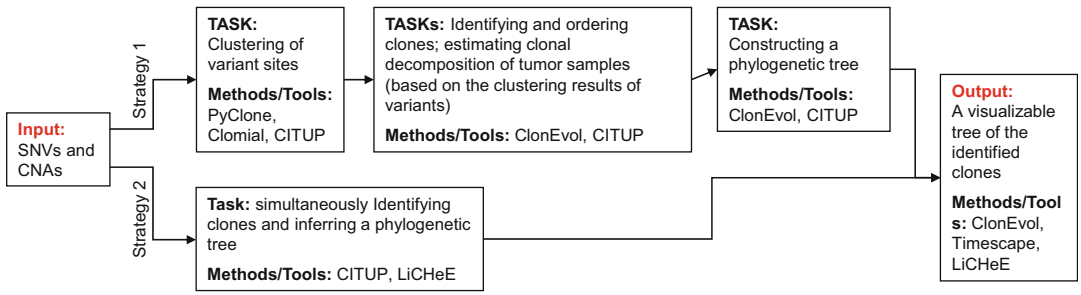


Fig. 1 Workflow for constructing a clonal evolutionary tree from multiregion samples

Table 1
Comparison of phylogenetic tree construction programs

	Inferring cluster	Inferring phylogeny	Phylogenetic characters	Adjusted VAFs	Source
PyClone	Y	N	S	Limited	https://github.com/aroth85/pyclone
Clomial	Y	N	S	N	https://www.bioconductor.org/packages/release/bioc/html/Clomial.html
LICHeE	optional	Y	S	Limited	https://github.com/viq854/lichee
CITUP	Y	Y	S	Limited	https://github.com/sfu-compbio/citup
ClonEvol				Limited	https://github.com/hdng/clonevol

C copy number alteration, *S* single-nucleotide variant, *CITUP* clonality inference in tumors using phylogeny, *LICHeE* lineage inference for cancer heterogeneity and evolution

4 Notes

Here we exemplify the construction of a tumor phylogenetic tree using multiregion sequencing data. First, we show examples of using clustering methods for grouping genetic variants into clones, which analysis is subsequently supplemented by other programs for subclone identification and clone ordering in order to achieve a clone tree construction. Second, we will illustrate how to construct a clone phylogenetic tree by inferring a clone tree structure simultaneously with identifying clonal composition.

Below, we provide a detailed introduction to data processing as well as step-by-step procedures for constructing and visualizing a tumor phylogenetic tree from read counts of variants collected from multiple samples. All the R and shell scripts used in the analysis can be found at https://github.com/stat-ahn/multiregion_sequencing_tutorial/. UNIX commands and code snippets are in `Courier New font`. UNIX commands are prefaced with a `$`, and R commends are prefaced with a `>`. The working directory for each program is assumed to be each subfolder.

1. Cancer cell fraction, variance allele frequency, and cellular prevalence: We provide illustrative examples of variance allele frequency (VAF), cancer cell fraction (CCF), and cellular prevalence adopted from [36, 41]. The VAF of somatic mutations is the relative frequency of a variant allele in a cell population. As a measure of diploid zygosity, the VAF in a heterozygous locus is close to 50%, whereas it is 100% for a homozygous locus, and close to 0% for a reference locus. The VAF is typically calculated as the number of reads supporting the variant allele over the total number of reads from the sequencing output, but is affected by the proportion of cancer

cells, proportion of tumor cells harboring the mutation, and copy number changes (*see* Supplementary Figure in [36]). To cope with multiple factors, it is common to incorporate CCF or cellular prevalence in the context of clonal tree construction. CCF is the fraction of cancer cells carrying the mutation in the variant population, and cellular prevalence is the fraction of cells carrying the mutations in the total cell population. CCF can be estimated from observed VAF by assuming that CCF follows a binomial distribution [32, 42].

$$\text{VAF} = \frac{\text{no.of mutant reads}}{\text{no.of mutant reads} + \text{no.of normal reads}}$$

$$\text{expected VAF} = \text{CCF} \times \text{mutation multiplicity} \times \frac{\rho}{\rho \times \text{copy.Mut} + (1-\rho) \times \text{copy.Norm}},$$

where ρ indicates the tumor purity and copy.Norm and copy.Mut indicate the copy numbers of normal cells and the tumor, respectively. Notably, for diploid genomes with 100% tumor purity and no copy number alterations, the CCF of a heterozygous mutation occurred in all tumor cells, where only one allele is affected by the same mutation, is double of its VAF, as every cell carrying the mutation contains one mutant and one wild-type allele.

Figure 2 illustrates hypothetical sequencing examples that the cell population has 80% of tumor cells and 20% of normal cells. Tumor cells consist of two subpopulations (clones): a star shape population with X and O mutations and an irregular star shape population with X mutation. If the percentage of tumor cells in a sequenced sample (i.e., a tumor purity) is assumed to be 100%, and a sample contains all diploid cells, then the CCFs of X and O mutation would be 100% and 62.5%, respectively, and the VAFs would be half of the CCFs, 50% and 31.25%. In Fig. 2a where a sample contains normal cells (which has reduced the observed tumor variant allele frequency), the VAFs of X and O mutations are now 44% and 28%, respectively. In Fig. 2b, tumor cells have copy number alterations which affect the VAFs of X and O mutations (67% and 21%, respectively). If we calculate the CCF given the observed VAF considering the true purity and copy numbers, then the most probable CCFs of X mutation and O mutation are 100% and 62.5%, respectively for both examples.

2. Identification of variants and preparation of input files for PyClone: In this protocol, we used the tutorial data from PyClone as our illustration datasets. Samples were generated to verify the low prevalence of mutations using ultra-deep targeted sequencing, and four Coriell DNA samples (NA12156, NA12878, NA18507, and NA19240) were mixed in different relative proportions (1%, 5%, 20%, and 74%), resulting in four different calibration samples and about

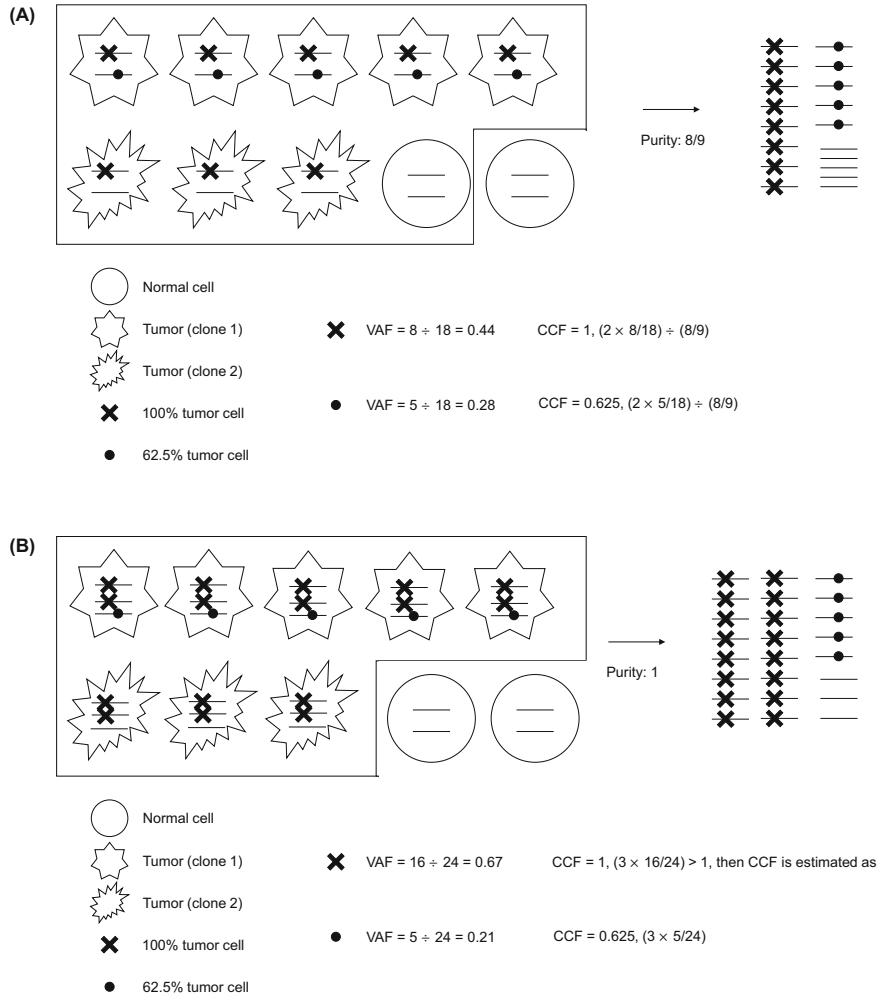


Fig. 2 (**a**) Calculating the cancer cell fraction (CCF) and variant allele frequency (VAF) from read counts in the diploid region when tumor purity is not 100%. (**b**) Calculating the CCF and VAF in the nondiploid region when tumor purity is 100%

100 mutations [43] (Fig. 3). In this example the nature of samples can be best characterized by the branching evolution model (*see* Subheading 3.4).

Here are the scripts to use the "head" command or "column" command to print the tab-delimited input files for PyClone.

```
$ head -n 5 examples/mixing/tsv/SRR385938.tsv
$ column -t examples/mixing/tsv/SRR385938.tsv | head -n 3
```

Six columns are required for PyClone: unique mutation identifier, number of reference allele read depth, number of variant alleles read depth, copy number of the mutant locus for the normal cells, and minor and major copy number of the tumor sample.

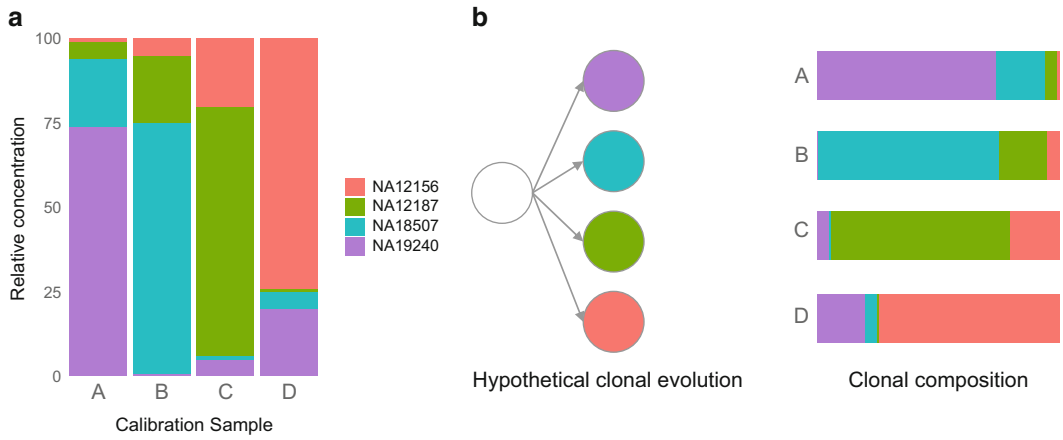


Fig. 3 (**a**) Relative proportion of the calibration samples (**b**) hypothetical clonal evolution model and clonal composition in each sample

3. Clustering variants using PyClone: PyClone identifies the clusters of variants by assuming that a sample is composed of a mixture of cellular populations and using observed VAF of variants as inputs. It further estimates cellular populations characterized by the allelic count data, copy number, and loss of heterozygosity information to convert VAF into cellular prevalence. Using a hierarchical Bayes model, beta-binomial density is used as the posterior density for the cellular prevalence [36]. The following command line seamlessly runs PyClone using a constraint of a maximum number of clusters of 4.

```
$PyClone run_analysis_pipeline --in_files tsv/SRR385938.tsv
tsv/SRR385939.tsv tsv/SRR385940.tsv tsv/SRR385941.tsv --work-
ing_dir output/pyclone --density pyclone_beta_binomial --
max_clusters 4
```

Alternatively, you can construct yaml file from each tsv file. A yaml format is intended to design for human readability and machine processability.

```
$PyClone build_mutations_file --in_file tsv/SRR385938.tsv --
out_file yaml/SRR385938.yaml
```

To finalize the analysis, config.yaml, which contains environment and computational settings, should be set up.

```
$PyClone run_analysis --config_file config.yaml
```

The configuration yaml file generated from the previous run_analysis_piepeline process can be reused.

```
$PyClone run_analysis --config_file outputall/config.yaml
```

The main outputs are two text files. "loci.tsv" is the core output indicating cluster estimates and cellular prevalence information for each locus. "cluster.tsv" contains the cluster overview. Importantly, the locus information file contains the estimated posterior cellular prevalence within the tumor. You can plot the cluster results from Pyclone as well (Fig. 4).

```
$PyClone plot_loci --config_file config.yaml --plot_file out-
put/plot --plot_type density
```

4. An alternative Clustering Tool: Clomial: Clomial intends to generate the cluster information of input variants as PyClone. Clomial can estimate clonal genotypes and frequencies from a given VAF matrix. Unlike PyClone, which requires a further distributional assumption a priori for clonal frequency, Clomial relies on matrix deconvolution. However, Clomial assumes that all loci are heterozygous for copy number neutral regions. Clomial is fast and works on R environment only, so it is a good alternative if there is less concern with the copy number alterations in the samples.

   An in-house R script, `code-clomial.R` generates input data when PyClone tsv input files are available. It extracts the normal allele counts and variant allele counts to obtain total counts.

```
> ClomialResult<-Clomial(Dc=Dc,Dt=Dt,maxIt=20,C=5,doParal=-
FALSE,binomTryNum=50,fliProb=0)
> chosen <- choose.best(models=ClomialResult$models)
```

5. Tree construction using ClonEvol from other clustering tools: PyClone or Clomial only produce cluster outputs and requires further steps to construct a tree. The clonal evolution trajectory can be inferred and visualized with ClonEvol. `code-clone-vol.R` is used to visualize the results given as the output file by PyClone.
6. Constructing tree using CITUP and visualizing tree using timescape: PyClone or Clomial only have cluster outputs and no tree structure information.

   CITUP enables the inference of clone phylogeny based on tree information using the frequencies of mutations in cells. The estimated copy number-corrected cellular frequencies from PyClone are used as input.

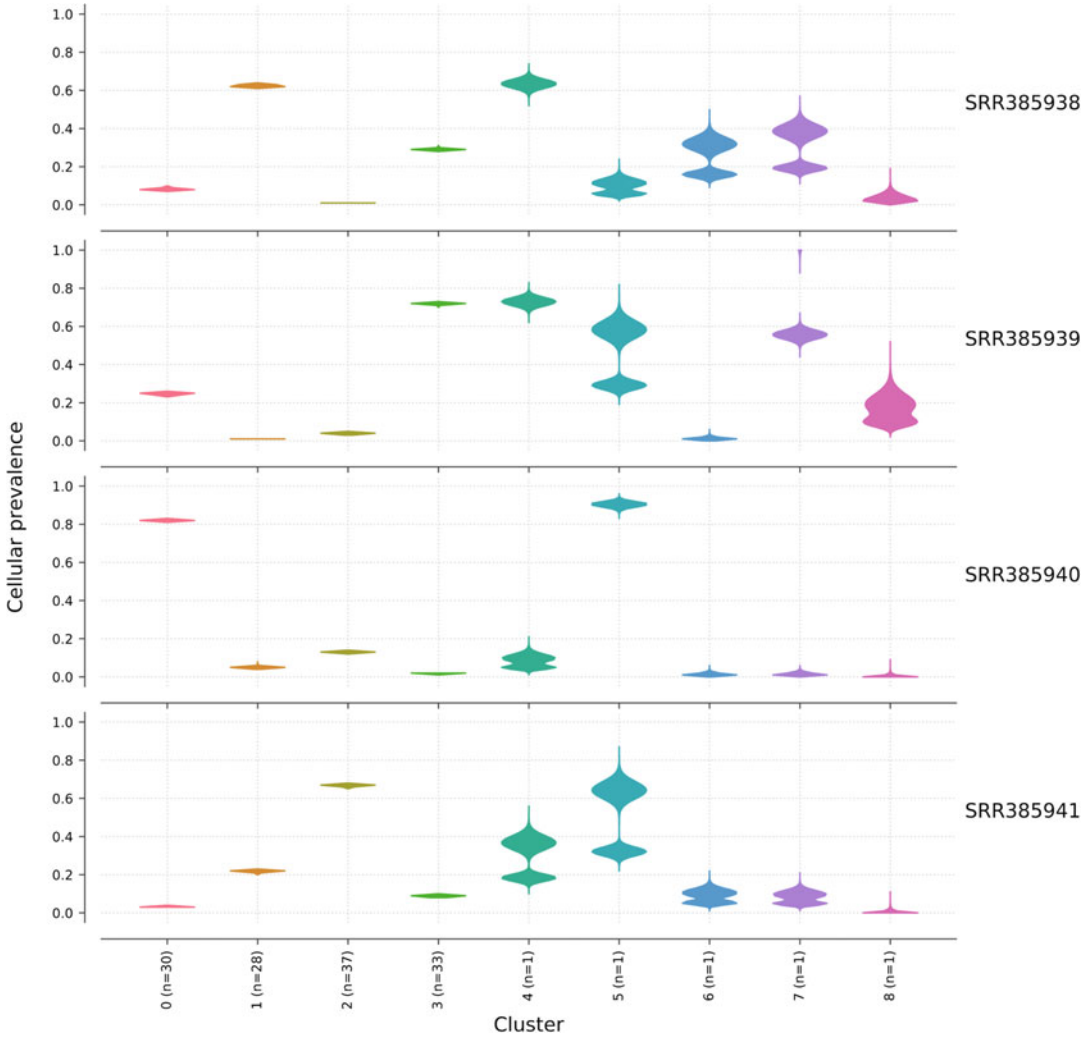


Fig. 4 PyClone clustering results

`code-citup-input.R` generates the input cellular frequency matrix for CITUP. It may be useful to remove some variants or clusters. It is advised to filter uncertain variants or to use a fixed number of variants. For example, variants with a broad posterior cellar prevalence distribution can be removed for a better estimate of subclusters in downstream analysis.

CITUP can be run in two different ways: iterative, and QIP version. Be aware that CITUP is computationally intensive and requires prefixed cluster information but runs much faster.

```
$run_citup_iter.py citup-input.tsv --submit local citup-SRR385.h5
$run_citup_qip.py citup-input.tsv citup-cluster.tsv citup-SRR385-qip.h5 --submit local
```

The default output of CITUP is a Python pandas binary HDF5 format. If the output is read using `code-citup-output.R`, the optimal tree solution can be determined, in which, each row of the data represents the adjacency list, edge pairs of the ancestor node and the corresponding descendant node. The estimated optimal tree from CITUP can be drawn with timescape in R with a provided R script `code-timescape.R`.

7. Constructing Tree Using LICHeE: LICHeE draws an estimated tree within a reasonable computing time. We generated the input file from the PyClone output. This step requires the VAF (or CCF) of a normal cell, which we assumed to be zero in our example. The following R script `code-lichee-input.R` also generates an input file.

```
./lichee -build -i ./SRR385-lichee.tsv  -maxVAFAbsent 0.005
-minVAFPresent 0.005 -n 0 -minPrivateClusterSize 2 -showTree
1 -s 1 -cp -o ./SRR385-lichee-output.tsv
```

Unlike other programs, LICHeE can efficiently identify a reconstructed tree by clustering variants and searching evolutionary constraint network. It also produces the output figure of an estimated tree.

Acknowledgments

This work was partially supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (grant NRF-2015R1D1A1A02061597).

References

1. Nowell PC (1976) The clonal evolution of tumor cell populations. Science 194 (4260):23–28
2. Carter SL et al (2012) Absolute quantification of somatic DNA alterations in human cancer. Nat Biotechnol 30(5):413–421
3. Schwartz R, Schaffer AA (2017) The evolution of tumour phylogenetics: principles and practice. Nat Rev Genet 18(4):213–229
4. Jahn K et al (2016) Tree inference for single-cell data. Genome Biol 17:86
5. Tomlinson K, Oesper L (2019) Parameter, noise, and tree topology effects in tumor phylogeny inference. BMC Med Genomics 12 (Suppl 10):184
6. Gerlinger M et al (2012) Intratumor heterogeneity and branched evolution revealed by multiregion sequencing. N Engl J Med 366 (10):883–892
7. Gupta RG, Somer RA (2017) Intratumor heterogeneity: novel approaches for resolving genomic architecture and clonal evolution. Mol Cancer Res 15(9):1127–1137
8. Yan T et al (2019) Multi-region sequencing unveils novel actionable targets and spatial heterogeneity in esophageal squamous cell carcinoma. Nat Commun 10(1):1670
9. Ramazzotti D et al (2019) Learning mutational graphs of individual tumour evolution from single-cell and multi-region sequencing data. BMC Bioinformatics 20(1):210
10. Malikic S et al (2019) Integrative inference of subclonal tumour evolution from single-cell and bulk sequencing data. Nat Commun 10 (1):2750
11. Shi W et al (2018) Reliability of whole-exome sequencing for assessing intratumor genetic heterogeneity. Cell Rep 25(6):1446–1457

12. McDonough SJ et al (2019) Use of FFPE-derived DNA in next generation sequencing: DNA extraction methods. PLoS One 14(4): e0211400
13. Jamal-Hanjani M et al (2017) Tracking the evolution of non-small-cell lung cancer. N Engl J Med 376(22):2109–2121
14. Van Allen EM et al (2014) Whole-exome sequencing and clinical interpretation of formalin-fixed, paraffin-embedded tumor samples to guide precision cancer medicine. Nat Med 20(6):682–688
15. Zhang J et al (2014) Intratumor heterogeneity in localized lung adenocarcinomas delineated by multiregion sequencing. Science 346 (6206):256–259
16. Weaver JMJ et al (2014) Ordering of mutations in preinvasive disease stages of esophageal carcinogenesis. Nat Genet 46(8):837–843
17. Damm F et al (2014) Acquired initiating mutations in early hematopoietic cells of CLL patients. Cancer Discov 4(9):1088–1101
18. Yu Y et al (2019) Genome-wide copy number variation analysis identified ANO1 as a novel oncogene and prognostic biomarker in esophageal squamous cell cancer. Carcinogenesis 40 (10):1198–1208
19. Rennstam K et al (2003) Patterns of chromosomal imbalances defines subgroups of breast cancer with distinct clinical features and prognosis. A study of 305 tumors by comparative genomic hybridization. Cancer Res 63 (24):8861–8868
20. Han X et al (2019) Comprehensive profiling of gene copy number alterations predicts patient prognosis in resected stages I-III lung adenocarcinoma. Front Oncol 9:556
21. Luo F (2019) A systematic evaluation of copy number alterations detection methods on real SNP array and deep sequencing data. BMC Bioinformatics 20(Suppl 25):692
22. Hu Z, Curtis C (2016) Inferring tumor phylogenies from multi-region sequencing. Cell Syst 3(1):12–14
23. Beerenwinkel N et al (2015) Cancer evolution: mathematical models and computational inference. Syst Biol 64(1):e1–e25
24. Niida A et al (2018) Understanding intratumor heterogeneity by combining genome analysis and mathematical modeling. Cancer Sci 109 (4):884–892
25. Davis A et al (2017) Tumor evolution: linear, branching, neutral or punctuated? Biochim Biophys Acta Rev Cancer 1867(2):151–161
26. Yates LR, Campbell PJ (2012) Evolution of the cancer genome. Nat Rev Genet 13 (11):795–806
27. Sottoriva A et al (2015) A Big Bang model of human colorectal tumor growth. Nat Genet 47 (3):209–216
28. Ling S et al (2015) Extremely high genetic diversity in a single tumor points to prevalence of non-Darwinian cell evolution. Proc Natl Acad Sci U S A 112(47):E6496–E6505
29. Williams MJ et al (2016) Identification of neutral tumor evolution across cancer types. Nat Genet 48(3):238–244
30. Fittall MW, Van Loo P (2019) Translating insights into tumor evolution to clinical practice: promises and challenges. Genome Med 11 (1):20
31. Lindstrom HJG et al (2019) Stochastic modelling of tyrosine kinase inhibitor rotation therapy in chronic myeloid leukaemia. BMC Cancer 19(1):508
32. Landau DA et al (2013) Evolution and impact of subclonal mutations in chronic lymphocytic leukemia. Cell 152(4):714–726
33. Papaemmanuil E et al (2013) Clinical and biological implications of driver mutations in myelodysplastic syndromes. Blood 122 (22):3616–3627; quiz 3699
34. Alves JM et al (2017) Multiregional tumor trees are not phylogenies. Trends Cancer 3 (8):546–550
35. DiNardo Z et al (2019) Distance measures for tumor evolutionary trees. Bioinformatics
36. Roth A et al (2014) PyClone: statistical inference of clonal population structure in cancer. Nat Methods 11(4):396–398
37. Zare H et al (2014) Inferring clonal composition from multiple sections of a breast cancer. PLoS Comput Biol 10(7):e1003703
38. Popic V et al (2015) Fast and scalable inference of multi-sample cancer lineages. Genome Biol 16:91
39. Malikic S et al (2015) Clonality inference in multiple tumor samples using phylogeny. Bioinformatics 31(9):1349–1356
40. Dang HX et al (2017) ClonEvol: clonal ordering and visualization in cancer sequencing. Ann Oncol 28(12):3076–3082
41. Dentro SC et al (2017) Principles of reconstructing the subclonal architecture of cancers. Cold Spring Harb Perspect Med 7(8)
42. McGranahan N et al (2015) Clonal status of actionable driver events and the timing of mutational processes in cancer evolution. Sci Transl Med 7(283):283ra254
43. Harismendy O et al (2011) Detection of low prevalence somatic mutations in solid tumors with ultra-deep targeted sequencing. Genome Biol 12(12):R124

Chapter 15

Overcoming Interpretability in Deep Learning Cancer Classification

Yue Yang (Alan) Teo, Artem Danilevsky, and Noam Shomron

Abstract

Since its inception, deep learning has revolutionized the field of machine learning and data-driven science. One such data-driven science to be transformed by deep learning is genomics. In the past decade, numerous genomics studies have adopted deep learning and its applications range from predicting regulatory elements to cancer classification. Despite its dominating efficacy in these applications, deep learning is not without drawbacks. A prominent shortcoming of deep learning is the lack of interpretability. Hence, the main objective of this study is to address this obstacle in the deep learning cancer classification. Here we adopt a feature importance scoring methodology (Gradient-based class activation mapping or Grad-CAM) on a quasi-recurrent neural network model that classify cancer based on FASTA sequencing data. In this study, we managed to formulate a nucleotide-to-genomic-region Grad-CAM scoring methodology, as well as, validate the use this methodology for the chosen model. Consequently, this allows for the utilization of the Grad-CAM scoring methodology for feature importance in deep learning cancer classification. The results from our study identify potential novel candidate genes, genomic elements, and mechanisms for future cancer research.

Key words Deep learning, Cancer classification

1 Introduction

Deep learning is a variation of machine learning that utilizes deep neural networks with hidden layers to extract abstract features of data [1]. Since its inception, deep learning has engendered unprecedented progress in many data-driven fields such as computer vision [2, 3] and speech recognition [4–6]. One of the more recent fields that has incorporated deep learning is genomics [7].

Genomics data are innately complex and voluminous in nature, which makes it extremely difficult to elucidate the relationships and functionalities of genetic elements. Deep neural networks can be exceptionally capable in computing abstract features from genomic

Yue Yang (Alan) Teo and Artem Danilevsky are equal contributors

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_15,

data and constructing meaningful predictions from these features. The possibility of enhanced performance of deep learning has made it a popular research direction amongst bioinformaticians and its applications include predicting molecular phenotypes [8–12], classification of cancer subtypes [13] and predicting transcription factor binding sites [14].

However, utilizing deep learning in genomics is not without its caveats. A prominent drawback of deep learning is the interpretability of the deep learning models [7]. The abstract features derived by these deep neural networks are isolated from the biological implications. This makes it hard for bioinformaticians to draw conclusive inferences about the relationship between the genomics data and the prediction. To resolve the shortage of interpretability in deep learning models, deep learning scientists have developed several methods to interrogate the deep learning models. For example, the use of explainability methods such as class activation mapping (CAM) [15] in deep learning models of computer vision (as illustrated in Fig. 1). From the Fig. 1, it shows that the CAM allows one to visualize and identify the important features used by the deep learning model to classify the images.

One such method is to assign feature importance scores to the input features. These scores reveal which components of the input contributes the most to the model's prediction [7]. From these scores, one can then posit biological explanations for the predictions made. Across deep learning in genomics, feature importance scores have been speculated to be useful in identifying DNA motifs [8, 14, 16], single nucleotide mutations [9, 16], and epistatic interactions [17].

An effective way of deriving feature importance scores for deep learning model is the CAM methodology mentioned above. The CAM methodology is designed for a class of deep neural network,

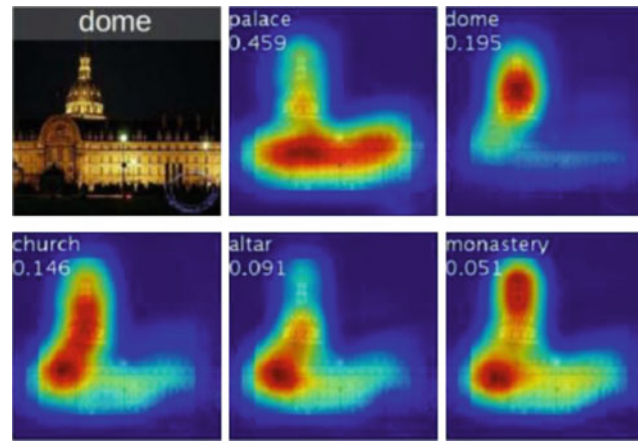


Fig. 1 Example of CAM as extracted from reference [15]

termed convolutional neural network (CNN). In essence, it identifies discriminative regions in images used for image classification by mapping the class scores back to the previous convolution layer [15]. A more accurate version of CAM, gradient-based CAM (Grad-CAM), was then proposed in 2016 [18].

Instead of mapping only to the previous convolution layer, Grad-CAM traverses backward across all the convolution layers, using the gradients and the weights of the network to formulate the feature importance scores [18]. Although Grad-CAM was initially used for image classification, the underlying theory can be generalized to deep learning models that utilizes CNNs. In fact, a 2018 study adopted Grad-CAM on a CNN model that classifies tumor cells according to genomic data. In the study, they managed to validate that the top scoring genes derived from Grad-CAM are related to tumor-specific pathways [19]. This establishes the veracity of Grad-CAM in assigning feature importance scores for genomics data in cancer classification tasks.

The mentioned paper used normalized read counts for each gene as their input for their CNN model. In contrast to using normalized read counts, this paper proposes the use of raw sequencing data as input for deep learning in cancer classification. Using raw sequencing data confers two main advantages:

1. The quantification of read counts results in uncertainty [20], which may propagate into the predictions and Grad-CAM scores. Using raw sequencing data removes this uncertainty, improving the veracity of the model's prediction and consequently the Grad-CAM scores.
2. Using quantification of gene read counts limits the Grad-CAM scores to known gene regions. Conversely, with raw sequencing data, one can map Grad-CAM scores onto genetic elements that are not necessarily coding genes or annotated genes but are still influential in the cancer classification. Alternatively, the Grad-CAM scores can also identify potential novel unknown cancer driver genes or genomic regions and mechanisms related to cancer.

Therefore, this paper adopts a deep learning classification of raw sequencing data into either cancer or healthy sample and computes Grad-CAM scores from the deep neural networks.

2 Materials

All code is performed in python 3.6 and the libraries required are seaborn [21], scipy [22], numpy [23], and cv2 [24]. The code is a specific extension of the model presented in Chap. 9.

Raw sequencing data is obtained from the [25].

The model used for the deep learning is a quasi-recurrent neural network (QRNN) model adopted from Chap. 9 and the details of the model as per implemented in the paper.

3 Methods

Grad-CAM is implemented as such:

1. The gradients and activation weights of the convolution layers are extracted for every correctly classified sample and separated into their respective class (cancer/healthy).
2. The gradients and activation weights are then used to calculate the Grad-CAM scores as per [18].
3. Each sequencing read is assigned a Grad-CAM score

To extrapolate the Grad-CAM scores from the read to the genome, the following steps are performed:

1. If the read maps onto a nucleotide position on a chromosome, the Grad-CAM score of the read is assigned to the nucleotide position on the chromosome, as well as the subsequent 99 nucleotides. These positions are also assigned a "hit," indicating that it has been assigned a score.
2. If a nucleotide has more than one hit, the mean of the scores is used.
3. A normalization constant is also introduced to the Grad-CAM score. The constant is to normalize high Grad-CAM scores with very little hits. The normalization constant, N_i, for score s_i is calculated with the following formula:

$$N_{\mathrm{i}} = \begin{cases} 1, x_{\mathrm{i}} \geq \mu_{\mathrm{x}} \\ 1 - e^{(x_{\mathrm{i}}/\mu_{\mathrm{x}})}, \text{otherwise.} \end{cases}$$

where x_{i} is the hits for score s_{i} and μ_{x} is the mean hit across all scores for the class; either cancer or healthy sample.

After the Grad-CAM scores are formulated at a nucleotide level, the scores are then mapped to annotated genomic regions in the human genome GRCh37.87 as this was the genome reference that the authors of the data used. The score of each genomic region is computed by taking the mean score of the scoring nucleotides within the region and multiplied it by a scaling constant, 10^4.

To validate that the Grad-CAM scores obtained are meaningful in a biological context for the cancer classification, we referred back to the source of the data, [25]. The source paper highlighted a list of 57 cancer driver genes that also were partially enriched using TEC-sequencing (which we denote as source-cancer genes or SC genes) in their Supplementary Table S1, that they investigated. For

the correctly classified samples, we obtained the Grad-CAM scores of these SC genes and compared them to the scores of the other protein-coding genes that are not identified as the cancer genes in the source paper (which we denote as non–source-cancer genes or non-SC genes). The intuition is as such: if the Grad-CAM scores of the SC genes are higher than that of non-SC genes, then it proves two points: (a) the Grad-CAM scores are useful in identifying the key features for the classification; (b) the SC genes are essential for the classifier in cancer classification.

Here we must discuss potential caveats in our method. Although our approach to compare scores of SC-genes to non-SC genes seems reassuring, half of the data comes from these SC-genes while the other half comes from the rest of the genome (which includes non-SC genes and noncoding DNA). This means that in the training dataset about 50% are reads from SC-genes compared to a very small fraction (way less than 50%) that comes from non-SC genes. This imbalance may lead to a significant bias in the findings. Yet, at the same time, we note that this potential bias is equally present both in the healthy and cancer samples, which might in fact balance each other out. Or in other words, if SC-genes received higher scores then they might genuinely be more significant. In order to address the bias, one can possibly normalize the SC-genes scores by a penalty which adds a factor by which they are overrepresented compared to non-SC genes.

The scores are compared using nonparametric Mann–Whitney U (MWU) test [26]. In this context, the MWU test compares the number of times a score from the SC gene is ranked higher than the score from the non-SC gene. Hence, the MWU test will be able to show if the Grad-CAM scores of the SC genes are significantly higher than that of the non-SC genes.

4 Results

The classifier is trained as described in Chap. 9 (Artem Danilevsky and Noam Shomron) with an accuracy of 83% in classification of healthy and breast cancer patients. Approximately 50% of the reads mapped to unannotated regions of the human genome for both classes.

For the cancer class, the SC genes have a mean Grad-CAM score of 493.53 and the non-SC genes have a mean Grad-CAM score of 106.29. The probability distribution of the scores are illustrated below (*see* Fig. 2 (top)). The MWU test shows that the Grad-CAM scores of the SC genes are significantly higher than the Grad-CAM scores of the non-SC genes (p-value: $1.04e^{-37}$). Since the SC genes have higher Grad-CAM scores, it proves that the SC genes are influential in the classification of cancer samples. In other

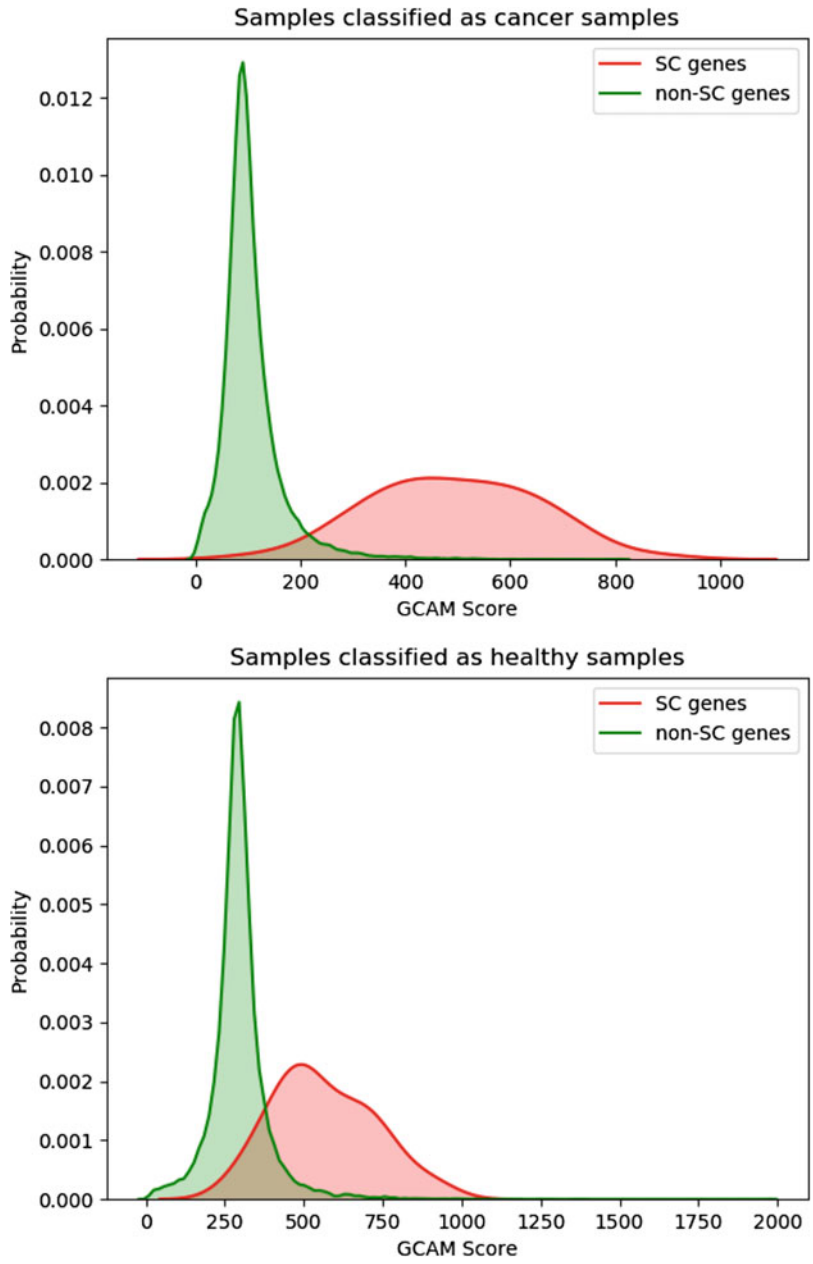


Fig. 2 Top, Probability distribution of Grad-CAM scores of genes samples classified as cancer samples. Bottom, Probability distribution of Grad-CAM scores of genes in samples classified as healthy samples

words, the QRNN model utilizes the raw sequencing data from these genes for cancer classification.

In the healthy class, the SC genes have a mean Grad-CAM score of 565.24 and that of non-SC genes is 294.85. The probability distribution of the scores for the healthy class are illustrated in Fig. 2 (bottom). Similarly, the MWU test shows the Grad-CAM

scores of the SC genes are significantly higher than the Grad-CAM scores of the non-SC genes (p-value: $1.02e^{-31}$). Using the same logic as before, this result signifies that the model uses the raw sequencing data from these genes for the healthy class classification.

These combined results reinforce the fact that the model uses the raw sequencing data originating in the SC genes for the classification of both cancer and healthy classes. Consequently, this proves the usefulness of the Grad-CAM scores in identifying crucial features used in the deep learning classification.

Subsequently, we investigated the top 20 highest scoring protein-coding genes in both classes, as shown in Table 1.

In the cancer class, most of the top 20 genes are associated with tumour suppression or promotion of tumour growth and 10 of these genes are SC genes. We then performed Gene Ontology (GO) analysis [27] on these genes with false discovery rate (FDR) < 0.05. The top few processes from the GO analysis include anoikis, regulation of DNA methylation, and response to UV. Unsurprisingly, these processes are associated with cancer. However, a more exciting avenue to investigate is the genes and processes that are not associated with cancer. For example, SLITRK5 is a gene that is not commonly associated with cancer but has been proposed to play a role in cancers such as leukemia [28] and has also been selected as a potential therapeutic target for cancer [29]. The relatively high Grad-CAM score of SLITRK5 from this paper further solidifies SLITRK5 as a candidate gene to study in cancer. Hence, this shows that the Grad-CAM scores can help to filter and provide novel potential candidate genes that may be involved in cancer. Consequently, this may engender the discoveries of new mechanisms in cancer as well as faciliate the development of more efficient therapeutic treatments.

For the healthy class, most of the top scoring genes are not involved in cancer or tumor growth and suppression. This suggests that genes that are not usually associated with cancer are significant for the classification of the healthy class. Consequently, this observation provides the possibility that these genes may be implicated in the resistance to cancer. Although GO analysis reveals that there are no biological processes linking these genes, it does not necessarily indicate a dead end. Studying these genes in conjunction may reveal an entirely novel mechanism used for cancer resistance.

We note here that when classifying a sample into only two classes, the probabilities should be complementary, and choosing between one or the other, is arbitrary. In addition, we note that some genes affect the model's classification to one class while others to another class, and this is affected also by the fact that we chose only samples that were classified correctly.

A further extension of the study is to map the scores to other annotated regions such as micro-RNA (miRNA), small nuclear RNA (snRNA) and pseudogenes, as well as unannotated regions.

Table 1
The top 20 highest scoring protein-coding genes in cancer and healthy samples

Name	Score	Description
Cancer samples		
HRAS[a]TF1	885.0023	Harvey rat sarcoma viral oncogene homolog
DCAF12L2	810.2549	DDB1 and CUL4 associated factor 12-like 2
VHLL	807.4536	von Hippel-Lindau tumor suppressor-like
ERAS	784.3349	ES cell expressed Ras
STK11	773.1047	Serine/threonine kinase 11
TP53	731.3213	Tumor protein p53
PIK3CA[a]TF1	722.8825	Phosphatidylinositol-4,5-bisphosphate 3-kinase catalytic subunit alpha
NRAS[a]TF1	714.4802	Neuroblastoma RAS viral (v-ras) oncogene homolog
MYC[a]TF1	698.649	V-myc avian myelocytomatosis viral oncogene homolog
GNAT1	687.3846	Guanine nucleotide binding protein (G protein) alpha transducing activity polypeptide 1
CDK4[a]TF1	681.8115	Cyclin-dependent kinase 4
TSPAN31	674.2671	Tetraspanin 31
APC[a]TF1	663.4065	Adenomatous polyposis coli
KIT[a]TF1	660.5033	V-kit Hardy-Zuckerman 4 feline sarcoma viral oncogene homolog
SLITRK5	656.4401	SLIT and NTRK-like family member 5
FGFR3[a]TF1	655.9411	Fibroblast growth factor receptor 3
SMAD4[a]TF1	650.2999	SMAD family member 4
ZXDB	646.7136	Zinc finger X-linked duplicated B
UBXN2B	636.9674	UBX domain protein 2B
JAK3[a]TF1	634.9979	Janus kinase 3
Healthy samples		
FANCF	1973.904	Fanconi anemia complementation group F
PCDHB15	1380.306	Protocadherin beta 15
UBE2C	1352.957	Ubiquitin-conjugating enzyme E2C
KRTAP4-11	1259.963	Keratin associated protein 4-11
REXO1L11P	1244	REX1 RNA exonuclease 1 homolog (*S. cerevisiae*)-like 11 pseudogene
MLNR	1186.086	Motilin receptor
INO80C	1171.363	INO80 complex subunit C
OR4N5	1075.506	Olfactory receptor family 4 subfamily N member 5
ZNF778	1047.006	Zinc finger protein 778

(continued)

Table 1
(continued)

Name	Score	Description
ARL4D	1042.431	ADP-ribosylation factor-like 4D
GDF2	1033.654	Growth differentiation factor 2
GAGE12C	1022.064	G antigen 12C
VHLL	1004.316	von Hippel-Lindau tumor suppressor-like
AC012493.2	1002.768	Uncharacterized protein
GSX1	991.1442	GS homeobox 1
TEKT2	986.434	Tektin 2 (testicular)
OR10J5	986.0762	Olfactory receptor family 10 subfamily J member 5
POP5	978.7176	Processing of precursor 5 ribonuclease P/MRP subunit (S. cerevisiae)
VPS25	970.8976	Vacuolar protein sorting 25 homolog (S. cerevisiae)
AL590714.1	964.8882	Uncharacterized protein

[a]Indicates that they are SC genes

By doing so, we may uncover hidden roles of these genomic regions in cancer. Examining noncoding regions may provide epigenomic mechanisms to the development of cancer [30]. On the other hand, one may also discover novel functional genomic regions from the unannotated regions.

For a starting point, we investigated the scores of the general genomic regions in the cancer class. These regions include but not limited to intronic regions, snRNA, miRNA, ribosomal RNA (rRNA), pseudogene, long intergenic noncoding RNA (lincRNA) and unannotated regions. From the probability distribution of scores across these different types of genomic regions in Fig. 3, we can observe that these genomic regions also have relatively high scores and may play essential roles for the classification and consequently, cancer development. In fact, 19 out of the top 20 scoring genomic regions are non–protein-coding genomic regions (see Table 2). These results show that the non–protein-coding regions may play a more influential role in cancer.

5 Conclusion

This chapter established the usefulness of Grad-CAM in assigning feature importance to genomic regions for the QRNN model adopted from studies in our laboratory oratory with raw sequencing data. The Grad-CAM scoring methodology used in this chapter

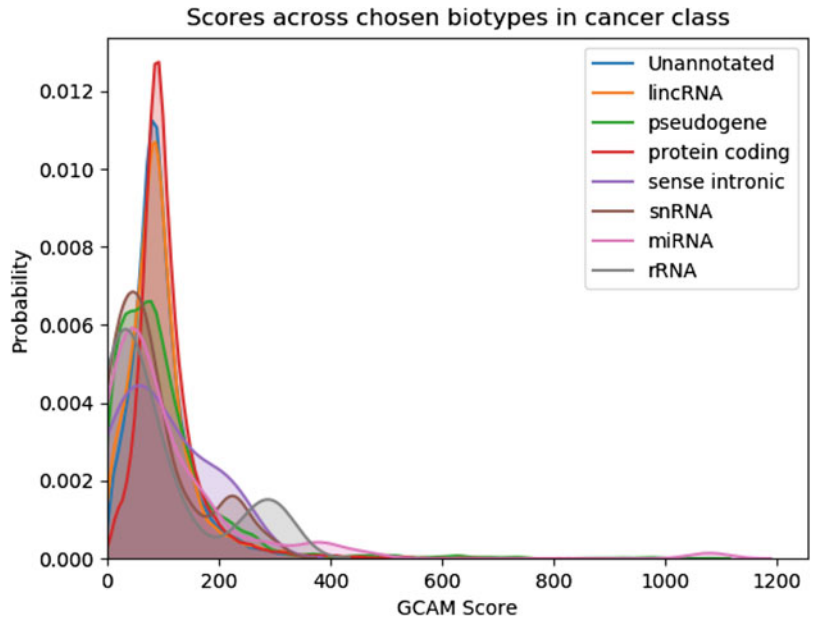


Fig. 3 Probability distribution of Grad-CAM scores of chosen genomic regions (biotypes) in samples classified as cancer samples

Table 2
Top 20 highest scoring general genomic regions in cancer class

Cancer class		
Name	**Score**	**Description**
Others (chr17: from 60885409 to 60885857)	1134.08	Logic name = CpG
AL161626.1	1078.91	miRNA
RP11-681H10.1	1075.359	Pseudogene
Unannotated (chrX: from 122898986 to 122899151)	1071.068	–
Others (chr22: from 22735604 to 22735867)	1066.86	Logic name = FirstEF
Others (chr11: from 72900511 to 72900573)	1025.409	Logic name = FirstEF
Unannotated (chr6: from 24126348 to 24126350)	1007.018	–
Others (chr7: from 64042946 to 64043007)	1001.014	Logic name = FirstEF
Others (chr17: from 45949768 to 45949897)	996.9688	Logic name = FirstEF

(continued)

Table 2
(continued)

Cancer class		
Name	**Score**	**Description**
Others (chr20: from 35899312 to 35899478)	996.6895	Logic name = FirstEF
RP11-188I24.1	991.8658	Pseudogene
Others (chr6: from 24126345 to 24126348)	983.3727	Logic name = Eponine
Others (chr5: from 1289276 to 1295970)	978.4887	Logic name = CpG
RP11-414B7.1	973.7564	Pseudogene
Others (chr17: from 60885799 to 60885860)	971.3349	Logic name = FirstEF
CTD-2272G21.2	970.3346	Pseudogene
KRASP1	966.6589	Kirsten rat sarcoma viral oncogene homolog pseudogene 1
Others (chr19: from 1283845 to 1283994)	961.8313	Logic name = FirstEF
Others (chr17: from 60142248 to 60143516)	961.8226	Logic name = CpG
Others (chr7: from 25989884 to 25990566)	960.992	Logic name = FirstEF

can also be easily adopted to another CNN-based model and input data type. Other than validating the veracity of the deep learning model, this chapter also proved that Grad-CAM scoring at a nucleotide level can be used to identify potential novel genomic regions and biological processes related to cancer.

However, it is prudent to take note that the deep learning parameters may also be arbitrary and the scores may have no biological importance. Hence, it is still imperative for experimental validation of the involvement of these genomic regions in cancer.

Once substantiated with experimental results, these genomic regions can aid in our quest to elucidate the mechanisms of cancer and hopefully design more effective diagnosis and therapeutic treatments. In conclusion, our study demonstrated the effectiveness of Grad-CAM scoring for feature importance in deep-learning classification of cancer using raw sequencing data.

References

1. Goodfellow I, Bengio Y, Courville A (2016) Deep learning. The MIT Press
2. Krizhevsky A, Sutskever I, Hinton GE (2012) ImageNet classification with deep convolutional neural networks. In: Pereira F, Burges CJC, Bottou L, Weinberger KQ (eds) Advances in neural information processing systems 25. Curran Associates, Inc., pp 1097–1105
3. Ciregan D, Meier U, Schmidhuber J (2012) Multi-column deep neural networks for image classification. In: 2012 IEEE conference on computer vision and pattern recognition, Providence, RI, pp 3642–3649. https://doi.org/10.1109/CVPR.2012.6248110
4. Hinton G, Deng L, Yu D et al (2012) Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Signal Process Mag 29 (6):82–97. https://doi.org/10.1109/MSP.2012.2205597
5. Morgan N, Bourlard H, Renals S et al (1993) Hybrid neural network/hidden markov model systems for continuous speech recognition. Intern J Pattern Recognit Artif Intell 07 (04):899–916. https://doi.org/10.1142/S0218001493000455
6. Lee C-H (2009) Developments and directions in speech recognition and understanding, part 1. IEEE Signal Process Mag 26(3):75–80
7. Eraslan G, Avsec Ž, Gagneur J et al (2019) Deep learning: new computational modelling techniques for genomics. Nat Rev Genet 20 (7):389–403. https://doi.org/10.1038/s41576-019-0122-6
8. Kelley DR, Snoek J, Rinn JL (2016) Basset: learning the regulatory code of the accessible genome with deep convolutional neural networks. Genome Res 26(7):990–999. https://doi.org/10.1101/gr.200535.115
9. Zhou J, Troyanskaya OG (2015) Predicting effects of noncoding variants with deep learning-based sequence model. Nat Methods 12(10):931–934. https://doi.org/10.1038/nmeth.3547
10. Kelley DR, Reshef YA, Bileschi M et al (2018) Sequential regulatory activity prediction across chromosomes with convolutional neural networks. Genome Res 28(5):739–750. https://doi.org/10.1101/gr.227819.117
11. Angermueller C, Lee HJ, Reik W et al (2017) DeepCpG: accurate prediction of single-cell DNA methylation states using deep learning. Genome Biol 18(1):67. https://doi.org/10.1186/s13059-017-1189-z
12. Zeng H, Gifford DK (2017) Predicting the impact of non-coding variants on DNA methylation. Nucleic Acids Res 45(11):e99. https://doi.org/10.1093/nar/gkx177
13. Rhee S, Seo S, Kim S (2018) Hybrid approach of relation network and localized graph convolutional filtering for breast cancer subtype classification. In: Proceedings of the twenty-seventh international joint conference on artificial intelligence, pp 3527–3534
14. Wang M, Tai C, Weinan E et al (2018) DeFine: deep convolutional neural networks accurately quantify intensities of transcription factor-DNA binding and facilitate evaluation of functional non-coding variants. Nucleic Acids Res 46(11):e69. https://doi.org/10.1093/nar/gky215
15. Zhou B, Khosla A, Lapedriza A, et al (2015) Learning deep features for discriminative localization, arXiv:1512.04150 [cs]
16. Alipanahi B, Delong A, Weirauch MT et al (2015) Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. Nat Biotechnol 33(8):831–838. https://doi.org/10.1038/nbt.3300
17. Greenside P, Shimko T, Fordyce P et al (2018) Discovering epistatic feature interactions from neural network models of regulatory DNA sequences. Bioinformatics 34(17):i629–i637. https://doi.org/10.1093/bioinformatics/bty575
18. Selvaraju RR, Cogswell M, Das A et al (2020) Grad-CAM: visual explanations from deep networks via gradient-based localization. Int J Comput Vis 128(2):336–359. https://doi.org/10.1007/s11263-019-01228-7
19. Lyu B, Haque A (2018) Deep learning based tumor type classification using gene expression data, bioRxiv, p 364323. https://doi.org/10.1101/364323
20. Conesa A, Madrigal P, Tarazona S et al (2016) A survey of best practices for RNA-seq data analysis. Genome Biol 17(1):1–19. https://doi.org/10.1186/s13059-016-0881-8
21. Hunter JD (2007) Matplotlib: a 2D graphics environment. Comput Sci Eng 9(3):90–95. https://doi.org/10.1109/MCSE.2007.55
22. Virtanen P, Gommers R, Oliphant TE, et al (2019) SciPy 1.0--Fundamental Algorithms for Scientific Computing in Python, arXiv:1907.10121 [physics]
23. van der Walt S, Colbert SC, Varoquaux G (2011) The NumPy array: a structure for efficient numerical computation. Comput Sci Eng

13(2):22–30. https://doi.org/10.1109/MCSE.2011.37

24. Bradski G (2000) The OpenCV library. Dr. Dobb's J Software Tools 120:122–125
25. Phallen J, Sausen M, Adleff V et al (2017) Direct detection of early-stage cancers using circulating tumor DNA. Sci Transl Med 9 (403):eaan2415. https://doi.org/10.1126/scitranslmed.aan2415
26. Leech NL, Barrett KC, Morgan GA et al (2014) IBM SPSS for intermediate statistics: use and interpretation, 5th edn. Routledge, New York
27. Mi H, Muruganujan A, Ebert D et al (Jan. 2019) PANTHER version 14: more genomes, a new PANTHER GO-slim and improvements in enrichment analysis tools. Nucleic Acids Res 47(D1):D419–D426. https://doi.org/10.1093/nar/gky1038
28. Proenca CC, Gao KP, Shmelkov SV et al (2011) Slitrks as emerging candidate genes involved in neuropsychiatric disorders. Trends Neurosci 34(3):143. https://doi.org/10.1016/j.tins.2011.01.001
29. Chano T, Kita H, Avnet S et al (2018) Prominent role of RAB39A-RXRB axis in cancer development and stemness. Oncotarget 9 (11):9852–9866. https://doi.org/10.18632/oncotarget.23955
30. Peschansky VJ, Wahlestedt C (Jan. 2014) Non-coding RNAs as direct and indirect modulators of epigenetic regulation. Epigenetics 9 (1):3–12. https://doi.org/10.4161/epi.27473

Chapter 16

Single-Cell Transcriptome Profiling

Guy Shapira and Noam Shomron

Abstract

Over the last decade, single cell RNA sequencing (scRNAseq) became an increasingly viable solution for analyzing cellular heterogeneity and cell-specific expression differences. While not as mature or fully realized as bulk sequencing, newly developed computational methods offer a solution to the challenges of scRNAseq data analysis, providing previously inaccessible biological insight at unprecedented levels of detail. Here, we go over the inherent challenges of single-cell data analysis and the computational methods used to overcome them. We cover current and future applications of scRNAseq in research of cellular dynamics and as an integrative component of biological research.

Key words Single-cell sequencing, Next-generation sequencing, Gene expression, Dimensionality reduction, R

1 Introduction

In standard, bulk-RNAseq, the sequenced RNA sample is extracted from a tissue, consisting of many thousands of individual cells. A major shortcoming of this approach is the loss of cell-type specific transcriptional differences. These cell-type specific differences are essential to many biological processes, but in a sample of many indistinguishable cell populations, the heterogeneity is either too subtle or completely saturated and undetectable [1].

Many advances were made in lab methodology and sequencing technology since the first published scRNAseq method over a decade ago [2], with over 130 different published methods at the time of writing [3]. The initial challenge of single-cell analysis is isolation of the single cell. This remains a heavily researched subject, with methods evolving from plate based, to microfluidic, nanopores and droplet-based [4]. After separation, RNA derived from each cell is barcoded with a unique sequence of nucleotides, a unique molecular identifier (UMI). The function of the UMI is twofold, association between cDNA fragment and molecule of origin and correction of PCR amplification bias [5]. Correction of

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_16,

amplification bias is especially crucial for scRNAseq, since each cell yields a subpicogram amount of RNA and therefore requires intensive amplification [6], this correction is essential for differential expression analysis [7].

Another aspect of single-cell analysis is the immense cellular heterogeneity, which is often at odds with concrete definitions and existing knowledge of cell variation. The study of diverse cell population commonly relied on cell-surface markers to classify cells into few general categories, failing to capture phenomena such as stochastic somatic alterations in clonal expansion [8] and phenotype switching in fluctuating environments [9]. Ido Amit's lab seminal work, MARS-Seq, allowed for an a priori single-cell profiling by using molecular, cellular, and plate-level barcodes, combined with unsupervised clustering [10]. The increased resolution allowed to explore immune cell heterogeneity in unprecedented resolution [11].

A sequencing technology is often made with a defined application and later extended and appropriated for many different use-cases [12]. The aforementioned MARS-Seq was recently used by Ido Amit's lab to characterize the cross talk of physically interacting cells [13]. scRNAseq technologies were used for detailed time-series analysis of the embryogenesis process [14]. Another exciting application of scRNAseq used it in combination with CRISPR-Cas9 induced genetic markers, following the lineage originating from the marked cell. By marking embryonic cells, lineage analysis was used to create a comprehensive atlas of all the cells of a planaria (a flatworm species) [15].

The recent explosion in scRNAseq data requires suitable computational methodologies. In the following section, we review the basic analysis workflow of Seurat, a prominent toolkit created to integrate the wealth of scRNAseq data [16].

2 Materials

2.1 Software and Hardware

We split the workflow presented here into to two parts: (a) raw data processing; and (b) count data analysis. We perform the memory and processor intensive first step on a Sun Grid Engine cluster, while the latter is performed on a fairly powerful workstation. Both parts are performed on a x86_64 Linux operating system, with most of the latter analysis carried out in R 3.6.3. Other packages are available either from the CRAN repository or from Bioconductor.

2.2 Data

We will use the publicly available GSE138852 dataset, consisting of eight single-nuclei libraries, each from entorhinal cortex samples of two individuals, healthy or one suffering from Alzheimer's disease. Additional data regarding sample preparation and methodologies outside our scope and can be found in Grubman et al. [17].

3 Methods

The single-cell RNAseq scheme and rationales are outlined below.

3.1 Raw Sequencing Data Processing

The raw sequencing data is deposited in the NCBI Short Read Archive (SRA) (https://www.ncbi.nlm.nih.gov/Traces/study/?acc=PRJNA577618). We first get a text file of the accession IDs of the raw data files we would like to download export a text file (SRR_Acc_List.txt). In order to download these files, we use the SRA toolkit.

```
# Download and extract the SRA toolkit
wget "http://ftp-trace.ncbi.nlm.nih.gov/sra/sdk/current/sratoolkit.current-ubuntu64.tar.gz"
tar -xzf sratoolkit.current-ubuntu64.tar.gz

# Configure SRA toolkit download path using the interactive menu
./vdb-config -i

# Download all required data from accession list
for l in `cat SRR_Acc_List.txt`; do ./prefetch -c $l; done

# Convert SRA files to fastq
for u in *.sra; do ./fastq-dump $u; done

# Compress fastq files
for f in *.fastq; do bgzip $f; done
```

To quantity gene expression from the fastq files(*see* **Note 1**), we use the software Cell Ranger, by 10X genomics with a prebuilt human genome assembly (requires EULA confirmation and some details) (https://support.10xgenomics.com/single-cell-gene-expression/software/downloads/latest) (*see* **Note 2**).

```
# Download and extract pre-built GRCh38 human genome reference assembly
curl -O http://cf.10xgenomics.com/supp/cell-exp/refdata-cellranger-GRCh38-3.0.0.tar.gz
tar -xvf refdata-cellranger-GRCh38-3.0.0.tar.gz

# Count gene expression for individual samples
cellranger count --id=sample_id --transcriptome=refdata-cellranger-GRCh38-3.0.0 \
--fastqs=R1.fastq.gz,R2.fastq.gz --sample=samplename
```

3.2 Count Data Quality Control

First, we load the data into a Seurat object and annotate it with condition and sample name (*see* **Note 3**).

```
# Install Seurat if not installed already
if (!requireNamespace("Seurat", quietly = TRUE))
    install.packages("Seurat")

library(Seurat)
library(tidyverse)

# Reading count data from Cell Ranger output (processed from raw seq
uencing data)
ds=CreateSeuratObject(Read10X(data.dir="/path/to/cellranger/output/"
), project="AD_brain")

# Alternatively, count data can be downloaded from the GEO database
using the GEOquery package
library(GEOquery)
geo=getGEOSuppFiles(GEO="GSE138852", fetch_files = TRUE)

# Use gzip -d GSE138852/GSE138852_counts.csv.gz to extract counts da
ta

# Create Seruat object from gene counts

ds=CreateSeuratObject(read.csv("GSE138852/GSE138852_counts.csv",
row.names=1), project="AD_brain")

# Add sample name and condition annotations using the column names
ds[["sample"]]=str_sub(colnames(ds), 18)
ds[["condition"]]=str_sub(colnames(ds), 18, -6)
```

Our counts matrix has 10,850 features(genes) represented by rows, from 12,906 barcodes(cells) represented by columns.

In addition to sequencing quality issues that need to be accounted for in bulk-sequencing technologies, single-cell sequencing introduces a new host of challenges related to the cell sorting process. It is likely that a few barcodes will be associated with two cells (commonly referred to as a doublet) or no cells at all, causing very high or very low overall gene count respectively. A large fraction of mitochondrial genes is an indication of a contamination, lower quality, or dying cells. We use visualizations to test for these quality issues (Figs. 1, 2, and 3).

```
ds[["percent.mt"]]=PercentageFeatureSet(ds, pattern="^MT-")
ds=subset(ds, subset= nFeature_RNA > 200 & nFeature_RNA < 1500 & per
cent.mt < 4)
ds=NormalizeData(ds)

VlnPlot(ds, features = c("nFeature_RNA", "nCount_RNA",
"percent.mt"), ncol = 3)
```

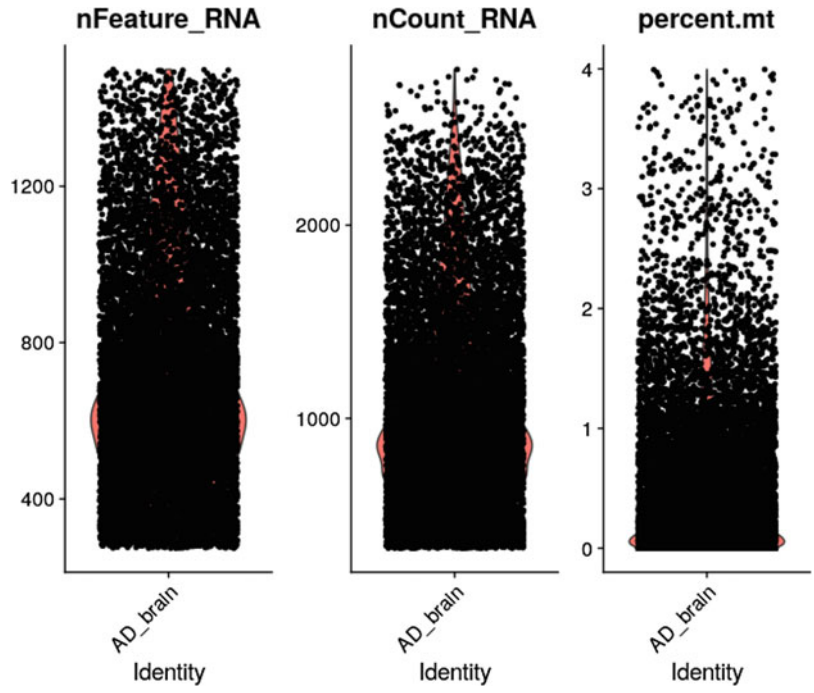


Fig. 1 Violin plot of the number of distinct features, overall counts, and percentage of mitochondrial count per cell

```
FeatureScatter(ds, feature1 = "nCount_RNA", feature2 = "nFeature_RNA")
```

Both in quality control and in the following clustering analysis, we will use visualizations to set arbitrary thresholds for the data. There is no real "gold standard" for filtering and clustering parameters, they are very dependent on the data composition and almost always boil down to a trade-off between data size and quality.

After filtering out unwanted cells, we normalize the count data using log-normalization. Each feature count is divided by the overall counts for each cell, multiplied by a factor, then transformed with natural log.

```
ds=NormalizeData(ds)
```

3.3 Dimensionality Deduction

Clustering is a common problem in the wider field of computational biology, it is essential for exploring the great diversity of cells, as well as the microbiome, tumors and many others. The goal is to cluster the cells in a biologically meaningful manner, with respect to cell type, tissue of origin, state, and more. Our most challenging hurdle is irrelevant variance, stemming from noise, technical issues, and the overall stochasticity of complex biological systems. In order to avoid distortions in our clustering, we first have to focus on the

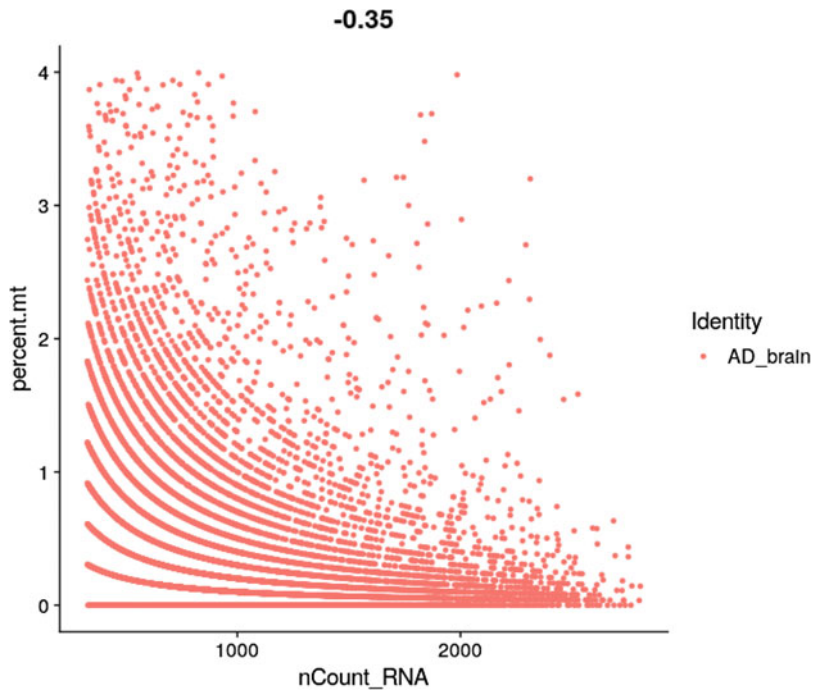


Fig. 2 Scatter plot of overall counts (*x*-axis) and percentage of mitochondrial count (*y*-axis) per cell

```
FeatureScatter(ds, feature1 = "nCount_RNA", feature2 = "percent.mt")
```

genes that change their expression in a significant manner, this process is called feature selection.

```
ds=FindVariableFeatures(ds, selection.method="vst", nfeatures=2200)
p=VariableFeaturePlot(ds)
LabelPoints(plot=p, points=head(VariableFeatures(ds), 25),
repel=TRUE)
```

Much like setting quality control thresholds, this is a somewhat arbitrary selection, aided by visualization (Fig. 4), that could be refined in later stages.

Features are akin to dimensions, continuous measures that we use to describe each entity (cell) in our data, in removing lowly expressed and invariable dimensions, we are left with a more informative description. Capturing the variance that separates one cell type from the other is not as trivial, it is spread over numerous dimensions. Experiments have dimensionalities too, for instance, samples from old and young patients, treated with a drug or not, can be represented by just two dimensions, one for each independent condition. When dealing with tens of thousands of diverse cells however, the dimensionality is unknown and has to be inferred from a high dimensional data, strongly unintuitive to us humans.

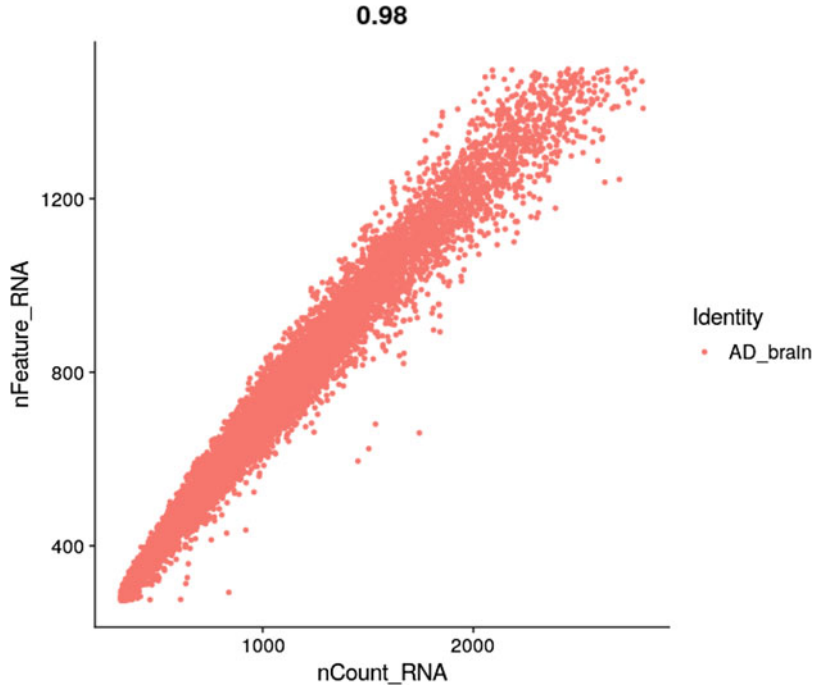


Fig. 3 Scatter plot of overall counts (x-axis) and number of distinct features (*y*-axis) per cell

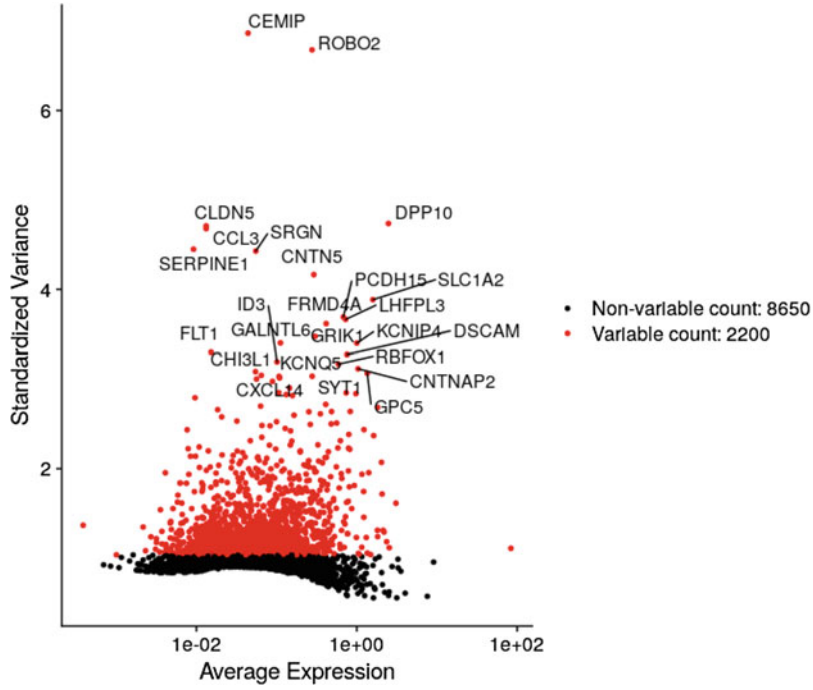


Fig. 4 Scatter plot of features by overall counts (*x*-axis) and variance across cells in standard deviations (*y*-axis). Features passing the filter are colored in red

In order to wrap our heads around high dimensional variance, we use the linear dimensionality reduction method principal component analysis (PCA). Instead of 2200 features, each cell is described by a small set of principal components that roughly represent its similarity to all other cells. The first principal components (PC1) represents the most variance, while each subsequent PC accounts for less variance.

```
# Scale counts to values with a mean of 0 and variance of 1
ds=ScaleData(ds, features=rownames(ds))
ds=RunPCA(ds, features=VariableFeatures(object=ds))
DimPlot(ds, reduction = "pca", group.by="condition")
```

Looking at just the two most significant PCs, we see that healthy samples are shifted away from the unhealthy ones on the *X*-axis (Fig. 5). This is a good result, denoting that the greatest amount of variance (PC1) partly corresponds with our main experimental condition. The first PC is clearly informative, but how can we find out how many PCs are representative of an actual cellular attribute? Ideally, biologically meaningful variance will be greater than noise and artifacts, using this assumption, let's examine the variance.

```
ElbowPlot(ds)
```

We can see a drop in the significance of the PCs around PC10, but it is not as clear-cut as we would hope (Fig. 6). Another helpful way to determine whether a PC is meaningful is by examining the features that contribute the most of its variance (Fig. 7) and its expression pattern in cells on both extremes of the PC (Fig. 8).

```
VizDimLoadings(ds, dims = 2, reduction = "pca")
```

The figures clearly illustrate that the expression of a handful of genes are highly distinct between a subset of cells (500 cells with the most extreme expression differences, in our example), but does this set of genes hold any sort of biological meaning? Using gene-set enrichment, we look for gene ontologies overly represented

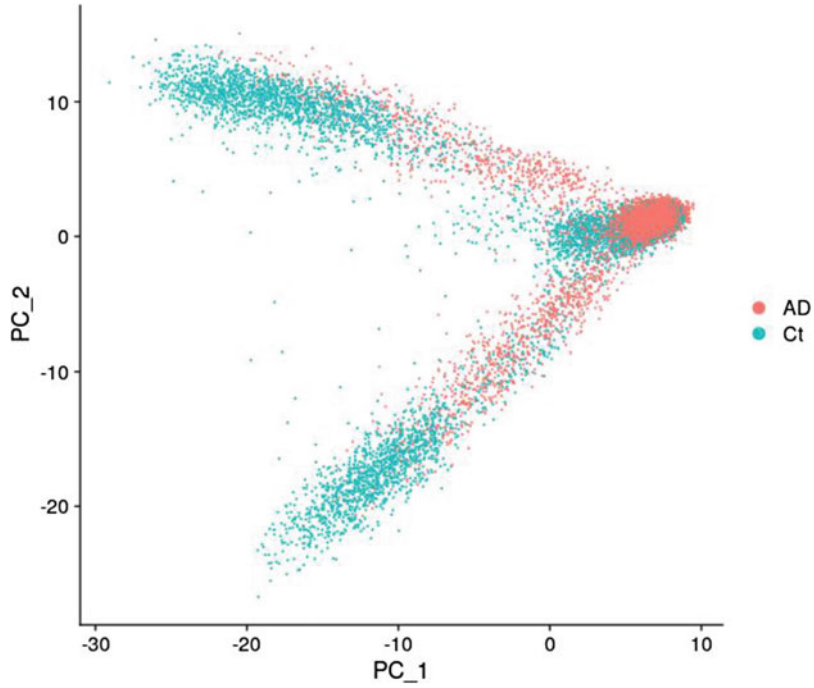


Fig. 5 Two-dimensional principal component analysis (PCA) plot of all cells

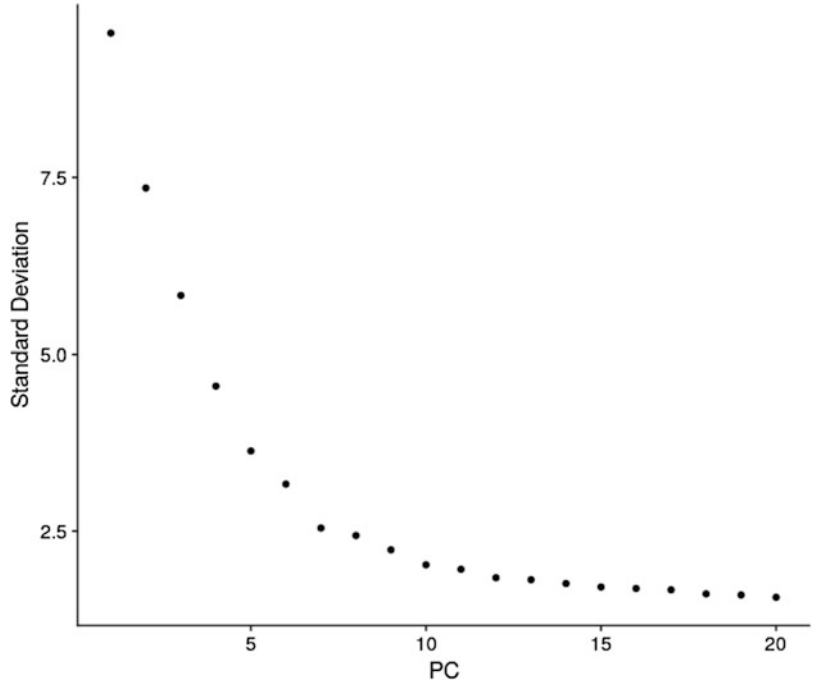


Fig. 6 Elbow plot of principal components (*x*-axis) by the amount of variance captured (*y*-axis)

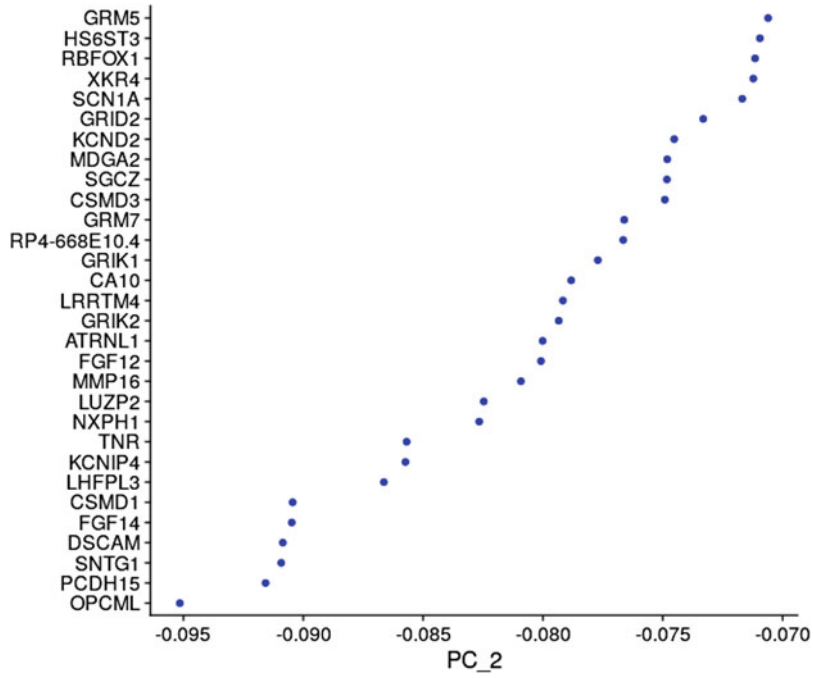


Fig. 7 Plot of features by their contribution to the variance of the second principal component

```
DimHeatmap(ds, dims = 2, cells = 500, balanced = TRUE)
```

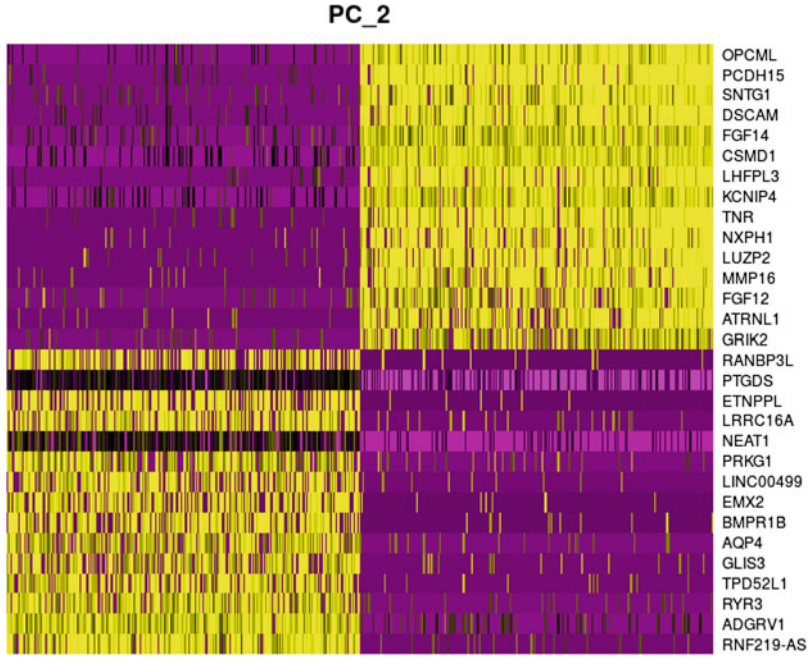


Fig. 8 Heatmap of the features with the largest contribution to the variance of the principal component

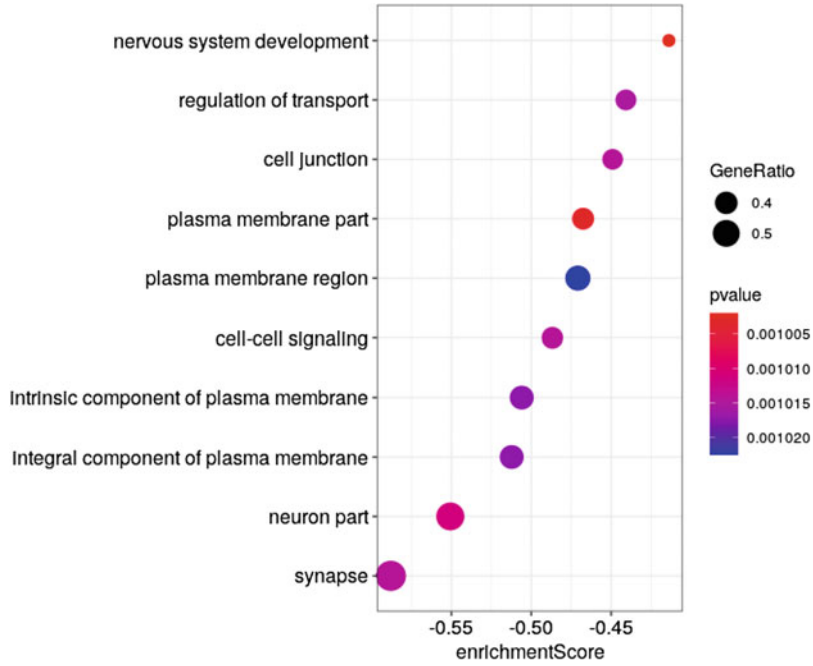


Fig. 9 Dotplot of the most enriched gene ontology terms in the set of genes that make up PC2. Represented metrics include enrichment score (*x*-axis), pvalue (color), and ratio of enriched genes out of the entire set (size)

among the most distinctive genes (Fig. 9).

```
# Obtain per gene contribution to PC2
featl=ds@reductions$pca@feature.loadings
gs=featl[,"PC_2"]
gs=sort(gs, decreasing=TRUE)

# Perform GO gene set enrichment analysis
library(org.Hs.eg.db)
library(clusterProfiler)

ego=gseGO(geneList=gs, OrgDb=org.Hs.eg.db, keyType="SYMBOL",
ont="ALL")
dotplot(ego, showCategory = 10, x = "enrichmentScore",
color="pvalue", size="GeneRatio")
```

As we can see, a significant portion of the genes that "make up" the PC have a common cellular component, molecular function or biological process, implying biological significance. If there is doubt regarding the significance of a PC, keep it in, we would rather have a slight misclustering than discarding meaningful data and the new clustering algorithms are more robust. Most importantly, the analysis could be repeated using different dimensionalities (Table 1).

3.4 Clustering

The following is a description of the Seurat v3 clustering algorithm. First, the FindNeighbors function applies K nearest neighbors (KNN), which uses the Euclidian distance between each pair of cells to group the cells together, but instead of calculating the distance based on the differences in gene expression values, it uses PC values and only the ones we deemed relevant.

The samples groups are now nodes in a graph, each pair connected by an edge with a weight value. The weight value is then refined according to Jaccard distance, or the similarity of their immediate neighbors. By the end of this phase, we converted our clustering problem to a graph problem. In the second phase, implemented in the FindClusters function, a modularity algorithm (Louvian modularity by default) is used to yield the clustering of the samples. Maximum modularity is achieved when the samples are grouped as such that the connection within the group is highest relative to connections outside the group. The number of clusters (also called granularity of clustering) is set using the resolution parameter, it is recommended to try several different granularity values.

```
ds=FindNeighbors(ds, dims=1:11)
ds=FindClusters(ds, resolution=0.5)
```

The clustering algorithm models the relationship between groups of samples in a nonlinear manner, therefore we complement it with a nonlinear dimensionality reduction method, like UMAP (Fig. 10).

```
ds=RunUMAP(ds, dims=1:12)
DimPlot(ds, reduction="umap", group.by="condition") + DimPlot(ds,
reduction="umap", label=TRUE)
```

3.5 Differential Expression

Now that the samples are clustered in a biologically meaningful manner, we can finally examine the differential expression between clusters and conditions. We start by relating clusters to cell types, this could be done by finding genes that are differentially expressed in one cluster compared to the rest (Table 2)(*see* **Note 4**).

```
c1_genes=FindMarkers(ds, ident.1=1, min.pct=0.33)
head(c1_genes, head=10) %>% knitr::kable()
```

Using a resource like CellMarker (http://biocc.hrbmu.edu.cn/CellMarker/index.jsp), we can determine the cell type represented by the cluster based on genes distinctly expressed in it. For example, genes like CNP are known to be distinct to Oligodendrocytes.

```
res=FindMarkers(ds, ident.1="AD", group.by="condition",
subset.ident="1", min.pct=0.33)
```

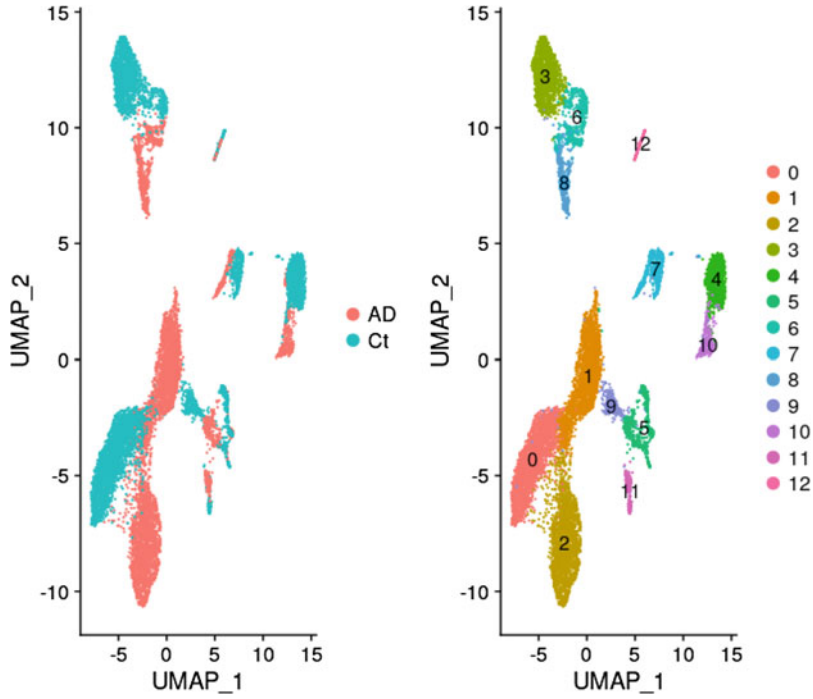


Fig. 10 UMAP dimensionality reduction plots, the right plot colored by experimental group condition (Alzheimer's disease and healthy control, colored orange and teal respectively), the left plot colored by cluster assignment

Table 1
The steps of our single-cell transcriptome profiling workflow, each with its purpose, the software implementations used for its demonstration in this chapter, and alternative software implementations that could be used instead

	Purpose	Software	Alternatives
Transcript quantification	Quantification of transcripts from sequencing data (raw or aligned)	Cell ranger (10× genomics)	Alignment based: STAR, TopHat alignment free: Cufflinks, RSEM
Normalization and general analysis	Normalization of count data, often includes other analysis methods	Seurat	Scater, scarn
Dimensionality reduction	Represent multidimensional data by fewer dimensions, relying on linear or nonlinear relations	PCA, UMAP	ICA, FA, tSNE, DCA
Clustering	Clustering of cells into groups of similar expression profiles	Seurat	SAIC, RaceID3, SC3

Each gene is accompanied by comparison statistics including *p*-value, average log-transformed foldchange, expression in cluster 1, expression outside cluster 1, and the corrected *p*-value

Table 2
Gene expression distinct to cluster 1 and common to at least 33% of its cells, with pvalue, log2 foldchange compared with the other cells, percentage of cells with detectable expression in and outside the cluster and the adjusted *p*-value

	p_val	avg_logFC	pct.1	pct.2	p_val_adj
HSPA1A	0	1.8143781	0.798	0.218	0
BOK	0	1.5664156	0.475	0.119	0
LINGO1	0	1.5488981	0.949	0.523	0
CRYAB	0	1.4573353	0.872	0.497	0
LINC00486	0	1.1187803	0.977	0.976	0
MALAT1	0	−0.6835068	0.999	1.000	0

According to our results, LINGO1 was significantly upexpressed in Oligodendrocytes of AD samples, in agreement with the results of the original paper [17] (*see* **Note 5**).

4 Notes

1. Unlike our data, raw sequencing data is sometimes saved in a multiplexed format, such as the Illumina base call file (BCL). In this case, demultiplexing into Fastq files can be performed using the Call Ranger utility mkfastq (https://support.10xgenomics.com/single-cell-gene-expression/software/pipelines/latest/using/mkfastq).
2. The per-sample output of Cell Ranger could be joined to a single file using the aggr command (Optional).
3. Single cell count matrices are very large, but also very sparse (contain mostly zeroes), since most reference genes aren't expressed in most cells. This is the reason we load the count matrix directly into a Seurat object, which represents sparse matrices in a memory efficient manner, instead of allocating many megabytes of memory for zeroes.
4. Partitioning of clusters for comparison could be adjusted using the FindMarkers function options, depending on the research question.
5. For easy access to the analysis results and associated visualization, the authors created a Shiny-based website: (http://adsn.ddnetbio.com/).

References

1. Wills QF, Livak KJ, Tipping AJ, Enver T, Goldson AJ, Sexton DW, Holmes C (2013) Single-cell gene expression analysis reveals genetic associations masked in whole-tissue experiments. Nat Biotechnol 31(8):748–752
2. Tang F, Barbacioru C, Wang Y et al (2009) mRNA-Seq whole-transcriptome analysis of a single cell. Nat Methods 6:377–382. https://doi.org/10.1038/nmeth.1315
3. Chen G, Ning B, Shi T. Single-Cell RNA-Seq Technologies and Related Computational Data Analysis. Front Genet. 2019 Apr 5;10:317. https://doi.org/10.3389/fgene.2019.00317. PMID: 31024627; PMCID: PMC6460256.
4. Habib N, Avraham-Davidi I, Basu A et al (2017) Massively parallel single-nucleus RNA--seq with DroNc-seq. Nat Methods 14:955–958. https://doi.org/10.1038/nmeth.4407
5. Islam S, Zeisel A, Joost S et al (2014) Quantitative single-cell RNA-seq with unique molecular identifiers. Nat Methods 11:163–166. https://doi.org/10.1038/nmeth.2772
6. Kivioja T, Vähärautio A, Karlsson K et al (2012) Counting absolute numbers of molecules using unique molecular identifiers. Nat Methods 9:72–74. https://doi.org/10.1038/nmeth.1778
7. Grün D, Kester L, van Oudenaarden A (2014) Validation of noise models for single-cell transcriptomics. Nat Methods 11:637–640. https://doi.org/10.1038/nmeth.2930
8. Elowitz MB, Levine AJ, Siggia ED, Swain PS (2002) Stochastic gene expression in a single cell. Science 297:1183–1186. https://doi.org/10.1126/science.1070919
9. Acar M, Mettetal JT, van Oudenaarden A (2008) Stochastic switching as a survival strategy in fluctuating environments. Nat Genet 40:471–475. https://doi.org/10.1038/ng.110
10. Jaitin DA, Kenigsberg E, Keren-Shaul H et al (2014) Massively parallel single-cell RNA-Seq for marker-free decomposition of tissues into cell types. Science 343:776–779. https://doi.org/10.1126/science.1247651
11. Papalexi E, Satija R (2018) Single-cell RNA sequencing to explore immune cell heterogeneity. Nat Rev Immunol 18:35–45. https://doi.org/10.1038/nri.2017.76
12. Tang X, Huang Y, Lei J et al (2019) The single-cell sequencing: new developments and medical applications. Cell Biosci 9:53. https://doi.org/10.1186/s13578-019-0314-y
13. Giladi A, Cohen M, Medaglia C et al (2020) Dissecting cellular crosstalk by sequencing physically interacting cells. Nat Biotechnol 38 (5):629–637. https://doi.org/10.1038/s41587-020-0442-2
14. Spanjaard B, Hu B, Mitic N et al (2018) Simultaneous lineage tracing and cell-type identification using CRISPR–Cas9-induced genetic scars. Nat Biotechnol 36:469–473. https://doi.org/10.1038/nbt.4124
15. Plass M, Solana J, Wolf FA et al (2018) Cell type atlas and lineage tree of a whole complex animal by single-cell transcriptomics. Science 360:eaaq1723. https://doi.org/10.1126/science.aaq1723
16. Butler A, Hoffman P, Smibert P et al (2018) Integrating single-cell transcriptomic data across different conditions, technologies, and species. Nat Biotechnol 36:411–420. https://doi.org/10.1038/nbt.4096
17. Grubman A, Chew G, Ouyang JF, Sun G, Choo XY, McLean C, Simmons RK, Buckberry S, Vargas-Landin DB, Poppe D, Pflueger J, Lister R, Rackh OJL (2019) A single-cell atlas of entorhinal cortex from individuals with Alzheimer's disease reveals cell-type-specific gene expression regulation. Nat Neurosci 22(12):2087–2097

Chapter 17

Biological Perspectives of RNA-Sequencing Experimental Design

Metsada Pasmanik-Chor

Abstract

The development of high-throughput technologies has changed the conduct of biological experiments in the last decade. From single gene studies, research has shifted to measuring gene signatures at the transcriptome level. The dramatic decrease in the financial expenses of next generation sequencing techniques has enabled their routine implementation. However, very often, economic constraints restrict the number of samples and sequence quality. Careful planning and design may overcome this limitation, and attain the maximum information from a given experiment.

Among the factors that affect the quality and quantity of data resulting from next generation sequencing experiments are sample size and the number of replicates, sequence depth and coverage, randomization, and batches. Here, we discuss the design of high-throughput experiments, while focusing on RNA-sequencing experiments. We suggest critical rules of thumb, from biological, statistical, and bioinformatics points of view, aimed to obtain a successful experiment, beyond the economic constraints.

Key words Experiment design, High-throughput (HT), Next generation sequencing (NGS), RNA-Seq, Sample variability, Sample size, Sequence depth, Sequence coverage, Batch, Differentially expressed genes (DGEs)

1 Introduction

Next generation sequencing (NGS) entails a novel set of cutting-edge technologies used for high-throughput (HT) data analysis aimed at determining nucleic acid sequences. These technologies have become standard tools that increase the amount, quality, and precision of transcription information; shorten sequencing time; and reduce the costs of experiments compared to previous sequencing methods. Methods that enable advanced genome mapping and reliable sequencing have been implemented in many laboratories for medical, basic research and industrial purposes. In recent years, the costs of NGS technologies dropped exponentially. This was coincident with the increase in the market to billions of dollars. This review will concentrate on the experimental design of

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_17,

RNA-Seq, the most widespread used transcriptome analysis. The main goal of RNA-Seq is the simultaneous measuring of the differential expression of thousands of genes (at the mRNA level), and the comparison of certain conditions or treatments with those of controls. Notably, many design issues arise in relation to NGS technologies. The experimental results can demonstrate tissue profiling and function, determine the genetic origin of cancers, identify biomarkers and gene signatures of cells and tissues, and suggest drug response.

Despite the massive use of NGS technologies nowadays, the biological design of experiments has received little attention. After NGS goals are defined, the rules of thumb for good experimental design follow standard biological experiment design. However, unique issues specific to NGS experiments must be considered, to attain optimal results and correct biological interpretation. Although various bioinformatics tools are available for NGS analysis, no dedicated software completely addresses automatic experimental design. Furthermore, despite its being the most crucial part of the study, NGS analysis is currently performed manually, aided by statisticians and\or bioinformaticians. R. A. Fisher, the founder of the experimental design statistics field of study [1] was quoted as saying, "To consult the statistician after an experiment is finished, is often merely to ask him to conduct a post mortem examination. He can perhaps say what the experiment died of." These words emphasize the importance of early involvement of the statistician in experimental design. As statistics have played a central role in the analysis of NGS experiment, these words are presently of particular importance.

Despite the drastic decreases in sequencing costs in the past few years, NGS experiments remain relatively expensive, rendering resources a major limiting constraint. Consequently, in most experiments, the sample number and sequencing properties are far less than desired for optimal statistical analysis. In some experiments, such concessions are possible, but in others they strongly impair the results. Moreover, biological replicates are often variable and background (noise) bias is ubiquitous. Large sample size is known to increase the precision of results by decreasing the effect of possible outliers. This is especially important when sample variability is high, which is often the case. Efforts to maximize experimental efficiency by proper advanced planning should be of high priority. Ideally, experimental planning should involve both the experimental biologist and the bioinformatician who would analyze the data. This multidisciplinary view may reduce pitfalls in the overall design and analysis, and maximize the precision of the experiment results and the validity of the biological conclusions.

Here we discuss the various facets to be considered when designing HT experiments, prior to their execution, with the aim of optimizing outcomes.

2 Materials and Methods: Experimental Design

2.1 Biological Sample Variability

NGS experiments are usually carried out with a hypothesis in mind and with the goal of affirming or refuting a given assumption. Mean or median expression values of treatments or conditions are calculated and compared to controls, and genomic profiling is evaluated. As biological data are heterogeneous in nature, identifying technical sources of variability, such as environmental bias, noise, and errors, is important for isolating the treatment effect. For example, in human samples, gender, age, ethnicity and unrelated diseases are crucial factors that may mask or hide treatment effects. Tissues that are composed of many cell-types are usually more variable than primary cells, and these are "noisier" than cell-lines, which are homogeneous in nature. In addition, not all tissues have the same degree of heterogeneity, which is often affected by the donor's lifestyle [2]. Breast cancer studies highlight the importance of tissue variability [3]. Clinical identification of the stage and type of breast cancer is essential for proper therapy (neoadjuvant endocrine or chemotherapy). To this end, prognostic panels of gene expression were introduced for determining treatment [3]. However, although assessing prognosis is theoretically simple, biological variability is often difficult to differentiate from "noise." This challenge is the main cause of therapy failure, although the molecular signature of breast cancer types has been characterized and published. The greater the sample variability, the larger the data required to achieve statistically significant, reliable molecular results. Understanding this and other design issues described below, may help reduce confounding effects due to variability. While such effects may be interesting, they are not related to the research questions and obscure the biological questions.

2.2 Sample Size

Large sample size may overcome sample heterogeneity, highlight distinct treatment effects, reduce background noise, and ensure sufficient statistical power to answer biological questions [4]. An optimal experimental setting, if no budget and ethical considerations exists, is unlimited sample size, the larger the better. But limited sample resources generally raise major fiscal considerations regarding experiment design. Several software programs were recently developed to address the critical issue of the number of experimental samples to be used in an experiment. The goal is to balance sample number and statistical power, for maximum accuracy [4, 5]. To estimate optimal sample size, datasets were simulated based on real data, using 10 or 50 million reads, producing 1000 or 3000 differentially expressed genes (DEGs) with 2, 3, or 5 replicates per condition [4]. However, tools that predict sample size were shown to produce different results. A small number of DEGs were obtained with two replicates, and increased

sequence depth caused only a minor change. Although increasing sequence depth is usually less expensive than increasing sample size, the latter is more advantageous. In addition, sample size estimation was shown to vary according to the type of analysis tools used. When the treatment effect is distinct, a small sample size may be sufficient. However, small effects often have major biological impact, but are hard to detect. When only a barely detected treatment effect is studied, more samples are required to significantly detect biological change.

2.3 Replicates and Pooling

For random and real effects, replicates of measurements are important. Biological replicates are independent samples that are prepared individually from the beginning of the experiment, and are therefore considered as reliable reflections of the biological entity of the samples [6]. Technical replicates are designed as a single biological material that is subdivided to multiple samples only for analysis, and are therefore expected to yield more homogeneous results. The reproducibility of RNA-Seq technology is known to be very high, but the preprocessing of the biological materials (such as RNA extraction and library preparation) entails noise. Therefore, technical replication is usually not requested, except in specific experiments in which biological variability is too high to assess [2]. Pooling of samples is often thought to replace biological replicates; accordingly, average expression levels result across all pooled samples. While pooling reduces experimental costs by lowering the sample size, such design prevents quality checks (QC) of individual samples and hides sample variations. If case pooling is used, stringent false discovery corrections and thorough validations should be applied. Poor correlation was observed in an analysis that compared pooled and individual samples; though this improved with an increased number of samples (from 3 to 8) [7]. Moreover, increasing replicate size was shown to be more effective in differential expression analysis than increasing sequence depth above ten million reads per sample (*see also* Subheading 2.7 "Sequence depth, coverage and quality score" below).

Statistical power of treatment effects is gained by increasing sample size. Only entries lower than a threshold p-value cutoff and a fold-change difference are considered statistically significant DEGs. *P* values adjusted by false discovery rate (FDR) have stronger statistical power and result in fewer false positives [8].

Knowledge of the biological material is crucial in the estimation of replicate number. Cell-line samples and inbred animal model strains are usually more homogeneous in nature than primary cells or whole tissue, and thus significant results may be obtained even with few replicates. In contrast, in whole subjects or tissues, or in the measure of environmental effects, especially when using human samples (and particularly when post-mortem tissues are used), data are more variable and a higher number of replicates is required to

obtain statistically significant results. Determination of the number of replicates should consider the effect of uncontrolled variables (noise) and the magnitude of the treatment effect. In addition, greater variability of the samples requires more replicates to conceal possible outliers or to detect small effects that could be of important biological consequences. Two or more biological replicates are generally needed for each treatment group, but the more the better, depending on the sample type, as described above. Sample number may also be predicted by reviewing the literature or simulating experiments previously performed using similar samples and conditions.

2.4 Randomization

Samples should be assigned and treated by chance, thus mitigating for technical bias. Randomization aims at creating homogeneous groups of samples, chosen by equal chance, which are mostly affected by the condition\treatment studied, while minimizing confounding bias as much as possible. However, complete randomization is considered as utopia in biological experiments. Simple random design can be achieved by coin tossing or dice rolling, or automatically generated by computers. Random block design is often used, to reduce heterogeneity within samples in a group and to ensure nearly equal numbers of samples in groups [9, 10]. However, certain experiments such as field experiments and clinical studies can be impossible to control, and are highly affected by confounding factors (e.g., age, sex, the growth chamber of plants). In clinical trials, stratified randomization is often used to keep variables similar within groups [11]. This 2-step procedure first groups patients according to a specific characteristic (age, gender, etc.), and then divides each group to subgroups that receive the treatments to be examined.

2.5 Batches

Homogeneous groups of samples (blocks) are theoretically designed to detect statistically significant differences between experimental groups. Such experimental design ensures that all samples are grown strictly in the same (optimal) conditions (chambers, rooms, etc.), with minimal influence of environmental stress. However, due to technical limitations, and especially when a large sample size is conducted, a single experimental batch is not always possible. More specifically, in human donor samples, controlling batches is usually impossible due to large variability in donors' lifestyle. Biological heterogeneity is sometimes hard to distinguish from technical, nonbiological factors. Artifacts of technical bias that create batches are often due to different technicians, experimental dates, reagents lots, and so on. The trend to decreasing costs of NGS experiments has led to increasing sample sizes; hence, overcoming and treating the resulting batches have gained importance. Notably, batches may be performed nonintentionally, according to the availability of biological samples. This may occur, for example,

when clinical samples are used from different human donors, operated on different days, and influenced by factors such as lifestyle, genetics, and treatments, which vary among people and may drastically influence the tissue gene signature [12]. Such technical bias is usually due to variations during sample availability, collection, processing, and sequencing [13]. Batches may mask the biological effect between the designed experimental groups, and are treated by batch-removal tools. The use of such tools is known to reduce statistical power [14]. Yet no algorithms automatically predict and detect various batches. This task is currently performed manually by researchers, based on biological knowledge and QC measures of the experiment. At the moment, these present a great challenge for algorithm developers.

Blocks of technical batches and treatment groups should not overlap. Accordingly, treatment and control samples should not be performed on separate dates. Rather, batches should contain a mix of treated and control samples, to avoid confounding technical and experimental factors [6]. Expected and unknown batches should be identified before analyzing and interpreting the data. These can often be viewed by means of graphical visualization of QC methods, applied to the normalized data. These methods include principal components analysis (PCA) and hierarchical cluster analysis. In PCA, the most common patterns that exist among features are estimated, and these are correlated with the treatment performed in the experiment [10]. Hierarchical cluster analysis is another way to graphically quantify features affected by treatment and batches among samples [15]. Data can be qualified in respect to sample homogeneity. If this does not match treatment design, known or obscure batches should be investigated. Large amounts of data in NGS experiments facilitate detecting batches and removing them. This is because this analysis is based on the multiple genes expressed, and errors are presumably only a small fraction of the data. In general, batches increase sample variability and decrease the statistical power of the biological signal [10], and thus may lead to misleading results. The importance of batch identification and neutralization was shown in the first report of the human and mouse ENCODE project, which resulted in gene expression clustering by species rather than by tissue [16]. This phenomenon was peculiar, as previous comparative studies reported high conservation between tissue gene signatures. After reanalysis of the results by batch removal, taking into account the confounding factors of different experiments performed for mice and humans, the data clustered by tissue, and not by species [17]. The idea behind batch removal software is to normalize the data based on known, predefined technical bias information, while retaining the biological effects intact [10, 14, 18, 19].

2.6 Multiplexing and Barcodes

Multiple samples can be sequenced on the same lane, assuming they can be separated prior to analysis. A short (6 bps) commercially supplied sample-specific oligonucleotide tag (barcode or index) is attached to each resulting DNA fragment, to distinguish samples one from another. The use of barcodes is particularly important when the sequence depth is high. Many reads are requested in an NGS experiment, which are not possible to obtain using only one lane of the flow-cell (Illumina). In this case, a few lanes are used and samples are equally divided and run on all lanes by using multiplexing, and then traced using barcode tags. The indexes are obviously specified and trimmed from the final sequences using bioinformatics tools before analysis [20].

2.7 Sequence Depth, Coverage and Quality Score

When resources are limited, typical RNA-Seq experiments tend to be under-sequenced, resulting in statistical bias in DEG detection [13]. Sequence depth is a quality measure for nucleotide sequencing. Accordingly, the number of times is counted that each base-pair is sequenced at high quality aligned reads that overlap a certain locus. Coverage is the average number of reads that align to a reference sequence, based on gene length and quality after alignment, and is a good measure of sequencing reliability. High sequence depth and coverage ensure credibility of the data, and may exclude sequence errors that could be mistakenly exchanged with variants [21]. Good sequence depth and coverage are crucial factors for robust statistical data analysis, and especially for the identification of low expressed genes. Several tools are available that deal with calculating the levels of sequence depth and the coverage required for experiments (for a recent review *see* [22]). These depend on RNA quality, read length, the type of sequences, the number of clusters run, quality of reference genome, and paired or unpaired reads. Sequence quality scores estimate the probability that a base is labelled correctly. Paired-end (PE) sequencing provides twice as much sequence data as single-read (SR), thus increasing the quality of mapping and improving sequence depth. However, the drawback of such is the higher expense and time investment. PE, deep sequencing, and high coverage are essential in the detection of low expressed genes, alternative splicing, epigenetic studies and mutation analysis, and when identifying novel transcripts and de novo sequencing. In such experiments, at least 100–200 million reads are suggested for each sample. A major question remains as to whether more biological replicates or greater sequence depth is preferred for robust data analysis.

Gene expression may be high or low; likewise, the expression of rare transcripts [23]. Longer genes are obviously better detected by NGS technology than shorter ones, as they harbor more reads. High sequence depth and coverage ensure more reliable detection with less errors when dealing with short genes, low expressed genes and small RNAs. Moreover, most of the statistical methodologies

that have been proposed rely on parametric assumptions and use negative binomial distributions to model gene counts. These statistical considerations often create biases in detecting low expressed genes. However, when high sequence depth is used, even after data normalization, the data may become noisy and some DEGs may appear as false-positives. Depletion of the ribosomal RNA fraction, which constitutes more than 90% of the RNA fraction in mammals, may specifically increase the number of mRNA reads that result in an experiment [21]. Various human datasets with different sequence depths were examined in order to evaluate optimal sequencing conditions [23]. Protein-coding and lncRNA data quantities were found to be abundant even when the sequence depth was low. This is compared to other noncoding RNAs (e.g., microRNAs, snoRNAs, pseudogenes) and low expressed transcripts, which may be of a regulatory function. Sequence depth is usually not the limiting factor of RNA-Seq experiments, as only half the sequence depth used (only 100 million reads) was found to be necessary for obtaining DEGs [24]. In another study, various statistical methods revealed different numbers of DEGs, and DEG expression was highly correlated with increased sequence depth [23]. Increased sequence depth also resulted in more false positives. Although normalization methods are commonly used, maintaining similar sequence depth in all samples is recommended. In general, a sequence depth of ten million reads in an RNA-Seq experiment will ensure that approximately 90% of all the genes will be covered by at least 10 reads [25].

2.8 NGS Platforms

Many NGS platforms are available, each has its specific characteristics, its pros and cons. The sequencing specifications of each platform should be considered, including read length, paired sequencing and error rate. This should be at the planning stage of the experiment, simultaneous with economic considerations, which are often the major issue considered when planning an experiment. Sequencing technologies and their properties and consequences have been detailed [26]. Illumina platforms currently dominate the market; they are very powerful and the results have been filling databases and the literature. De novo sequencing and structural genomic variants are the main limitation of these short-read techniques. Techniques that produce longer reads (e.g., Pacific Biosciences and Oxford Nanopore platforms) have been shown to yield better data accuracy [26].

2.9 Validations

A practical way to evaluate and quantify the effect of an experimental treatment or condition may be a wet-laboratory quantification step of a few selected known genes. This is usually performed by applying polymerase chain reaction (PCR) analysis of specific key genes, even prior to the NGS experiment. Other validations such as Northern, Western, Nanostring, and mass-spec may be performed.

Real-time or quantitative PCR is a simple low-cost technology that estimates the expression level of specifically selected genes in RNA samples, based on their unique nucleotide primer design. The amount of starting material is low and has little quantification limits. The results may suggest if the treatment performed in the biological experiment was successful and detectable compared to the control, and may indicate a need to perform a subsequent HT, and much more expensive, experiment. Genes are chosen according to their known functional importance and specific involvement in gene ontology or in a pathway (e.g., from previously published results or databases), or from unpublished knowledge. Genes that are known as drug targets are often preferred for validation, as they may later serve as critical factors in designing possible treatments. Genes that are specific to a cell type or tissue may be useful for validating cell or tissue constitution. The PCR experiment may ensure effectiveness of the treatment, and is often used to confirm the reliability of NGS results. It is thus known as the "gold standard" validation for HT experiments.

2.10 Result Reporting and Presentation

After careful design and precise laboratory and data analysis, the process culminates with intuitive and simple demonstration of the results. The NGS results should be presented in a user-friendly, easy to read and understandable format that use minimal space and text. Graphical presentation is especially important in HT experiments, as the results often contain large amounts of information that become impossible to observe and understand by tables and text alone. QC graphs are routinely presented by NGS analysis tools, to estimate the success of treatment and the uniformity of replicates, and thus, the reliability of the results [27]. Logarithmic scales can be used to distribute large range expression values more evenly across scales. The drawback of these illustrations is that outliers and unequal sample size are often hidden when summarizing results. Dot plots and scatterplots of large sample data may therefore better present HT data.

In addition to graphical representation, description of the data, experiment files, analysis tools, parameters, and results should be organized and uploaded onto an NGS repository. Such public repositories enable free data access, and downloading and preservation of results, and exist in all major NGS centers (GEO—https://www.ncbi.nlm.nih.gov/geo/, SRA—https://www.ncbi.nlm.nih.gov/sra, ArrayExpress—https://www.ebi.ac.uk/arrayexpress/, etc.). Availability of these files is important for future research (as seen in [17]), and is usually a requirement for manuscript submission.

3 Conclusions

Careful design, randomization of samples, and minimal technical variations, in addition to large sample size and deep sequencing, are crucial issues that lead to the success of HT experiments. Achieving these factors may prevent performing unnecessary batches, and reduce background and noise that could hamper statistically significant results. New sophisticated NGS technologies and instruments are quickly evolving. These are more accurate and produce a larger amount of high-quality sequences in a short time and at lower cost. Following these advances, new algorithms have been developed for analyzing results, and for data deposition to standardized databases from which data can be stored and retrieved. Means of assessing treatment effect have been improved and simplified; this enables comprehensive data analysis with significant results. Biological interpretation and graphical presentation of the most crucial, carefully selected, results highlight the quality of the results and the reliability and importance of the experiment. Advances in NGS technologies have led to a dramatic shift in analyzing clinical diagnostics issues and powerful drug matching, and in other molecular profiling, including pathogen sequencing and forensic studies. Organized and standardized storage hardware enables additional use of the free available data.

We have presented and demonstrated various issues that need to be addressed in experiment design, based on biological, statistical, and bioinformatics knowledge. Considerations of these complex topics may aid in the design of appropriate, high quality experiments, and the achievement of reliable results that may overcome limitations due to economic constraints.

References

1. Craig CC, Fisher RA (1936) The design of experiments. Am Math Mon 43:180–181. https://doi.org/10.2307/2300364
2. The ENCODE Consortium (2011) Standards, guidelines and best practices for RNA-Seq. Vasa. https://doi.org/10.1073/pnas.0703993104
3. Guler EN (2017) Gene expression profiling in breast cancer and its effect on therapy selection in early-stage breast cancer. Eur J Breast Health 13(4):168–174. https://doi.org/10.5152/ejbh.2017.3636
4. Poplawski A, Binder H (2018) Feasibility of sample size calculation for RNA-seq studies. Brief Bioinform 19(4):713–720. https://doi.org/10.1093/bib/bbw144
5. Zhao S, Li CI, Guo Y, et al (2018) RnaSeqSampleSize: real data based sample size estimation for RNA sequencing. BMC Bioinformatics 19(1). https://doi.org/10.1186/s12859-018-2191-5
6. Klaus B (2015) Statistical relevance—relevant statistics, part I. EMBO J 34(22):2727–2730. https://doi.org/10.15252/embj.201592958
7. Rajkumar AP, Qvist P, Lazarus R et al (2015) Experimental validation of methods for differential gene expression analysis and sample pooling in RNA-seq. BMC Genomics 16:548. https://doi.org/10.1186/s12864-015-1767-y
8. Benjamini Y, Hochberg Y (1995) Controlling the false discovery rate: a practical and powerful approach to multiple testing. J R Stat Soc Ser B 57:289–300. https://doi.org/10.1111/j.2517-6161.1995.tb02031.x

9. Matts JP, Lachin JM (1988) Properties of permuted-block randomization in clinical trials. Control Clin Trials 9(4):327–344. https://doi.org/10.1016/0197-2456(88)90047-5
10. Leek JT, Scharpf RB, Bravo HC et al (2010) Tackling the widespread and critical impact of batch effects in high-throughput data. Nat Rev Genet 11(10):733–739
11. Sil A, Kumar P, Kumar R, Das N (2019) Selection of control, randomization, blinding, and allocation concealment. Indian Dermatol Online J 10(5):601–605. https://doi.org/10.4103/idoj.idoj_149_19
12. Festing MFW, Altman DG (2002) Guidelines for the design and statistical analysis of experiments using laboratory animals. ILAR J 43(4):244–258. https://doi.org/10.1093/ilar.43.4.244
13. Bass AJ, Robinson DG, Storey JD (2019) Determining sufficient sequencing depth in RNA-Seq differential expression studies. bioRxiv. https://doi.org/10.1101/635623
14. Nygaard V, Rødland EA, Hovig E (2016) Methods that remove batch effects while retaining group differences may lead to exaggerated confidence in downstream analyses. Biostatistics 17(1):29. https://doi.org/10.1093/biostatistics/kxv027
15. Guess MJ, Wilson SB (2002) Introduction to hierarchical clustering. J Clin Neurophysiol 19(2):144–151
16. Lin S, Lin Y, Nery JR et al (2014) Comparison of the transcriptional landscapes between human and mouse tissues. Proc Natl Acad Sci U S A 111(48):17224–17229. https://doi.org/10.1073/pnas.1413624111
17. Mizrahi-Man O, Gilad Y (2015) A reanalysis of mouse ENCODE comparative gene expression data. F1000Res 4:121. https://doi.org/10.12688/f1000research.6536.1
18. Papiez A, Marczyk M, Polanska J, Polanski A (2019) BatchI: batch effect identification in high-throughput screening data using a dynamic programming algorithm. Bioinformatics 35(11):1885–1892. https://doi.org/10.1093/bioinformatics/bty900
19. WWB G, Wang W, Wong L (2017) Why batch effects matter in omics data, and how to avoid them. Trends Biotechnol 35(6):498–507
20. Jiang H, Lei R, Ding SW, Zhu S (2014) Skewer: a fast and accurate adapter trimmer for next-generation sequencing paired-end reads. BMC Bioinformatics 15:182. https://doi.org/10.1186/1471-2105-15-182
21. Sims D, Sudbery I, Ilott NE et al (2014) Sequencing depth and coverage: key considerations in genomic analyses. Nat Rev Genet 15(2):121–132
22. Wiewiórka M, Szmurło A, Kuśmirek W, Gambin T (2019) SeQuiLa-cov: a fast and scalable library for depth of coverage calculations. Gigascience 8(8). https://doi.org/10.1093/gigascience/giz094
23. Tarazona S, García-Alcalde F, Dopazo J et al (2011) Differential expression in RNA-seq: a matter of depth. Genome Res 21(12):2213–2223. https://doi.org/10.1101/gr.124321.111
24. Griffith M, Griffith OL, Mwenifumbo J et al (2010) Alternative expression analysis by RNA sequencing. Nat Methods 7(10):843–847. https://doi.org/10.1038/nmeth.1503
25. Hart SN, Therneau TM, Zhang Y et al (2013) Calculating sample size estimates for RNA sequencing data. J Comput Biol 20(12):970–978. https://doi.org/10.1089/cmb.2012.0283
26. Levy SE, Myers RM (2016) Advancements in next-generation sequencing. Annu Rev Genomics Hum Genet 17:95–115. https://doi.org/10.1146/annurev-genom-083115-022413
27. Klaus B (2016) Statistical relevance—relevant statistics, part II: presenting experimental data. EMBO J 35(16):1726–1729. https://doi.org/10.15252/embj.201694659

Chapter 18

Analysis of microRNA Regulation in Single Cells

Wendao Liu and Noam Shomron

Abstract

MicroRNAs (miRNAs) regulate gene expression by binding to mRNAs. Consequently, they reduce target gene expression levels and expression variability, also known as "noise." Single-cell RNA sequencing (scRNA-seq) technology has been used to study miRNA and mRNA expression in single cells, and has demonstrated its strength in quantifying cell-to-cell variation. Here we describe how to investigate miRNA regulation using data with both mRNA and miRNA expression in single cell format. We show that miRNAs reduce the expression levels and also expression noise of target genes in single cells. Finally, we also discuss potential improvements in experimental design and computational analysis of scRNA-seq in order to reduce or partition the technical noise.

Key words Single-cell sequencing, miRNA regulation, Gene expression level, Gene expression noise, Single-cell data denoising

1 Introduction

MicroRNAs (miRNAs) are small noncoding RNA molecules which regulate gene expression in metazoan organisms. They function post-transcriptionally by regulating target genes through facilitated mRNA degradation or translational repression. Therefore, they are expected to reduce mRNA and protein levels [1–3]. Apart from the effect on the level of gene expression, miRNAs also function in reducing gene expression variability, or "noise," in particular of lowly expressed genes [4, 5]. It is hypothesized that this effect reduces the detrimental stochasticity in gene expression and confers robustness to genetic pathways [6].

Single-cell RNA sequencing (scRNA-seq) is a recently emerging and rapidly developing technology enabling direct profiling of gene expression in a single cell resolution. Cell-to-cell variability can be quantified with scRNA-seq, which provides deep insights into expression levels and heterogeneity, and the stochastic features of gene expression [7]. A derivative of scRNA-seq, single-cell small RNA sequencing, reveals the expression pattern of the

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_18,

small fraction of RNAs, including miRNAs, tRNAs, and snoRNAs [8]. In order to investigate the miRNA–mRNA regulatory interplay, the majority of previous studies integrated miRNA–mRNA data from bulk sequencing experiments (as opposed to single cell) focusing on the anticorrelative expression levels between the miRNAs and their target genes [9, 10]. These studies were limited in their ability to measure the effect of miRNAs on the mRNA expression variability. Given the advancements of single cell mRNA and miRNA sequencing technologies, and the availability of the sequencing data, it is possible to explore the interplay between miRNA levels and noisy gene expression [11].

Despite scRNA-seq's many strengths over other technologies, various technical factors cause substantial challenges in analyzing single-cell data. For example, one such hurdle is overcoming the technical noise in single-cell sequencing experiments [12]. Some approaches have been proposed to resolve these technicalities, such as unique molecular identifier (UMI) [13] and external RNA spike-ins [14]. Nevertheless, technical noise in scRNA-seq is greater than that in bulk RNA-seq (non-single cell) and hence the accurate quantification and decomposition of technical and biological noise in scRNA-seq remains challenging. Here we introduce some computational approaches to denoise single-cell data, and we show how they can improve the analysis of miRNA regulation in single cells.

2 Materials

2.1 Software

1. Download and install the latest version of *R* (https://www.r-project.org/), and *Rstudio* (https://rstudio.com/).

   For advanced programmers, we suggest installing Jupyter notebook with *R* kernel (*see* **Note 2**) and run the example notebook provided by us (*see* **Note 1**).

2. Download and install *Anaconda* (https://www.anaconda.com/), an open-source distribution of Python for scientific computing. Then install *TensorFlow* (https://www.tensorflow.org/install) [15] and *DCA* (Deep Count Autoencoder, https://github.com/theislab/dca) [16] using Python package manager pip (*see* **Note 3**).

2.2 Data Files

We will illustrate the workflow using a dataset from Wang et al. 2019 [17] and miRNA target prediction results from TargetScan [18] (*see* **Note 4**). Wang et al. performed co-sequencing of miRNAs and mRNAs in same K562 single cells using a half-cell genomics approach. The dataset is available at GSE114071 (https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE114071). It contains mRNA and miRNA co-sequencing results of 19 successfully profiled single cells. Specifically, single-cell mRNA sequencing

is conducted using Smart-seq protocol [19]. Unluckily, the raw read count matrix of mRNA is not provided in this dataset, which is required in our analysis. Therefore, we quantified gene expression from raw sequencing files using RSEM [20], with GENCODE v27 annotation (*see* **Note 5**). All data are available at https://github.com/nshomron/sc-miReg/tree/master/data.

1. Predicted target genes of miRNA families from TargetScan, **Predicted_Targets_Info.default_predictions.txt**.
2. Information of miRNAs and their families from TargetScan, **miR_Family_Info.txt**.
3. miRNA expression table from dataset GSE114071, **GSE114071_NW_scsmRNA_K562_norm_log2.gct**.
4. mRNA raw read count table from RSEM output, **GSE114071_count.tsv**.
5. mRNA RPKM table from RSEM output, **GSE114071_RPKM.tsv**.
6. mRNA effective length table from RSEM output, **GSE114071_length.tsv**.
7. Denoised mRNA expression table from DCA output, **GSE114071_DCA.tsv**.

3 Methods

In this section, we will use mRNA expression table in unit RPKM (*see* **Note 6**) to estimate the expression levels and noise of mRNAs, and then measure how miRNAs regulate them. All codes in this section are in R programming language.

3.1 Import Data

1. Read and process miRNA expression matrix. We retain columns containing the name of miRNAs and expression values of 19 successfully profiled single cells. After that, column 1 contains the names of miRNAs and column 2 ~ 20 correspond to 19 successfully profiled single cells. Expression values in the matrix are normalized read counts. The raw read count of a miRNA is divided by the library size, applied with a minimum expression level of 10^{-4} (i.e., values less than 10^{-4} were set to 10^{-4}), and log2-transformed to get the normalized read count. We call this normalized read count "log2 fraction." The minimum value in the matrix is $\log_2(10^{-4}) = -13.287712$.

```
mirna <- read.table('GSE114071_NW_scsmRNA_K562_norm_log2.
gct', header=T, stringsAsFactors=F)

mirna <- mirna[,c(1,3:21)]
```

2. Read and process mRNA expression matrix. The expression values in the matrix are RPKM values. We match the columns of this matrix to the columns of miRNA expression matrix, such that column 1 contains the names of genes and column 2 ~ 20 contain RPKM values of successfully profiled single cells.

```
mrna_rpkm <- read.table('GSE114071_RPKM.tsv', header=T,
stringsAsFactors=F)

mrna_rpkm <- mrna_rpkm[,c(1:6,8:21)]

colnames(mrna_rpkm) <- colnames(mirna)
```

3. Read target prediction table from TargetScan. As TargetScan predicts target genes by miRNA families, we will use "family_info" to get the miRNA family of each miRNA, and "TS" to get the predicted target genes of each miRNA family.

```
TS <- read.table('Predicted_Targets_Info.default_predic-
tions.txt', header=T, sep='\t', stringsAsFactors=F)

family_info <- read.table('miR_Family_Info.txt', header=T,
sep='\t', stringsAsFactors=F)
```

3.2 Estimate Expression Levels and Noise of mRNAs

In this section, we are going to estimate the expression levels (average expression) and expression noise (the variability of expression) of mRNAs in single cells with RPKM values.

1. Calculate mean and SD of RPKM. In order to get an accurate estimate of mean and variability, we first filter out mRNAs expressed in less than five single cells. Then we calculate the mean and SD of RPKM of retained mRNAs. These meta data are saved in a data frame "meta".

```
index_cells <- 2:20

gene_filt <- which(apply(mrna_rpkm[,index_cells] > 0, 1, sum)
>= 5)

meta <- data.frame(gene=mrna_rpkm[gene_filt, 'Name'])

meta$mean_rpkm <- apply(mrna_rpkm[gene_filt, index_cells],
1, mean)

meta$sd_rpkm <- apply(mrna_rpkm[gene_filt, index_cells],
1, sd)
```

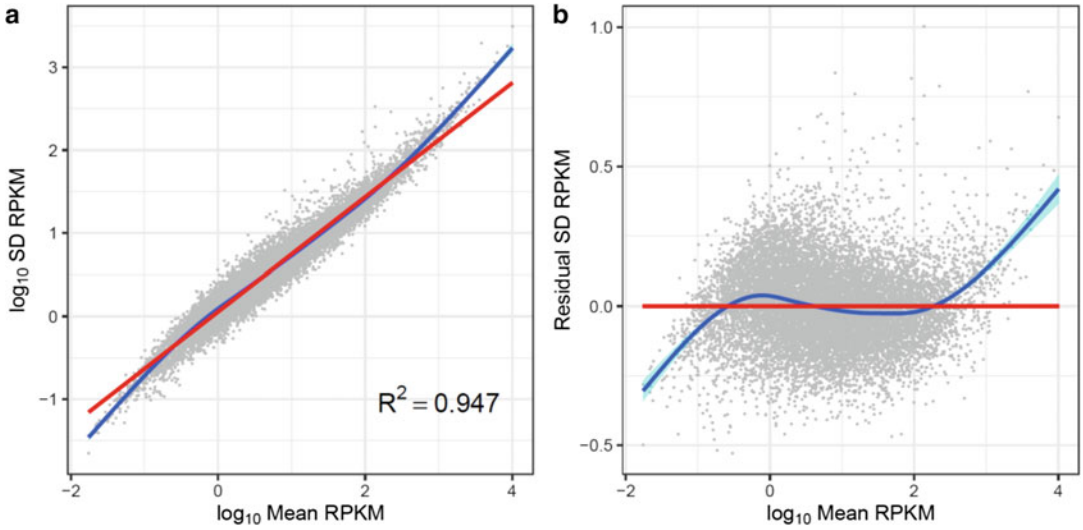


Fig. 1 (**a**) Linear relationship between mean and SD of RPKM values of genes on a logarithmic scale. The red line is fitted by ordinary least squares (OLS) regression and the blue line is fitted by smoothing method Generalized Additive Models (GAM). (**b**) Residual SD RPKM values are defined as residuals of the linear regression model. Overall, Residual SD RPKM measures the relative noise of mRNAs compared to the mean noise of mRNAs at the same expression level

2. Calculate Residual SD of RPKM. Unluckily, SD is not a good measure to quantify expression noise of mRNAs in single cells. This is because SD RPKM values generally tend to increase when mean RPKM values increase in single cells. In other word, mean RPKM and SD RPKM are not independent. Figure 1a displays the strong linear correlation between mean RPKM and SD RPKM values on a logarithmic scale ($R^2 = 0.947$).

```
fit <- lm(log10(sd_rpkm) ~ log10(mean_rpkm), meta)

summary(fit)
```

If we choose SD RPKM values as expression noise of mRNAs, highly expressed mRNAs will always have higher expression noise. This is undesirable, because in this way we cannot separately measure miRNAs' regulative effect on target genes' expression levels and noise. Therefore, we have to account for the effect of mean RPKM on SD RPKM. A good solution is to choose the residuals of the linear model after regressing out mean RPKM. We call them "Residual SDs". A Residual SD measures the relative noise of a mRNA compared to the mean noise of mRNAs at a same expression level.

Figure 1b shows that it is not linearly correlated with mean RPKM, so we will use it as the expression noise.

```
meta$rsd_rpkm <- fit$residuals
```

In summary, we will use **mean RPKM** (meta$mean_rpkm) as the expression level and **Residual SD RPKM** (meta $rsd_rpkm) as the expression noise of mRNAs in the following analysis.

3.3 Measure miRNA Regulation

After quantifying the expression levels and expression noise of mRNAs, we now focus on measuring how miRNAs regulate target mRNAs' expression levels and expression noise.

1. Target prediction using TargetScan. Here we define a function "mir2meta_var" to get the target mRNAs' expression levels and expression noise of any miRNA. As TargetScan predicts target mRNAs by miRNA families, we first obtain the families of miRNAs, and then get target mRNAs. Finally, we get the expression levels or noise of target mRNAs, which has already been saved in "meta".

```
mir2meta_var <- function(mir, meta_var)

{

mirna_family <- family_info[family_info$MiRBase.ID %in% mir,
'miR.family']

 mirna_target <- unique(TS[TS$miR.Family %in% mirna_family,
'Gene.Symbol'])

 target_meta_var <- meta[meta$gene %in% mirna_target, meta_-
var]

return(target_meta_var)

}
```

There are two input arguments for this function. "mir" should be a string vector of miRNA names, and "meta_var" should be the name of a variable saved in the data frame "meta".

2. Divide miRNAs into groups by expression levels. We assume that highly expressed miRNAs will be more likely to regulate their target mRNAs. Therefore, we divide miRNAs into four groups according to their expression levels:

   Background: $0.0001 <$ mean fraction in single cells $\leq 2^{-12} \approx 0.000244$.

   Lowly expressed (LE): $2^{-12} <$ mean fraction in single cells $\leq 2^{-9} \approx 0.001953$.

   Mediumly expressed (ME): $2^{-9} <$ mean fraction in single cells $\leq 2^{-6} = 0.015625$.

   Highly expressed (HE): mean fraction in single cells $> 2^{-6} = 0.015625$.

   For miRNAs in each group, we get their target mRNAs' expression levels and noise using the function "mir2meta_var" defined above (*see* **Note 7**).

```
mirna_mean <- apply(mirna[,index_cells], 1, mean)

min_expression <- min(mirna[,-1])

level_background <- mir2meta_var(mirna[mirna_mean > min_e-
xpression & mirna_mean <= -12, 1], 'mean_rpkm')

level_LE_mirna <- mir2meta_var(mirna[mirna_mean > -12 &
mirna_mean <= -9, 1], 'mean_rpkm')

level_ME_mirna <- mir2meta_var(mirna[mirna_mean > -9 & mir-
na_mean <= -6, 1], 'mean_rpkm')

level_HE_mirna <- mir2meta_var(mirna[mirna_mean > -6, 1],
'mean_rpkm')

noise_background <- mir2meta_var(mirna[mirna_mean > min_e-
xpression & mirna_mean <= -12, 1], 'rsd_rpkm')

noise_LE_mirna <- mir2meta_var(mirna[mirna_mean > -12 &
mirna_mean <= -9, 1], 'rsd_rpkm')

noise_ME_mirna <- mir2meta_var(mirna[mirna_mean > -9 & mir-
na_mean <= -6, 1], 'rsd_rpkm')

noise_HE_mirna <- mir2meta_var(mirna[mirna_mean > -6, 1],
'rsd_rpkm')
```

Table 1
The numbers of miRNAs and their target mRNAs in different groups. miRNAs are grouped by their expression levels

Group	miRNA number	Target mRNA number
Background	210	6201
LE	31	4914
ME	19	3632
HE	5	1144

We can check the numbers of miRNA and their target mRNAs in each group and save them in a data frame "group_size". The results are shown in Table 1 (*see* **Note 8**).

```
group_size <- data.frame(mirna_expression=c('background','LE','ME','HE'), mirna_number=c(sum(mirna_mean > min_expression & mirna_mean <= -12), sum(mirna_mean > -12 & mirna_mean <= -9), sum(mirna_mean > -9 & mirna_mean <= -6), sum(mirna_mean > -6)), target_number=c(length(level_background), length(level_LE_mirna), length(level_ME_mirna), length(level_HE_mirna)))
```

In order to test whether target mRNAs' expression levels or noise in different groups are from the same distributions, we use a nonparametric test named Kruskal–Wallis test. Figure 2a, c show that target mRNAs' expression levels or noise in different groups are from different distributions ($P = 1.22 \times 10^{-9}$ for expression levels, and $P = 2.05 \times 10^{-5}$ for expression noise). The statistical significance of difference between expression levels and noise from any two groups can be determined with Kolmogorov–Smirnov (KS) tests, which is displayed in Fig. 2b, d (*see* **Note 9**).

```
meta_var_df <- data.frame(group=rep(group_size$mirna_expression, group_size$target_number), level=c(level_background, level_LE_mirna, level_ME_mirna, level_HE_mirna), noise=c(noise_background, noise_LE_mirna, noise_ME_mirna, noise_HE_mirna))

kruskal.test(level ~ group, data = meta_var_df)

kruskal.test(noise ~ group, data = meta_var_df)

ks.test(level_background, level_LE_mirna)

ks.test(noise_background, noise_LE_mirna)
```

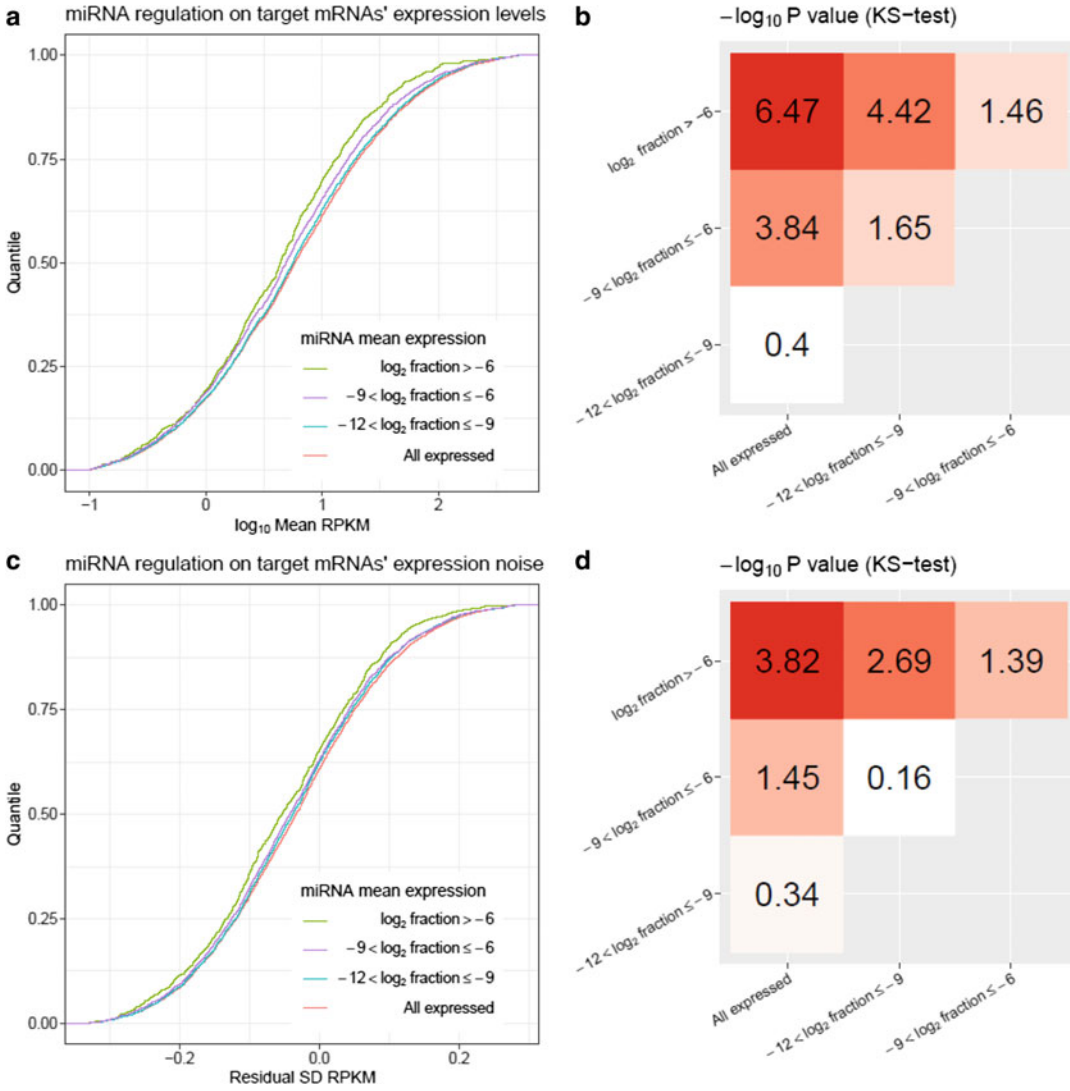


Fig. 2 (**a**) Empirical cumulative distribution functions (ECDFs) of target mRNAs' expression levels, estimated by mean RPKMs. Target mRNAs are grouped by miRNA mean expression levels. When miRNA mean expression level increases, their target mRNAs' expression levels significantly decrease. (**b**) $-\log_{10}$ *P*-values between any two ECDFs in (**a**) calculated using KS-test. (**c**) ECDFs of target mRNAs' expression noise, estimated by Residual SD RPKMs. When miRNA mean expression level increases, their target mRNAs' expression noise significantly decreases. (**d**) $-\log_{10}$ *P*-values between any two ECDFs in (**c**) calculated using KS-test

Based on these results, we find that highly expressed miRNAs have significantly lower expression levels and noise than those of lowly expressed miRNAs. This suggests that the overall expression levels and noise of target mRNAs tend to decrease as miRNA mean expression levels increase. In other word, miRNAs regulate their target mRNAs by reducing their expression levels and noise (*see* **Note 10**).

4 Using DCA to Denoise Gene Expression Matrix

It is well established that scRNA-seq can provide valuable insights about the variability within a population of cells. However, various technical factors like amplification bias, library size differences and low RNA capture rate lead to substantial technical noise in scRNA-seq experiments. A most common feature of single-cell data is the sparsity of gene expression matrix. This is because some transcripts are not captured in reverse transcription step and it produces false zero read counts. These false zero read counts are also known as "dropout" events. To address this problem, some tools have been developed to denoise gene expression matrix for the downstream analysis [21–23]. Here we introduce a denoising method named Deep Count Autoencoder (DCA) [16]. It takes a raw count matrix as input, denoises the matrix with an autoencoder, and outputs the denoised count matrix.

4.1 Run DCA

Type the following command in Shell to run DCA with default parameters. The two input arguments are input data matrix and the name of output directory. Specify --help argument for more information about adjusting parameters in the model.

```
dca GSE114071_count.tsv GSE114071_DCA_results
```

The denoised, library size-normalized count matrix is available at "**GSE114071_DCA_results/mean_norm.tsv**". We rename it into "**GSE114071_DCA.tsv**" in the downstream analysis. Notice that if you run DCA multiple times, the output matrices will be slightly different. Here we demonstrate the workflow using only one output matrix, but validating the result by repeating the analysis with more output matrices is also favorable.

4.2 Process DCA Output

DCA is designed for denoising UMI count matrix (*see* **Note 6**), so it does not take gene length into account. As Smart-seq is designed for full-length mRNA sequencing and does not incorporate UMIs, normalization for gene length is necessary. Therefore, we have to further normalize DCA output, dividing it by gene length. The resultant values, named DCA normalized counts, are similar as RPKM values but are denoised. All codes in the following analyses are in R programming language.

```
mrna_dca <- read.table('GSE114071_DCA.tsv', header=T, stringsAsFactors=F, row.names = NULL)
mrna_dca <- mrna_dca[,c(1:6,8:21)]
colnames(mrna_dca) <- colnames(mirna)
```

```
mrna_length <- read.table('GSE114071_length.tsv', header=T,
stringsAsFactors=F)

mrna_length <- mrna_length[,c(1:6,8:21)]

mrna_dca[,-1] <- mrna_dca[,-1] / mrna_length[mrna_length$Name
%in% mrna_dca$Name, -1]
```

As DCA removes genes that are unexpressed in all single cells, there are fewer genes (rows) in DCA output. We have to match genes in DCA output to genes in RPKM matrix.

```
mrna_dca <- mrna_dca[mrna_dca$Name %in% meta$gene,]

matched_rows <- which(meta$gene %in% mrna_dca$Name)
```

4.3 Perform Analysis with DCA Output

Like RPKM, we calculate the mean, SD and Residual SD of DCA normalized counts.

```
meta[matched_rows, 'mean_dca'] <- apply(mrna_dca[,index_-
cells], 1, mean)

meta[matched_rows, 'sd_dca'] <- apply(mrna_dca[,index_-
cells], 1, sd)

fit <- lm(log10(sd_dca) ~ log10(mean_dca), meta)

meta[matched_rows, 'rsd_dca'] <- fit$residuals
```

After calculating mean and Residual SD of DCA normalized counts, downstream analyses can be performed by simply replacing "mean_rpkm" and "rsd_rpkm" with "mean_dca" and "rsd_dca".

Compared with the results obtained from RPKM values, more significant differences in expression levels and noise among different groups is observed when using DCA normalized counts ($P = 7.48 \times 10^{-11}$ for expression levels, and $P = 5.49 \times 10^{-9}$ for expression noise, Kruskal–Wallis test). Besides, Fig. 3 shows that the distributions of expression levels and noise between any two groups also become more significantly different. These results further support that **miRNAs regulate their target mRNAs by reducing their expression levels and noise** (*see* **Note 10**).

5 Notes

1. A vignette demonstrating the complete analysis workflow for miRNA regulation in single cells, R codes, and data files are available at https://github.com/nshomron/sc-miReg.
2. Jupyter notebook is a web-based interactive development environment. It is part of Anaconda, and will be available after install Anaconda. For more information about using the R programming language in Jupyter notebook, refer to https://docs.anaconda.com/anaconda/navigator/tutorials/r-lang.
3. TensorFlow is an open-source platform for machine learning. It uses Python programming language to build machine learning models. DCA depends on Tensorflow to build machine learning models. TensorFlow supports both CPU and CUDA-enabled GPU. For new users and users working on small datasets, we suggest installing CPU version. For users working on large datasets, GPU version is recommended. As the dataset used in the analysis is very small, CPU version is enough. For larger datasets with thousands of cells, GPU version will greatly accelerate running speed.
4. There are many tools for miRNA target prediction. A detailed review summarized those tools and described most popular ones among them [24]. The authors concluded that TargetScan is the most robust one and the best choice in most cases. Therefore, we use it in the analysis.
5. RSEM is a popular software for accurately quantifying gene and transcript expression levels from RNA-Seq data. Here we use it to quantify gene expression in single cells. For more information about RSEM, refer to Li et al. [20] and https://github.com/deweylab/RSEM. There are many other methods with similar functions, and a benchmark analysis of these methods suggests that most of them have similar performance [25].
6. RPKM stands for reads per kilo base per million mapped reads. It is a commonly used unit of gene expression for RNA sequencing, because it considers library size and gene length for normalization. For read-based scRNA-seq protocols like Smart-seq [19], RPKM is commonly used as a unit of gene expression. However, many other scRNA-seq protocols like Drop-seq and 10× Genomics [26] incorporate UMIs to remove RCP amplification bias, which are also known as UMI-based protocols. In these protocols, gene expression is quantified using UMI count rather than normalized read count like RPKM. Unlike RPKM measuring relative expression, UMI measures the absolute molecule counts of mRNAs. As a result, UMI counts are not normalized for library size and gene length.

7. In this analysis, target mRNAs in four groups are not exclusive. For example, a mRNA can be targeted by both a highly expressed miRNA and a lowly expressed miRNA, and therefore it is present in more than one group. We notice that more than 90% mRNAs in HE, ME and LE groups are also in background group. This does not disturb our conclusion, but can be further processed to get more significant results. We tried removing mRNAs in lower expression groups that were also present in higher expression groups, like removing mRNAs in background group that were also present in HE, ME, and LE groups. We found that distributions of expression levels and noise between any two groups became more significantly different after doing so (see the vignette mentioned in **Note 1**).
8. There is another study which performed single-cell miRNA and mRNA sequencing in human embryonic stem cells [8]. Unlike the dataset used in the analysis workflow, miRNA and mRNA sequencing were performed in different single cells in this study. The dataset generated in this study is available at https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE81287. Similar results can be obtained using the analysis workflow demonstrated in this chapter. We suggest that readers try analyzing this dataset to verify the conclusion of this chapter.
9. The figures in this paper are created using ggplot2, a common graphics package used for R. For more information about ggplot2, refer to https://ggplot2.tidyverse.org/. Codes used to create these figures are available at https://github.com/nshomron/sc-miReg.
10. We propose several potential improvements for future studies concerning miRNAs' regulation on gene expression with scRNA-seq. These include (a) using UMI-based protocols like Drop-seq and 10× Genomics rather than read-based protocol in mRNA sequencing. This will greatly reduce amplification noise and enable more accurate quantification of mRNA expression [13]; (b) using spike-ins in mRNA sequencing and noise decomposition tools in analysis. Some tools are able to decompose total noise into technical noise and biological noise with spike-ins [12, 27]; (c) sequencing sufficient samples and sequencing each library to a nearly saturated depth such as one million reads for Smart-seq protocol and 0.1 million reads for 10× Genomics 3′ protocol. This will be helpful to accurately quantify cell-to-cell variability and gene expression noise.

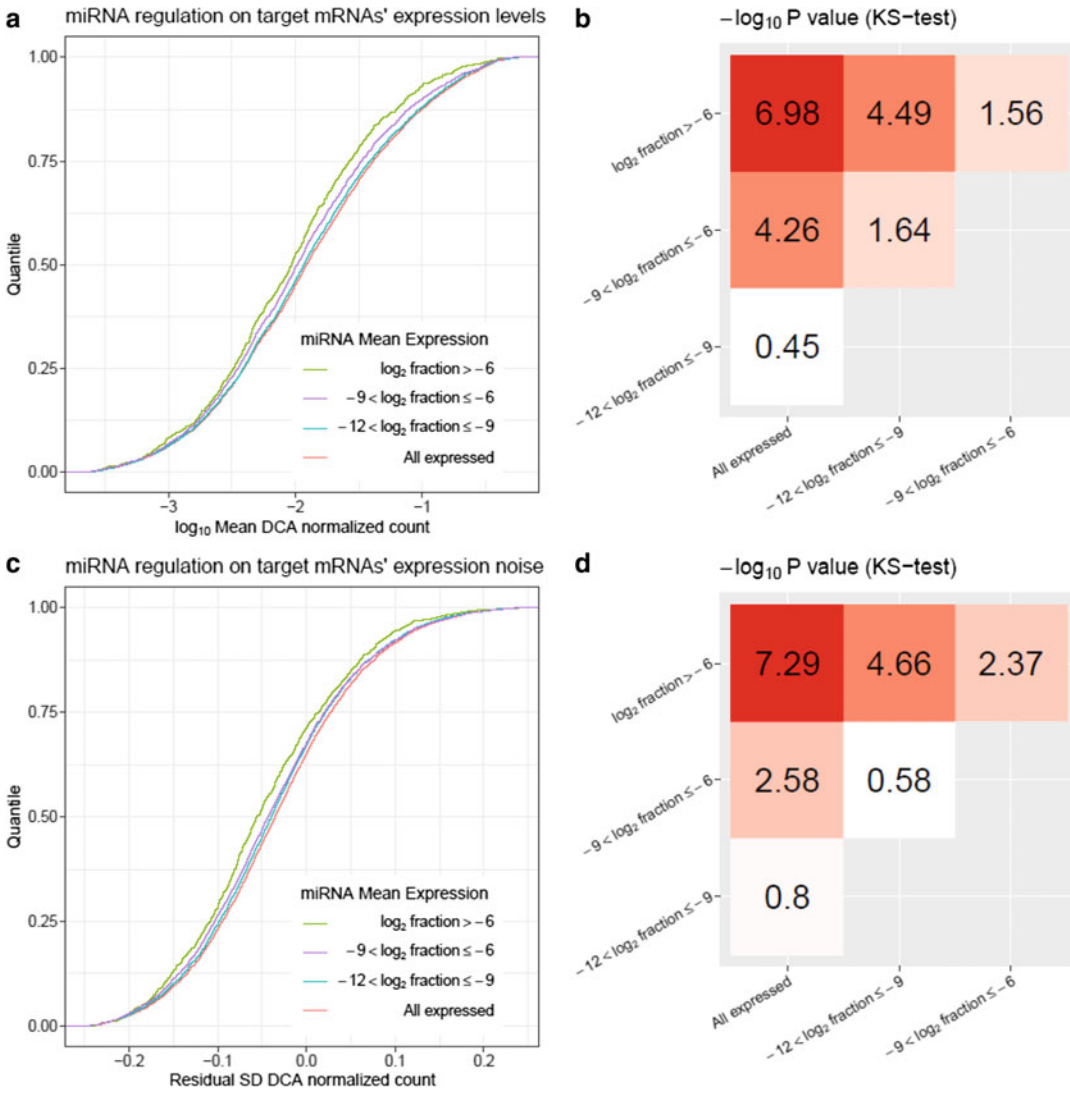


Fig. 3 (**a**) Empirical cumulative distribution functions (ECDFs) of target mRNAs' expression levels, estimated by mean DCA normalized counts. Target mRNAs are grouped by miRNA mean expression levels. (**b**) $-\log_{10}$ *P*-values between any two ECDFs in (**a**) calculated using KS-test. (**c**) ECDFs of target mRNAs' expression noise, estimated by Residual SD DCA normalized counts. (**d**) $-\log_{10}$ *P*-values between any two ECDFs in (**c**) calculated using KS-test

6 Conclusions

We have described the basic workflow to analyze miRNA regulation with scRNA-seq data. Accurately estimating the expression levels and noise of mRNAs is a vital step before measuring miRNAs' regulative effect. We introduced an approach to independently measure miRNA regulation on target mRNAs' expression levels and variability in expression which leads to noise. We observed

that highly expressed miRNAs tend to reduce the expression levels and noise of their target mRNAs. Besides, we also showed that denoising single-cell data could amplify biological signals and improve the power of the analysis. Finally, we would like to emphasize that although we focused on bioinformatics analysis in this chapter, improvements in experimental designs are also critical to generate high-quality data for further research in this field.

Acknowledgments

This work was supported by Tsinghua Xuetang Life Science Program.

References

1. Guo H, Ingolia NT, Weissman JS et al (2010) Mammalian microRNAs predominantly act to decrease target mRNA levels. Nature 466 (7308):835–840. https://doi.org/10.1038/nature09267
2. Baek D, Villen J, Shin C et al (2008) The impact of microRNAs on protein output. Nature 455(7209):64–71. https://doi.org/10.1038/nature07242
3. Selbach M, Schwanhausser B, Thierfelder N et al (2008) Widespread changes in protein synthesis induced by microRNAs. Nature 455 (7209):58–63. https://doi.org/10.1038/nature07228
4. Ebert MS, Sharp PA (2012) Roles for microRNAs in conferring robustness to biological processes. Cell 149(3):515–524. https://doi.org/10.1016/j.cell.2012.04.005
5. Schmiedel JM, Klemm SL, Zheng Y et al (2015) Gene expression. MicroRNA control of protein expression noise. Science 348 (6230):128–132. https://doi.org/10.1126/science.aaa1738
6. Hornstein E, Shomron N (2006) Canalization of development by microRNAs. Nat Genet 38 (Suppl):S20–S24. https://doi.org/10.1038/ng1803
7. Kolodziejczyk AA, Kim JK, Svensson V et al (2015) The technology and biology of single-cell RNA sequencing. Mol Cell 58 (4):610–620. https://doi.org/10.1016/j.molcel.2015.04.005
8. Faridani OR, Abdullayev I, Hagemann-Jensen M et al (2016) Single-cell sequencing of the small-RNA transcriptome. Nat Biotechnol 34 (12):1264–1266. https://doi.org/10.1038/nbt.3701
9. Buffa FM, Camps C, Winchester L et al (2011) microRNA-associated progression pathways and potential therapeutic targets identified by integrated mRNA and microRNA expression profiling in breast cancer. Cancer Res 71 (17):5635–5645. https://doi.org/10.1158/0008-5472.CAN-11-0489
10. Camps C, Saini HK, Mole DR et al (2014) Integrated analysis of microRNA and mRNA expression and association with HIF binding reveals the complexity of microRNA expression regulation under hypoxia. Mol Cancer 13:28. https://doi.org/10.1186/1476-4598-13-28
11. Hoffman Y, Pilpel Y (2015) Gene expression. MicroRNAs silence the noisy genome. Science 348(6230):41–42. https://doi.org/10.1126/science.aaa9841
12. Kim JK, Kolodziejczyk AA, Ilicic T et al (2015) Characterizing noise structure in single-cell RNA-seq distinguishes genuine from technical stochastic allelic expression. Nat Commun 6:8687. https://doi.org/10.1038/ncomms9687
13. Islam S, Zeisel A, Joost S et al (2014) Quantitative single-cell RNA-seq with unique molecular identifiers. Nat Methods 11(2):163–166. https://doi.org/10.1038/nmeth.2772
14. Jiang L, Schlesinger F, Davis CA et al (2011) Synthetic spike-in standards for RNA-seq experiments. Genome Res 21(9):1543–1551. https://doi.org/10.1101/gr.121095.111
15. Abadi M, Barham P, Chen J et al (2016) Tensorflow: a system for large-scale machine learning. In: 12th USENIX symposium on operating systems design and implementation (OSDI 16), 2016. pp 265–283
16. Eraslan G, Simon LM, Mircea M et al (2019) Single-cell RNA-seq denoising using a deep count autoencoder. Nat Commun 10(1):390. https://doi.org/10.1038/s41467-018-07931-2

17. Wang N, Zheng J, Chen Z et al (2019) Single-cell microRNA-mRNA co-sequencing reveals non-genetic heterogeneity and mechanisms of microRNA regulation. Nat Commun 10 (1):95. https://doi.org/10.1038/s41467-018-07981-6
18. Agarwal V, Bell GW, Nam JW et al (2015) Predicting effective microRNA target sites in mammalian mRNAs. Elife 4:e05005. https://doi.org/10.7554/eLife.05005
19. Picelli S, Bjorklund AK, Faridani OR et al (2013) Smart-seq2 for sensitive full-length transcriptome profiling in single cells. Nat Methods 10(11):1096–1098. https://doi.org/10.1038/nmeth.2639
20. Li B, Dewey CN (2011) RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome. BMC Bioinformatics 12:323. https://doi.org/10.1186/1471-2105-12-323
21. Huang M, Wang J, Torre E et al (2018) SAVER: gene expression recovery for single-cell RNA sequencing. Nat Methods 15 (7):539–542. https://doi.org/10.1038/s41592-018-0033-z
22. Li WV, Li JJ (2018) An accurate and robust imputation method scImpute for single-cell RNA-seq data. Nat Commun 9(1):997. https://doi.org/10.1038/s41467-018-03405-7
23. van Dijk D, Sharma R, Nainys J et al (2018) Recovering gene interactions from single-cell data using data diffusion. Cell 174 (3):716–729.e727. https://doi.org/10.1016/j.cell.2018.05.061
24. Riffo-Campos AL, Riquelme I, Brebi-Mieville P (2016) Tools for sequence-based miRNA target prediction: what to choose? Int J Mol Sci 17(12):1987. https://doi.org/10.3390/ijms17121987
25. Teng M, Love MI, Davis CA et al (2016) A benchmark for RNA-seq quantification pipelines. Genome Biol 17:74. https://doi.org/10.1186/s13059-016-0940-1
26. Macosko EZ, Basu A, Satija R et al (2015) Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. Cell 161(5):1202–1214. https://doi.org/10.1016/j.cell.2015.05.002
27. Vallejos CA, Marioni JC, Richardson S (2015) BASiCS: Bayesian analysis of single-cell sequencing data. PLoS Comput Biol 11(6): e1004333. https://doi.org/10.1371/journal.pcbi.1004333

Chapter 19

DNA Data Collection and Analysis in the Forensic Arena

Sydnie Grabell and Noam Shomron

Abstract

Recent scientific advancements in the field of genetics have fostered significant changes for the criminal justice system. Growing National DNA databases, public DNA databases, private direct-to-consumer (DTC) DNA testing companies, and improvements in next-generation sequencing (NGS) have resulted in effective methods for tracking down criminals and exonerating the innocent. While these recently discovered and profound techniques seem to provide benefits, their use in forensic detection has become subject to harsh legal opposition. Ultimately, should law enforcement be permitted to analyze DNA found at crime scenes and DNA that has accumulated in national, public, and private databases to aid in their investigations, or are individuals' privacy rights breached in the process?

Key words DNA, Data collection, Forensic DNA, NGS

1 Advancements in DNA Reading Ability

Over the course of the last century, advancements in science and technology have led to the drastic transformation in methods used to solve crime. While fingerprinting and blood splatter techniques were once major means of criminal identification, more recent advancements used to read and analyze specific regions of DNA have created a new niche in forensics—DNA profiling. The up-and-coming use of DNA in the forensic arena is effectively aiding in criminal investigations with the help of national DNA databases, the growing private direct-to-consumer industry, and public DNA databases as they provide reference samples for finding DNA matches. Together, they provide a far more consistent and trustworthy criminal identification system that has been used to reliably crack criminal cases that would not have been solved otherwise. These overall changes from older to newer technological advancements are shown in Fig. 1 with the introduction of more recent sequencing methods and databases aiding in effective forensic detection. Although major developments in DNA analysis have granted access to personalized medical information and solving

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6_19,

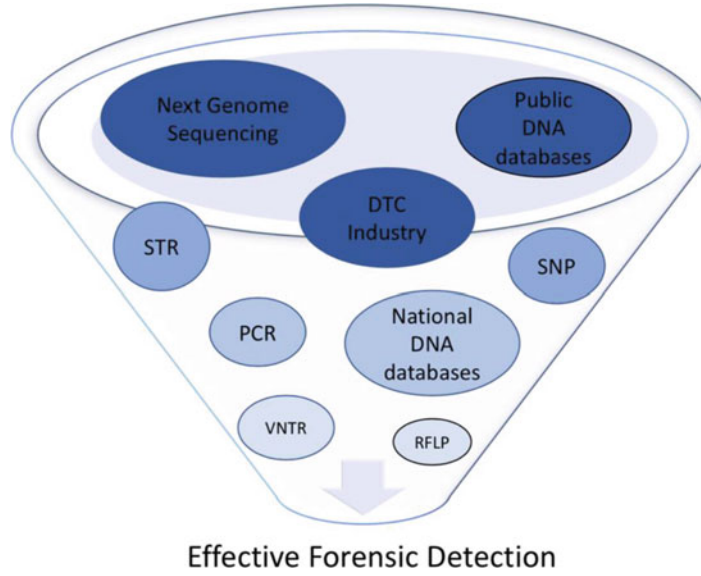


Fig. 1 A schematic representation of the DNA based tools used today for forensic detection

criminal cases through familial searching, its use in forensics and crime scene investigation is highly debated.

2 Human DNA Analysis

The human population shares 99.9% of DNA [1]. While most of our DNA is identical, the 0.1% that is unique to each person is responsible for our differences [1]. This 0.1% of DNA found in our genome is vital to forensics as it is used as a means of identification. The invention and development of scientific techniques used to read and analyze this small portion of DNA in order to generate DNA profiles has revolutionized the forensic realm.

DNA analysis of blood, bodily fluid, hair, or skin cells has transformed over time [2]. The shift from using variable number tandem repeats (VNTR) and restriction fragment length polymorphism (RFLP) to using the polymerase chain reaction (PCR), short tandem repeats (STR), and next generation sequencing (NGS, or deep sequencing) has resulted in more efficient and accurate DNA profiling techniques. VNTRs, repeating nucleotide sequences present within the 0.1% of nonidentical DNA, were the initial type of polymorphisms used to generate DNA profiles in the late 1900s [2]. To identify VNTRs within a DNA sample, restriction enzymes were employed to isolate polymorphisms by length [2]. These polymorphisms represent alleles used as a genetic marker for identification [2]. This process known as RFLP was a valuable procedure to match DNA found at crime scenes to individuals

responsible for crimes as VNTRs are different from one individual to the next [2]. Several limitations to this technique, however, inhibited its use in forensics. In order to analyze VNTRs by RFLP, an abundant nondegraded DNA sample was required [2]; however, in most instances, this was not possible, as most DNA samples acquired from crime sites are of low quantity and quality. Scientific breakthroughs including PCR, STRs, and NGS were therefore necessary, as they extended the effectiveness of DNA evidence in forensics.

3 Early Forensic Science

With the development of PCR in the mid-1980s, early forensic scientists had a faster and highly sensitive method for amplifying specific regions of a DNA sample found at a crime scene. PCR, through a series of heating and cooling events, uses DNA polymerase, primers, and the DNA taken from the crime scene to produce an abundant amount of DNA used for a more accurate DNA profile [3]. This amplified DNA is then added to four dideoxynucleotide triphosphates, four deoxynucleotide triphosphates, polymerase, and a primer to separate the DNA by size in a process known as capillary electrophoresis [3]. Sanger sequencing, a process involving PCR and capillary electrophoresis, was a key advancement for DNA data analysis introduced to forensics.

PCR was initially used to amplify VNTRs for identification; however, DNA contamination and the tendency for longer alleles to go undetected were major concerns and limitations [4]. The replacement of PCR-based VNTR technologies with PCR-based STR genotyping technologies in the early 2000s was a significant development in that it provided a DNA profile with fewer errors and therefore an overall more reliable identification system. This is due to the fact that STRs have a higher polymorphic sensitivity [2], which is shown by the repetition of only 2–6 base pairs in comparison to the repetition of 10–100 base pairs in VNTRs [2].

4 STRs and the Combined DNA Index System (CODIS)

Recent advancements presented by laboratories and PCR kit manufacturers have allowed numerous STR markers to be analyzed simultaneously, have incorporated STRs with even higher polymorphic sensitivity, have discovered additional STR loci, and have identified STRs specifically on the Y chromosome (Y-STRs) to pinpoint males responsible for crimes [2]. Originally 13 STR loci were used to create individual DNA profiles for identification, but this number is slowly increasing [5]. Today, the Combined DNA Index System (CODIS), and the US national DNA database,

creates DNA profiles based on 20 STR loci, known as the 20Plex [6], while the European Standard Set is based on 17 STR loci [7]. With progressing scientific research, a total of 73 STR markers are being studied for their potential involvement in DNA profiling [6]. The improvements in STR markers and technologies to read STRs with higher efficiency and sensitivity have paved the way for STR analysis to become the main mode behind the creation and analysis of DNA profiles [3]. With these transformations, DNA profiling is and will continue to be even more dependable as a means of finding exact or partial DNA matches between DNA found at crime scenes and DNA in various databases. Still, STRs in DNA profiling have been challenged as a means of identifying criminals due to the requirement of nondegraded and high-level DNA samples [2]. These restraints have inspired scientists to find other means of constructing genetic fingerprints. Single Nucleotide Polymorphisms (SNPs) have been investigated for DNA profiling because only a single nucleotide needs to be measured, as such valuable information can be extracted even from degraded DNA [3]. In addition, SNPs have a far fewer mutation rate and can be used more effectively in familial searching than STR [3]. However, as discussed below, generating DNA profiles based on SNPs could violate privacy rights as SNPs have the potential to reveal personal medical information.

5 NGS Used for Forensics

The most recent emerging development for DNA analysis in forensics is NGS. Compared to Sanger sequencing or first-generation sequencing, which is expensive (per DNA nucleotide) and can only read a small selection of bases at once, NGS is much more efficient and less expensive (per DNA nucleotide) [8]. This genomic improvement has provided the basis for a higher throughput of DNA, faster analysis, and a more specific DNA profiling system.

NGS is especially efficient in forensic investigations as it allows for quantitative and rapid analysis of mitochondrial DNA (mtDNA), which is often limited in its quantity. In addition, it provides higher throughput and sensitivity compared to that of Sanger sequencing. The development and authentication of amplicons covering control regions of the mitochondrial genome has proven to be highly sensitive and precise in both repeatability and reproducibility [9], as well as being able to handle degraded DNA. It has been experimentally proven to produce more than a 20-fold increase in sensitivity when compared to the Federal Bureau of Investigation (FBI) Laboratory's current Sanger sequencing–based protocols [9]. Lately, this technique has been used with samples of calcified tissue and hair [9]. While new NGS developments are being introduced to solve current crimes, they are also

used in previously unsolved cases. For example, the skeletal remains of the largest Second World War family massacre in Slovenia are being identified through autosomal and Y-STR typing [10]. As of 2020, a 21-plex STR panel has been validated, and hence was used in forensic STR genotyping [11]. Nevertheless, using NGS for forensic science purposes is expected to result in the use of varying sample preparation and sequencing methods, which will impact mtDNA haplotypes. In a recent study, researchers concluded that disparities in length heteroplasmy LHP displays are important to consider as mtDNA interpretation guidelines may need to be revisited with the implementation of NGS [12]. Overall, while these up-and-coming technological advancements seem to be beneficial, they must be further investigated prior to being used in the field of forensics.

6 National DNA Databases

Obtaining a DNA sample, which is then placed within a national DNA database, is now a required and legal booking procedure for countless offenses in many countries. In the USA, recent legal cases have expanded mandatory DNA sampling, which increases the amount of DNA available to law enforcement. *Maryland v. King* (2013) ruled that "taking and analyzing a cheek swab of the arrestee's DNA is, like fingerprinting and photographing, a legitimate police booking procedure … when officers make an arrest supported by probable cause … for a serious offense" [13]. The FBI can therefore obtain DNA from individuals who have been arrested, facing charges, or convicted of a crime as well as from individuals who are on probation, parole, or supervised release [5] and store it within CODIS, the United States National DNA database. However, in the USA, laws and regulations at the state level regarding whose DNA can be taken and stored within CODIS differs. Some states, for example, only obtain DNA samples from those who have committed any felony [14]. Meanwhile others are able to obtain DNA samples from individuals who have committed any enumerated misdemeanors and/or felonies [14]. Similarly, the Israel Police DNA Index System (IPDIS) accumulates and stores DNA samples from suspects as well as those accused and convicted of crimes [15]. The national database in the UK, however, differs from the USA, Israel, and most countries in that it deletes DNA profiles of suspects who are later deemed innocent. In the past, the national DNA database in England and Wales permanently stored DNA profiles of both guilty and innocent people, but the Protection of Freedoms Act passed in 2012 called for the elimination of DNA profiles of the innocent [16].

DNA stored within CODIS can only be used by law enforcement and the FBI [17]. This is analyzed using STR technology [1]. In criminal investigations, the STRs analyzed from the DNA of convicted criminals and arrestees is compared against STRs of the DNA found at specific crime scene locations in order to find DNA matches [1]. In doing so, law enforcement can compare allelic differences and determine suspects for more recent crimes based on past criminal history. National DNA databases do not rely on sequencing the whole genome to generate a DNA profile.

These growing national DNA databases have resulted in the exoneration of innocent people, identification of victim remains, and prosecution of criminals. In the USA alone, there have been 367 instances of DNA exonerations since 1989 [18]. DNA has also recently been used to identify victims of fatal events, such as 9/11 and wildfires [19]. The most significant use of DNA testing within national DNA databases has been to solve cold cases that are only solved through scientific advancements of DNA technology. In addition, entering DNA into a national DNA database for a first offense increases the chances of incrimination following future offenses.

7 Direct-to-Consumer (DTC) DNA Access

In addition to the benefits that national DNA databases offer for successful DNA profiling and matching, private DNA databases have also greatly expanded criminal investigation efforts. Private DNA sequencing and analysis of individual genomes has coined the term direct-to-consumer (DTC) genetics industry. Unlike DNA testing for national databases like CODIS, private databases for ancestry genetics use SNP genotyping to read millions of pieces of DNA [20]. In comparison to STRs analyzed in national DNA databases, SNPs can identify biological relationships, mutations, risks of disease, and other health information [5]. One of the most popular private DNA testing company, *23andMe*, provides information about ancestry, wellness, health predispositions, carrier status, and traits all in just a short 5-week waiting period. As technology continues to improve and as more competition in this field arises, the cost of sequencing our DNA continues to decline [2]. As such, a larger percentage of the population can purchase these services, which in turn results in a larger private DNA database that can be used to reference samples for criminal investigations.

Recently, certain private genetic companies have released their consumers' genetic information to the FBI to aid in search efforts. FamilyTreeDNA, another private DNA-testing company, willingly handed over millions of DNA profiles, granting law enforcement access to millions of peoples' DNA—access to the DNA from

customers who had their DNA tested and also access to DNA of their family members [21]. Not only can the FBI use DNA sequenced by private companies to find an exact match to crime scene DNA, but the FBI can build family trees to find the person of interest. Familial searching, building family trees from a close relative to identify a match, has been exceptionally beneficial for law enforcement's investigation efforts. Although FamilyTreeDNA voluntarily released DNA profiles to the FBI, other private genetic companies aim to protect their customers' health information and are legally obliged to do so.

Since most private DNA-testing companies do not release confidential health information to the FBI, they are valuable for solving crime in other ways. The introduction and advancement of private genomics companies within the DTC industry has led to the establishment of large public genealogy databases, which are legally subject to law enforcement evaluation [5]. As private DNA testing becomes cheaper with more advanced technology, public databases are expanding. Consumers can take their DNA sequenced by private companies and upload it to public databases, like GEDmatch, which has an expanded pool of individuals increasing the possibility of finding familial matches. DNA profile matching is more likely to occur in open-public databases as opposed to private due to their wide accessibility [22]. Since all private DNA-testing sites analyze SNPs in both coding and noncoding regions of DNA, the sequenced DNA from variable sites can be easily compared [23].

Law enforcement can upload crime scene DNA into public genealogy databases when no matches are found within national DNA databases. They can create a fake profile to find an exact match to a criminal that has uploaded his or her DNA to the public site or to a close relative of that criminal. The most famous success story of using a public genealogy site for crime scene investigation was with the incrimination of the Golden State Killer in 2018. Law enforcement uploaded crime scene DNA into GEDmatch from what they believed to be the man who committed a series of rapes and murders in the 1970s and 1980s [5]. A partial match to the DNA obtained at the crime scene was found within the system, and a family tree was constructed to find the likely suspect. Joseph James DeAngelo was identified by familial searching, as one of his relatives had uploaded DNA into the public database years earlier [5].

8 Privacy DNA Rights in the USA

Although private and public DNA databases have significant benefits for understanding personal health information, ancestry, and solving criminal cases effectively, the challenges and consequences they incite due to familial searching and violation of privacy rights established in the US Constitution may be too overbearing.

Although people who have been arrested or charged with committing a crime do not have access to the same privacy rights as other citizens [24], does law enforcement accessing private DNA testing information violate privacy rights of individuals who have agreed to certain Terms and Agreements, and could this indirectly violate the privacy of their relatives? Is law enforcement permitted to upload crime scene DNA to public genealogy sites, or does this violate privacy rights of suspects, criminals, and their family members? In the USA, these questions are frequently debated with the Fourth Amendment in mind.

Privacy rights in the USA are not explicitly stated within the Constitution. In *Griswold v. Connecticut*, the Court recognized that the Constitution has certain penumbras that construct the privacy rights of citizens [5]. These privacy rights protect the sampling, analysis, and release of personal information. The Court has upheld that DNA collection and analysis constitutes a search, and thus triggers Fourth Amendment rights [24]. However, improving technology poses certain challenges for the protection of privacy, as DNA testing has become more intrusive [5].

Certain privacy protections for individuals' genetic information has been established due to recent pioneering scientific advancements. The Genetic Information Nondiscrimination Act (GINA) passed in 2008 prevents employers and health insurance agencies from discriminating against individuals because of their genetic information, [25]; however, it does not protect against law enforcement searches. In 1996, the Health Insurance Portability and Accountability Act (HIPAA) established privacy and security standards for medical information. HIPAA laws were updated in 2013 to include the protection of genetic information [26]. Even with these legal advancements, there are many shortcomings for DNA protection within forensics.

In the USA, the Supreme Court has ruled that convicted offenders have no reasonable expectation of privacy in their DNA or DNA profile [24]. However, it is argued that the privacy rights of family members are overlooked. Once the FBI stores DNA profiles from an arrestee or a convicted criminal, their DNA can be used to both incriminate themselves and their relatives because DNA is hereditary [5]. In the past, the creation of DNA profiles used in criminal investigations did not reveal information about ancestry, health, or susceptibility to disease because law enforcement used STR technology [5]. While during that time, privacy rights seemed to be respected, as discussed earlier; however, the recent introduction of SNPs in human identification could violate privacy rights of those whose DNA is stored in national DNA databases [5]. This is because recent research has linked SNPs to health-related impacts [27], releasing more highly personal information. With advancing and constantly changing technologies that analyze DNA, privacy violations need to be reevaluated. As of today, the question of

whether or not generating DNA profiles using SNPs is a violation of privacy rights is undetermined because it is too new and has not yet been legally investigated [5].

9 Familial DNA Searching

Law enforcement can use national DNA databases for familial searching. Even if the privacy of suspects and convicted criminals is not violated by the collection and analysis of their DNA, the privacy of their relatives can be affected. Law enforcement searches within national databases to find a familial match to those who have already been convicted or arrested for criminal activity [1]. Family members of those whose DNA is stored in CODIS and other national DNA databases have not given up their privacy rights and have not consented to their DNA being subject to law enforcement's scrutiny [28]. When crime scene DNA is entered into CODIS, exact matches to offenders are highly unlikely [13]. The link to partial matches makes familial searching within CODIS so valuable [13]. In the USA, partial familial matching is only permitted on the local and state level and is prohibited on the national level [1]. However, as of 2018, only 12 states allow for the use of familial searching [29]. Each state has different rules and regulations set in place for when familial searching can be performed [29]. In 2002, the UK was the first to use familial DNA searching to find and incriminate a suspect [13], and since 2017, it has used familial searching to solve 210 cases [30]. Familial searching can be a beneficial last effort to solve criminal cases, as they lead to a much higher percentage of criminals being caught for the crimes that they commit; however, familial searching both within CODIS and other databases is still debated today as family members and other third parties have privacy rights protected under the Fourth Amendment [13]. Furthermore, familial searching in CODIS could reinforce racial discrimination, as a larger percentage of minority groups are represented within CODIS [13].

10 Private DNA-Testing Companies

When customers purchase DNA testing kits through private DNA-testing companies, they are agreeing to specific policies set forth by each company. In 23andMe's Privacy Highlights, they describe that they will not share any data with public databases, and they will not release data to insurance companies or law enforcement unless there is a search warrant [31]. AncestryDNA also declares that it will release DNA only if it is necessary to comply with legal warrants [32]. Under these privacy conditions, law enforcement does not have access to this information, unless

there is a search warrant issued for a reasonable cause or customers decide to personally release information on a public genealogy site. Therefore, these policies protect customers' privacy rights under the Fourth Amendment. Because the DTC industry tests DNA for intrusive genetic information and DNA tested by private companies is not protected by HIPAA laws, releasing data to law enforcement without a warrant could be an extreme violation of individual privacy.

11 Public DNA Databases

Public genealogy sites, most notably GEDmatch, have created serious issues regarding privacy of arrestees, convicted criminals, and innocent family members. Individuals can voluntarily release their DNA to the public; however, is law enforcement permitted to release personal information of suspected criminals to the public without their consent? When law enforcement does not find a match between the DNA collected from a crime scene and the DNA in national DNA databases, they are permitted to create a fake profile on a public database. Once the data is inputted, partial matches can be identified, and a family tree can be created to help identify potential suspects [5]. Because public genealogy sites are open, it is legal for law enforcement to search for partial familial matches without a warrant. Even though criminals have a reduced expectation of privacy, they are still protected from the release of private information under the Fourth Amendment [5]. While CODIS and other national databases are only available to law enforcement and are therefore kept out of the public eye, releasing DNA onto an open site broadcasts personal information to the public without the consent of the suspect [5].

Not only does releasing a suspect's DNA onto GEDmatch violate privacy rights in regard to the public having access to DNA without consent from a suspect, but law enforcement must also perform DNA testing that complies with that of the private DNA testing companies [5]. Unlike the STR testing completed for analysis of DNA in CODIS, which is nonintrusive and does not reveal ancestry or medical information, law enforcement must perform more invasive SNP testing that reveals personal medical information [5]. New research from Stanford University reveals that each STR marker is surrounded by SNPs that are typically inherited together with the STR [33] and therefore STR profiles from national DNA databases can be compared with SNP profiles generated from private DNA testing companies for partial familial matching [34]. Ultimately, STRs reveal more information than originally thought. In the past, people believed that law enforcement using DNA within national DNA databases for crime scene investigation purposes protected privacy rights. This is because

research proved that STRs did not release personal medical information, but these opinions may change if law enforcement employs new technologies and scientific information to find SNPs within crime scene DNA and compare it to DNA in public consumer databases. Furthermore, when law enforcement releases this anonymous crime scene information, they are also uploading the DNA of all the family related to that suspect's DNA without consent. As The New Republic explains, "when you upload your DNA, you are potentially becoming a genetic informant on the rest of your family" [35]. Is it legal for law enforcement to release this information if it could violate the privacy of their family as well?

When individuals choose to voluntarily upload their DNA to public sites, they understand that their personal medical information becomes available to the public, but are they aware that law enforcement can access their data and use it for criminal investigation purposes? In the Terms and Services listed on GEDmatch's website, GEDmatch notifies its users that law enforcement may search for exact or partial matches to crime scene DNA. As stated, "we may disclose your raw data, personal information, and/or genealogy data if it is necessary to comply with legal obligation . . . if you require absolute privacy and security, you agree that you will not provide personal information, raw data, or genealogy data to GEDmatch" [36]. As of May 2019, GEDmatch changed their Terms of Service by granting its users an option of opting in or opting out of law enforcement matching using their personal DNA information [37]. If they did not choose to opt in, they were automatically opted out. This significantly increases the privacy of public DNA database users, as they can privatize their genetic information from law enforcement inspection, and it critically reduces the amount of DNA available for law enforcement to solve criminal investigations and DNA profile matching.

12 Should Law Enforcement Have Access to all Genetic Data?

With the strengths of DNA in the forensics arena, however, come weaknesses to the system as a whole. With the potential violation of the Fourth Amendment to the US Constitution and HIPPA, the public is hesitant to rely fully on new genetic advancements for resolving criminal mysteries. In a study done using Amazon Mechanical Turk, 62% of people agreed that law enforcement should be able to use public genetic websites, such as GEDmatch to disclose genetic information because it helps to solve crime. However, 80% of people said that genetic information stored within public databases should only be used for violent crimes. Does this mean that the public feels comfortable with law enforcement having access to all genetic data at will, or should they have limited access depending on the nature of the crime? Moreover, does the

public want to ensure that criminals rightfully serve time or do they want to protect their personal life? Although privacy is a highly debated issue in today's society, according to an article written by The Conversation, CeCe Moore held that people are definitely willing to allow law enforcement to use their genetic information without considering the privacy of their family members [38]. As Moore says, people are happy to help law enforcement solve criminal investigations because they want dangerous people off the streets. This has been shown in a study performed in PLOS Biology, which found that "79% of 1,578 responses ... support police searches of GEDmatch" [38].

It may be easy for us to say that law enforcement should have access to DNA within private and public databases because locking criminals away is important. But how would we feel if we were placed in a situation that could incriminate us and/or our family members? Possibly, even for a crime we do not know we/they have committed? Ultimately, the decision of whether or not to read your DNA is perplexing. Although it can be beneficial to use genetic information to understand important medical information, obtaining this knowledge could be used against you and your family in crime scene investigation. Do not take this decision lightly, for "once you read it, you can't go back" [39].

References

1. Hodge SD Jr (2018) Current Controversies in the Use of DNA in Forensic Investigations. Univ Baltimore Law Rev 48(1):39–66; HeinOnline
2. Imam J et al (2018) Chapter 1: DNA fingerprinting: discovery, advancements, and milestones. In: DNA fingerprinting: advancements and future endeavors. Springer, Singapore, pp 3–24. https://doi.org/10.1007/978-981-13-1583-1_1
3. McCord BR, Butler JM (2001) The application of capillary electrophoresis in the analysis of PCR products used in forensic DNA typing. In: Clinical and forensic applications of capillary electrophoresis. Humana Press, Totowa, NJ, pp 261–284. https://doi.org/10.1007/978-1-59259-120-6_14
4. Decorte R, Cassiman J-J (1993) Forensic medicine and the polymerase chain reaction technique. J Clin Forensic Med 30:625–632. https://doi.org/10.1016/1353-1131(94)90066-3
5. Guest C (2019) DNA and law enforcement: how the use of open source DNA databases violates privacy rights. Am Univ Law Rev 68 (3):1015–1052; HeinOnline
6. Novroski NMM et al (2018) Expanding beyond the current core STR loci: an exploration of 73 STR markers with increased diversity for enhanced DNA mixture deconvolution. Forensic Sci Int Genet 38:121–129. https://doi.org/10.1016/j.fsigen.2018.10.013
7. Rahman AS et al (2014) Evaluation of five new European Standard Set (ESS) STR Loci in a Bangladeshi Population Sample. Mal J Forensic Sci 5(2):13–17. https://doi.org/10.1016/j.fsigen.2011.04.006
8. Pinxten W, Howard HC (2014) Ethical issues raised by whole genome sequencing. Best Pract Res Clin Gastroenterol 28:269–279. https://doi.org/10.1016/j.bpg.2014.02.004
9. Brandhagen MD et al (2020) Validation of NGS for mitochondrial DNA casework at the FBI laboratory. Forensic Sci Int Genet 44:102151
10. Pajnič IZ et al (2020) Identifying victims of the largest Second World War family massacre in Slovenia. Forensic Sci Int 306:110056
11. Silvery J et al (2020) Developmental validation of the MonSTR identity panel, a forensic STR multiplex assay for massively parallel sequencing. Forensic Sci Int Genet 46:102236

12. Sturk-Andreaggi K et al (2020) Impact of the sequencing method on the detection and interpretation of mitochondrial DNA length heteroplasmy. Forensic Sci Int Genet 44:102205
13. Joh EE (2013) Maryland v. King: policing and genetic privacy. Ohio State J Crim Law 19:281–294. https://doi.org/10.1017/9781108290364.007
14. DNA arrestee laws. In: National conference for State legislatures, 2013. http://www.ncsl.org/Documents/cj/ArresteeDNALaws.pdf
15. Zamir A et al (2012) The Israel DNA database—the establishment of a rapid, semi-automated analysis system. Forensic Sci Int Genet 6:286–289. https://doi.org/10.1016/j.fsigen.2011.06.003
16. What is the UK National DNA database? Yourgenome, The Public Engagement Team at the Wellcome Genome Campus. 11 Dec 2014. https://www.yourgenome.org/facts/what-is-the-uk-national-dna-database
17. Pattock A (2011) It's all relative: familial DNA testing and the fourth amendment. Minnesota J Law Sci Technol 12(2):851–876
18. DNA exonerations in the United States. Innocence Project. https://www.innocenceproject.org/dna-exonerations-in-the-united-states/
19. Zimmer K. Rapid DNA analysis steps in to identify remains of wildfire victims. The Scientist Magazine, 30 Nov 2018. https://www.the-scientist.com/news-opinion/rapid-dna-analysis-steps-in-to-identify-remains-of-wildfire-victims-65156
20. Saey TH. What consumer DNA data can and can't tell you about disease risk. Science News, 3 June 2018. www.sciencenews.org/article/health-dna-genetic-testing-disease
21. Brown KV. Major DNA testing company sharing genetic data with the FBI. Bloomberg.com, Bloomberg, 1 Feb 2019. https://www.bloomberg.com/news/articles/2019-02-01/major-dna-testing-company-is-sharing-genetic-data-with-the-fbi
22. Murphy H. Genealogists turn to cousins' DNA and family trees to crack five more cold cases. The New York Times, The New York Times, 27 June 2018. https://www.nytimes.com/2018/06/27/science/dna-family-trees-cold-cases.html
23. Zhang S. How a tiny website became the Police's go-to genealogy database. The Atlantic, Atlantic Media Company, 1 June 2018. https://www.theatlantic.com/science/archive/2018/06/gedmatch-police-genealogy-database/561695/
24. DNA Databanking: selected fourth amendment issues and analysis. Congressional Research Service, 16 Aug 2011. https://fas.org/sgp/crs/misc/R41847.pdf
25. Genetic discrimination. Genome.gov, National Human Genome Research Institute. https://www.genome.gov/about-genomics/policy-issues/Genetic-Discrimination
26. Genetic information privacy. Electronic Frontier Foundation. https://www.eff.org/issues/genetic-information-privacy
27. What are single nucleotide polymorphisms (SNPs)? - Genetics Home Reference - NIH. U.S. National Library of Medicine, National Institutes of Health, 9 July 2019. https://ghr.nlm.nih.gov/primer/genomicresearch/snp
28. Debus-Sherrill S, Field MB (2019) Familial DNA searching- an emerging forensic investigative tool. Sci Justice 59:20–28. https://doi.org/10.1016/j.scijus.2018.07.006
29. Rainey J. Familial DNA puts elusive killers behind bars. But only 12 States use it. NBCNews.com, NBCUniversal News Group, 28 Apr 2018. https://www.nbcnews.com/news/us-news/familial-dna-puts-elusive-killers-behind-bars-only-12-states-n869711
30. Murray A et al (2017) Familial DNA testing: current practices and familial DNA testing: current practices and recommendations for implementation. Investig Sci J 9
31. Privacy highlights. 23andMe, 2019. https://www.23andme.com/about/privacy/
32. Ancestry guide for law enforcement. Ancestry. https://www.ancestry.com/cs/legal/lawenforcement
33. Edge MD (2017) Linkage disequilibrium connects genetic records of relatives typed with disjoint genomic marker sets. Proc Natl Acad Sci U S A 114:5671–5676. https://doi.org/10.1101/345322
34. Callaway E. Supercharged crime-scene DNA analysis sparks privacy concerns. Nature News, Nature Publishing Group, 11 Oct 2018. https://www.nature.com/articles/d41586-018-06997-8
35. Ford M. How the Supreme Court could rewrite the rules for DNA searches. The New Republic, 30 Apr 2018. https://newrepublic.com/article/148170/supreme-court-rewrite-rules-dna-searches
36. GEDmatch.Com terms of service and privacy policy. GEDmatch. 9 Dec 2019. https://www.gedmatch.com/tos.htm

37. Ram N. The genealogy site that helped catch the Golden State killer is grappling with privacy. Slate Magazine, Slate, 29 May 2019. https://slate.com/technology/2019/05/gedmatch-dna-privacy-update-law-enforcement-genetic-geneology-searches.html
38. Lageson SE. Privacy concerns don't stop people from putting their DNA on the internet to help solve crimes. The Conversation, 11 Dec 2019. http://theconversation.com/privacy-concerns-dont-stop-people-from-putting-their-dna-on-the-internet-to-help-solve-crimes-118091
39. Shomron N. TEDxTelAvivUniversity, 22 Feb 2016. https://www.youtube.com/watch?v=NgVwPj54TEo

INDEX

Noam Shomron (ed.), *Deep Sequencing Data Analysis*, Methods in Molecular Biology, vol. 2243,
https://doi.org/10.1007/978-1-0716-1103-6, © Springer Science+Business Media, LLC, part of Springer Nature 2021

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

Y