Sequence Analysis

Andrey D Prjibelski, Anton I Korobeynikov, and Alla L Lapidus, St. Petersburg State University, St Petersburg, Russia

Glossary

Adjusted p-value The lowest familywise significance level, at which a certain comparison will be considered as statistically significant as part of the multiple comparison testing.
Alignment score A measure that reflects the level of similarity between sequences.
Contig Long genomic fragment, usually generated by the genome assembly software.
Copy number variation (CNV) An alteration between the number of the same genomic fragment being repeated in the genome within the population.
de Bruijn graph An approach used for *de novo* sequence assembly from short reads.
De novo genome assembly The process of constructing the genome sequence directly from the sequencing reads without any supplementary information.
Differential expression analysis A process of studying changes in expression of the genes between samples, usually under different conditions.
Duplication A rearrangement event in which a genomic fragment is copied into the same chromosome.
Edit distance The minimum number of operations (insertions, deletions or substitutions) needed to turn one sequence into another.
FASTA format A text-based file format for storing biological sequences.
GC content The percentage of cytosine and guanine nucleotides relative to the total length of nucleic sequence.
Genome rearrangement A large genomic mutation, which alternates the whole chromosomes or their long fragments.
Graph simplification The process of removing erroneous edges from the graph (usually referred to de Bruijn graph).
Hamming distance The number of positions in two strings of the same length, in which the symbols do not match.
Hybrid assembly An approach for *de novo genome assembly* using multiple different sequencing technologies simultaneously.
Indel Insertion or deletion.
Inversion A rearrangement event in which a genomic fragment is reversed within the chromosome.
k-mer A sequence of length k.
Metagenomics A field of genomics, which studies genomes of the entire bacterial communities (or other samples containing different species).
Misassembly An error in the assembly when two distant genomic fragments are joined into a single contigs/scaffold.
N50 The maximum length X for which the collection of all contigs of length $>=$ X covers at least 50% of the assembly.
Overlap-Layout-Consensus approach (OLC) An *de novo* genome assembly method used for long sequencing reads.
Paired-end reads A couple of reads sequenced from the different ends of the same genomic fragment (typically from different strands); also can be referred to as left and right reads, or *mate reads*.
Phylogenetic tree A tree that demonstrates evolutionary relationships between selected biological species.
Read alignment (read mapping) The process of aligning sequencing reads to a longer sequence (typically reference genome) to detect the position, from which the read was originally sequences.
Read mapping See *read alignment*.
Repeat resolution The process resolving ambiguities during the assembly, which are caused by the genomic repeats.
SAM/BAM formats Formats for storing read alignment data. SAM is a text-based format and BAM is a compressed binary format.
Scaffold An ordered set of contigs, typically separated by regions with unknown bases (N).
Scaffolding The process of ordering *contigs* into *scaffolds*.
Sequence alignment The process of detecting similarities between biological sequences.
Single nucleotide polymorphism (SNP) A variation in a single position in the genome that appears in appreciable part of the population.
Strand-specific RNA-Seq A method for sequencing RNA molecules, from which it is possible to deduce the strand on which the gene is located.
String graph An approach used for *de novo* sequence assembly from short reads.
Synteny block A set of consecutive genes on the chromosome, which are typically inherited together and preserved by rearrangement events.
Translocation A rearrangement event in which a genomic fragment is exchanged with a fragment from another chromosome.
Universal single copy-ortholog A gene, which does not have paralogs and is typical for a given set of organisms.
VCF/BCF formats Formats for storing variations and mutations. VCF is a text-based format and BCF is a compressed binary format.

Introduction to Sequence Analysis

Sequence analysis is a term that comprehensively represents computational analysis of a DNA, RNA or peptide sequence, to extract knowledge about its properties, biological function, structure and evolution. To carry out sequence analysis efficiently, it is important to first understand the source of the data, i.e., the different experimental methods used for determining the biological

 doi:10.1016/B978-0-12-809633-8.20106-4

sequence. We then need to follow analytical strategies, depending on whether the sequence is genomic, transcriptomic or proteomic. Databases currently warehousing the enormous data on these biomolecules will need to be first checked for the presence of similar sequences, which might direct experimental assays for functional investigations. Software tools and web services are often used for carrying of the bioinformatics analysis. After analysis of DNA, RNA and protein sequences, it is important to understand how they are connected by protein to genome mapping. The small organic molecules or metabolites that are essential for organisms to live and grow also need to be studied in the context of their interaction with genes and proteins, *via* metabolic pathways. This article aims to provide an overview of sequence analysis, at the DNA, RNA and proteins levels, with metabolic pathways describing their interplay.

Nucleic Acid Sequencing

DNA Sequencing Technologies

DNA sequencing is the process of determining the order of nucleotide bases of molecules of DNA. If the DNA of the whole genome of an organism is used as the DNA of interest, then the process is referred to as genomic sequencing. This process is currently widely used both in fundamental biological research and in many applications. Information about the primary DNA sequence is a crucial piece of data used in such areas of research as medical studies, gene therapy, drug development, evolutionary studies, agriculture, improvement of nutritional quality, biotechnology, forensic and many other.

First-generation sequencing

The era of DNA sequencing began about 30 years ago, when two methods of primary DNA structure determination appeared almost simultaneously. The one developed by A. Maxam and W. Gilbert at Harvard University was based on the use of chemical modifications of DNA. The other method was published by F. Sanger and A. Coulson from Cambridge and is called the chain-termination method. These methods were named 1st generation sequencing (Heather and Chain, 2016).

And although over time, both the Gilbert and the Sanger DNA sequencing methods have gone through a number of significant changes (one of which was the automation of the Sanger DNA method that assured its wider application), first generation sequencing methods continued to remain labor intensive, had a low throughput speed and were very expensive. To overcome these limitations and in response to the National Human Genome Research Institute (NGHRI) call to lower the cost of sequencing of an individual human genome to $1000, with an intermediate goal of reducing the cost of sequencing a mammalian-sized genome to $100,000, next generation sequencing technologies (NGS) were rapidly developed and began to be used in pyrosequencing, base-by-base sequencing by synthesis (SBS), sequencing by ligation, nanopore sequencing, and single-molecule sequencing by synthesis in real time strategies.

These newest sequencing technologies that are currently at different stages of development and implementation can be split into two groups: Second generation sequencing and the third generation approaches.

Second-generation sequencing

Compared to the traditional Sanger sequencing, the greatest advantage of the second generation sequencing is a very high level of parallelization: up to 10^7 reactions in one experiment. In addition, the minuscule volume of the individual wells (picoliters), within which the sequencing reactions are run and the high data density (10,000 times higher, than in case of the latest microelectrophoresis-based capillary instruments) significantly decrease the volume of reagents needed to perform the reactions. Another positive aspect of the second generation methods is the ability to circumvent DNA cloning and propagation in *Escherichia coli* cells, thus avoiding the problems of biased genome coverage due to the presence of genomic areas, which are hard or even impossible to clone in E. coli (so called unclonable areas). The downside of these technologies, as compared with the Sanger approach, however, is the length of the produced reads.

The 454 Genome Sequencer 20 System instrument (the first commercial NGS platform) produced reads of about 110 bp. The latest GS FLX Titanium chemistry (Margulies *et al.*, 2005) has managed to increase read length up to 1 kb (454 Life Sciences was shut down in 2013 and 454 sequencing instruments are not supported any longer). Meanwhile, technologies using SBS and sequencing by ligation (Applied Biosystem SOLiD; polony-based approaches) produced reads of only 25–50 bp long (Shendure *et al.*, 2005; Valouev *et al.*, 2008). This meant that although 454 sequencing has been successfully applied to de novo genome sequencing resulting in a gapped assembly with an error rate of about 1/3000–1/5000 bp, the ultra-short reads produced by Solexa technology appeared to be of use for resequencing purposes only, in this case a high-quality reference sequence would be present for alignment. The use of 25–50 bp reads for de novo genome sequencing appeared to be very problematic.

After being acquired by Illumina, the Solexa technology was renamed to Illumina technology. This approach uses the SBS sequencing to generate massively parallel genomic data and detect single bases as they are incorporated into growing DNA strands. Read lengths vary from 100 bp to 300 bp depending on the sequencing instrument used (MiSeq, NextSeq 500, different HiSeq). The low error rate of Illumina reads ($\leq$0.1%) makes this sequencing technology very attractive for a number of applications including de novo genome assembly.

Ion Torrent is yet another NGS approach (Rothberg *et al.*, 2011). It uses the semiconductor sequencing technology of SBS. The Ion PGM sequencer detects changes in pH when a nucleotide is incorporated into the DNA molecules and a proton is released.

This technology produces reads of 400 bp long with about 1% error rate, allowing single nucleotide polymorphism (SNP) detection as well as whole genome assembly.

The second Ion Torrent instrument released in 2012, the Ion Proton, produces shorter reads (200 bp), but provides 10 time larger (~10 Gb) output than its predecessor. The read length provided by recently developed Ion S5 and Ion S5 XL Systems (www.thermofisher.com) is ranging from 200 bp to 600 bp depending on the chip type and the output can be as high as 10–15 Gb.

Third-generation sequencing

Among the numerous methods on the drawing board, special attention should be given to real-time sequencing of DNA molecules. Such methods use very small amounts of genomic DNA and require supersensitive detection methods. The success of these methods wholly depends on the quality of the genomic DNA used in the experiment: if the DNA is damaged while being isolated, the resulting sequence will be corrupt. In theory, the read length produced by this type of technology should be equal to that of a full genome.

*P*acBio (SMRT) sequencing is one of the first real-time single-molecule sequencing techniques capable of generating long reads (10–15 Kb) without requiring any amplification steps. Additionally, its unique ability to distinguish methylated bases from normal nucleotides (Levene *et al.*, 2003) opens up the possibility of studying the epigenetic potential of different organisms. Read error rate is high, but errors are random and thus can be corrected via an increase in coverage. Long reads play a crucial role in the gap closing step of the whole genome assembly and PacBio gave genome finishing, a step that for several years was set aside as time-consuming and not cost efficient, a much needed boost.

The UK company Oxford Nanopore Technologies developed a next generation sequencing approach based on nanopores. The MinION system allows to produce more than 3 Gbp of data from a single run. The resulting reads are of a variable length and can be up to 900,000 bases long. The current error rate of nanopore reads is 5%–15%. It was observed that this technology produces systematic sequencing errors, which end up being more challenging to correct bioinformatically than random ones.

The list of companies and university-based groups of researchers working on improving sequencing technology goes on and on. Some of them have already managed to achieve commercial success, others are still only working on approaching this goal and many have either already failed to do so or will end up failing eventually (Lapidus, 2015). In most cases a combination of different sequencing approaches allows to produce a high-quality sequence fairly quickly and at very low cost.

Rna-Seq

RNA molecules play a significant role in all living cells. Messenger RNA (mRNA) for example is responsible for translating the genetic information stored in DNA into proteins composed of different combinations of 20 amino acids, each of which has its own type of transfer RNA (tRNA) that binds and transports them to the growing polypeptide chain when needed. Ribosomal RNA (rRNA) is an integral part of both large and small subunits of ribosomes – A complex molecule responsible for protein synthesis in cells.

Transcriptomics allows to study the nature and amount of each of the RNA molecules in a given cell under a specific condition and at a given moment. The RNA sequencing (RNA-seq) approach uses next-generation sequencing (NGS) to perform direct sequencing of cDNA fragments (Hrdlickova *et al.*, 2017).

RNA-seq can detect: (i) Gene expression level; (ii) transcript abundance; (iii) alternative splicing that causes transcriptional diversity in eukaryotes; (iv) different isoform usage; (v) single nucleotide variations; (vi) fusion genes; (vii) post-transcriptional modifications. It can identify genes, exons, splicing events, ncRNAs, novel genes or transcripts.

A typical RNA-Seq experiment consists of RNA isolation and purification, assessment of the RNA purity and yield, enrichment of target RNAs (through polyA selection, size selection, or removal of non-target sequences through hybridization to probes), fragmentation of RNA, amplification, sequencing, followed by data analysis that in its turn includes quality control, read alignment to a reference genome or known transcripts, transcript quantification and *differential expression analysis* (Kukurba, 2015).

Most of the methods used for RNA-seq data analysis rely on well-annotated reference genomes and on accurate gene models of model organisms. The *de novo* transcriptome assembly approach helps to reconstruct the sequence of the expressed transcripts and is used when reference genomes are not available.

Sequencing Data Formats

The simplest and one of the most common formats for storing nucleic and amino acid sequences is the *FASTA format* (see "Relevant Websites section"). Typically FASTA files have ".fasta" or ".fa" extensions, but other extensions, such as ".fna" for nucleic sequences and ".faa" for proteins are also sometimes used. A single entry in a FASTA file contains the sequence name (sequence ID) and the sequence itself. Sequence names may also contain some meta information, such as for example sequence length. When the sequence is long, it is stored in multiple lines for added legibility. FASTA files can contain an unlimited number of sequence entries.

Sequencing reads can also be stored in the *FASTQ format* (see "Relevant Websites section") that contains information about reads quality in addition to read sequence and read ID. In the process of sequencing, the sequencer can estimate the probability of each nucleotide being detected incorrectly. The decimal logarithm of the error probability is then rounded to the nearest integer,

added to a fixed offset and written to the FASTQ file as the corresponding ASCII character. However, there are several different formats for storing quality values, which vary by offset values and the way the error probability is converted to an integer number. Read ID in FASTQ file may contain additional information about the reads. For example, Illumina sequencer stores flowcell lane number, tile number, coordinates of the cluster and the information about read pairing (if reads are *paired*).

There are two possible ways to store *paired-end reads* in FASTQ files. The most common method stores left and right reads in the two separate files. The order of the reads in both files should be the same, i.e., first read from the file with left reads is a mate for the first read from the file with right reads. Paired reads can also be saved into a single file (called interlaced FASTQ file), in which left and right reads are placed one after another.

Since both FASTA and FASTQ formats are text formats, they typically require a huge amount of disk space. To save space, FASTQ files are usually either compressed using standard archiver software, or converted to SRA or unmapped BAM binary formats.

Beside FASTA/FASTQ, there exists a large variety of file formats suitable for storing supplementary information, such as sequence alignments, genomic variations and genome annotation. Some of these formats will be described further throughout the article.

Analyzing Genomic Sequences

Sequence Alignment

Classic alignment algorithms

Sequence alignment is the process of comparing and detecting similarities between biological sequences. What "similarities" are being detected will depend on the goals of the particular alignment process. Sequence alignment appears to be extremely useful in a number of bioinformatics applications.

For example, the simplest way to compare two sequences of the same length is to calculate the number of matching symbols. The value that measures the degree of sequence similarity is called the *alignment score* of two sequences. The opposite value, corresponding to the level of dissimilarity between sequences, is usually referred to as the *distance* between sequences. The number of non-matching characters is called the *Hamming distance*. **Fig. 1** shows an example of two sequences with Hamming distance (Bookstein *et al.*, 2002) equal to 3.

It is, however, worth noting that comparing sequence characters position by position as described above can barely be referred to as alignment process, since it does not take into account such typical biological events as deletions and insertions. The classical notion of sequence alignment includes calculating the so called *edit distance*, which generally corresponds to the minimal number of substitution, insertions and deletions needed to turn one sequence into another. **Fig. 2** demonstrates an example of two sequences with edit distance equal to 3.

The problems of computing edit distance and various types of sequence alignment have exact solutions, e.g., (Smith and Waterman, 1981) and (Needleman and Wunsch, 1970) algorithms. Since these algorithms were initially developed for protein-protein alignment and later adapter for DNA sequence alignment, they are described in the section 'Protein-protein alignment'. In most real-life cases, however, these algorithms appear to be impractical for DNA alignment due their running time and memory requirements.

BLAST and similar database search methods

To overcome the limitations of the classic alignment algorithms, which are usually used for rather short sequences, new methods and heuristics were developed. By heuristic algorithm we imply a non-exact solution, which works faster than exact algorithms (sometimes much faster) and provides biologically meaningful results.

Undoubtedly, the most popular sequence alignment tool in bioinformatics is the BLAST algorithm (Altschul *et al.*, 1990) and its variations, such as BLASTN (Acland *et al.*, 2014) and MegaBLAST (Morgulis *et al.*, 2008). The paper that describes the first version of the BLAST algorithm now has more than 60 thousand citations and the count keeps going up. Various modifications of the BLAST algorithms are used for aligning amino acid sequences to nucleic sequences (tblastn) and vice versa (blastx). BLAST is

Fig. 1 Example of two sequences with Hamming distances equal to 3.

Fig. 2 Example of two sequences with edit distances equal to 3.

widely used on the NCBI web site for searching sequences in various databases, such as 16s rRNA sequences and the nucleotide collection (all assembled genomes).

Beside the BLAST family algorithms, multiple alignment methods for various goals were developed during the past 20 years. Depending on the biological problem and alignment objective, different methods employ different algorithms and heuristic ideas, such as suffix trees and suffix arrays, k-mer indexing, seed-and-extend approaches and Hidden Markov Models. More information about specific sequence alignment algorithms can be found in Section "De Novo Transcriptome Assembly" (protein sequence alignment) and Section "Sequencing Data Formats" (mapping sequencing reads to the reference genome).

Comparative genomics

Nucleic sequence alignment algorithms are widely used in comparative genomics and phylogenetic studies. Comparative genomics studies similarities between two or more genomes at the level of large *rearrangement* events, such as *inversions*, *duplications*, *translocations*, large insertions and deletions. Since the comparative genomics usually does not take into account small structural variations and *single nucleotide polymorphisms* (SNPs), it requires a specific kind of alignment software. These methods typically detect *synteny blocks* – long sequences shared between genomes being compared. Indeed, those sequences may have differences at the nucleotide level, but are still highly similar overall. The genomes are then represented as a sequence of synteny blocks and rearrangements are detected (Hannenhalli and Pevzner, 1999).

Phylogenetic studies use various multiple sequence alignment methods to detect the level of sequence dissimilarity. The distance between compared sequences is used to construct *phylogenetic trees*, in which the length of the branches typically correspond to the distance between analyzed sequences. To construct a biologically meaningful and realistic tree, various clustering methods can be used as well as different sequences may be provided as input (Felsenstein, 1981; Kumar *et al.*, 1994). Phylogenetic studies can be done using whole genomes and rearrangement events, genes and proteins sequences, or even SNPs for closely related organisms.

Preprocessing Sequencing Data

Quality control of sequencing data

Regardless of what technology, protocol or sample was used to generate sequencing data, quality control remains an integral part of every experiment. When performed correctly during the early stages of a project, quality control helps save time and thus, money. There have been many cases when false conclusions were made due to the poor quality of initial data, and, as a known saying states, "garbage in – garbage out", meaning that great results do not come from low-quality data.

FastQC is one of the most popular tools for basic quality control of different kinds of sequencing data (Andrews, 2010). FastQC does not require any additional data, such as a reference genome, and the QC is performed based on just a FASTQ file with sequences and corresponding quality values. Below we will take a look at several important statistics produced by FastQC, how to interpret them and what the differences between high- and low- quality data are.

The first and one of the most important plots is the per base sequence quality plot. It demonstrates the average quality value amongst the reads. Illumina reads typically have lower quality and a higher number of sequencing errors towards the end of the read. **Fig. 3** demonstrates the plot of a per base sequence quality distribution of a normal Illumina dataset, while **Fig. 4** gives an example of low-quality Illumina reads and shows the significant drop in the average quality towards reads' end.

One of the basic, but very important nucleic sequence properties is the *GC content*. It is calculated as the total number of guanine and cytosine nucleotides relative to the total sequence length. The per sequence GC content plot is usually used to detect the presence of contamination in the sample. In case of non-contaminated sequencing data, the GC content distribution is typically bell-shaped and matches the theoretical distribution (**Fig. 5**). In the presence of contamination this distribution may contain several separate peaks, each of which would correspond to a different genome with different GC content (**Fig. 6**).

The following group of statistics and plots is typically used to detect a very common problem – The presence of sequencing adapters in the reads. These statistics include per base sequence content, adapter content, k-mer content and overrepresented sequences. The figures below demonstrate how sequence content plots look when working with clean data (**Fig. 7**) and with reads containing adapters (**Fig. 8**).

Adapter and quality trimming

As was demonstrated, sequencing adapters and low-quality ends (for Illumina reads) are the common problems that often arise when analyzing sequencing data. Both of them may significantly affect performance and the quality of the results generated during the next steps of the analysis, such as, for example, genome assembly. Therefore, during the preprocessing step low-quality ends are clipped and the adapters are removed.

One of the tools used for read trimming is called Trimmomatic (Bolger *et al.*, 2014). This tool is capable of clipping reads using quality values and removing adapters from both single-end and paired-end reads. One of the most useful methods for trimming reads is the detection of low-quality ends based on the average quality in the sliding window of a fixed size. A read is cut once the average quality in a window drops below a given threshold. **Fig. 9** demonstrates what the per base sequence quality plot looks like once the reads have been trimmed (plot for original reads is shown on **Fig. 4**). To remove the adapters from the reads a FASTA file with the adapter sequences needs to be fed into Trimmomatic. Other adapter-removing tools, such as cutadapt (Martin, 2011) or NxTrim (O'Connell *et al.*, 2015), can also be used.

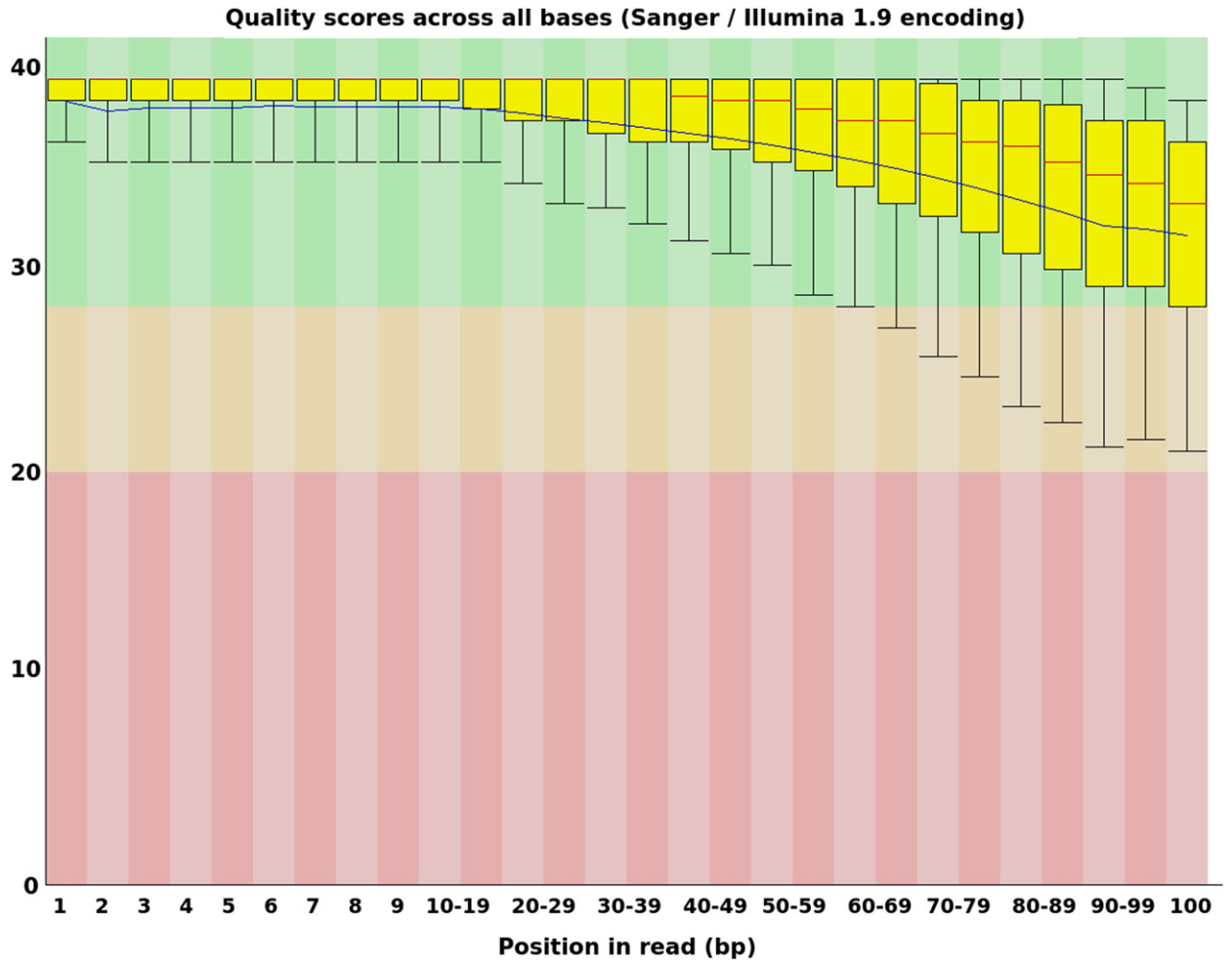

Fig. 3 Per-base sequence quality plot for normal Illumina reads.

Filtering contamination

Another typical problem for sequencing data is contamination – the presence of reads from the genomes of species you did not intend to sequence. Read contamination usually occurs when the sample has not been completely purified and contains the DNA of other organisms. As mentioned above, in cases when the unwanted genome contaminant has a different GC content, its presence can be easily detected using the per sequence GC content plot in FastQC. Otherwise, contamination may be detected using *read alignment* (see next section) or once the genome has been assembled.

In order to filter out the contaminants, one needs to have a database to screen the reads against. In most cases, a reference genome of the suspected contaminant can be used for this purpose. Contaminant reads can be cleaned using read alignment or using a simple tool called bbduk from the bbtools package (Bushnell, 2014).

Mapping Sequencing Data

Goals and problems of short reads alignment

The problem of aligning a read to a reference genome can be reformulated as an attempt to find the position, from which the read was originally sequences. *Read alignment* (or *read mapping*) is essential for many biological and medical applications, such as detecting genomic variations in the sequenced organism and phylogenetic studies.

Although both the local and the global alignments have exact solutions, these algorithms have a quadratic running time and memory consumption, which makes them inapplicable for read alignment due to the large size of genomic sequences (e.g., human genome is almost 3 Gbp long). Additionally, a single sequencing library may contain up to 100 million reads. Thus, in order to perform read alignment for the entire dataset in an observable amount of time, every separate read alignment has to be performed extremely efficiently. Several techniques used to speed up read alignment are discussed in the next section.

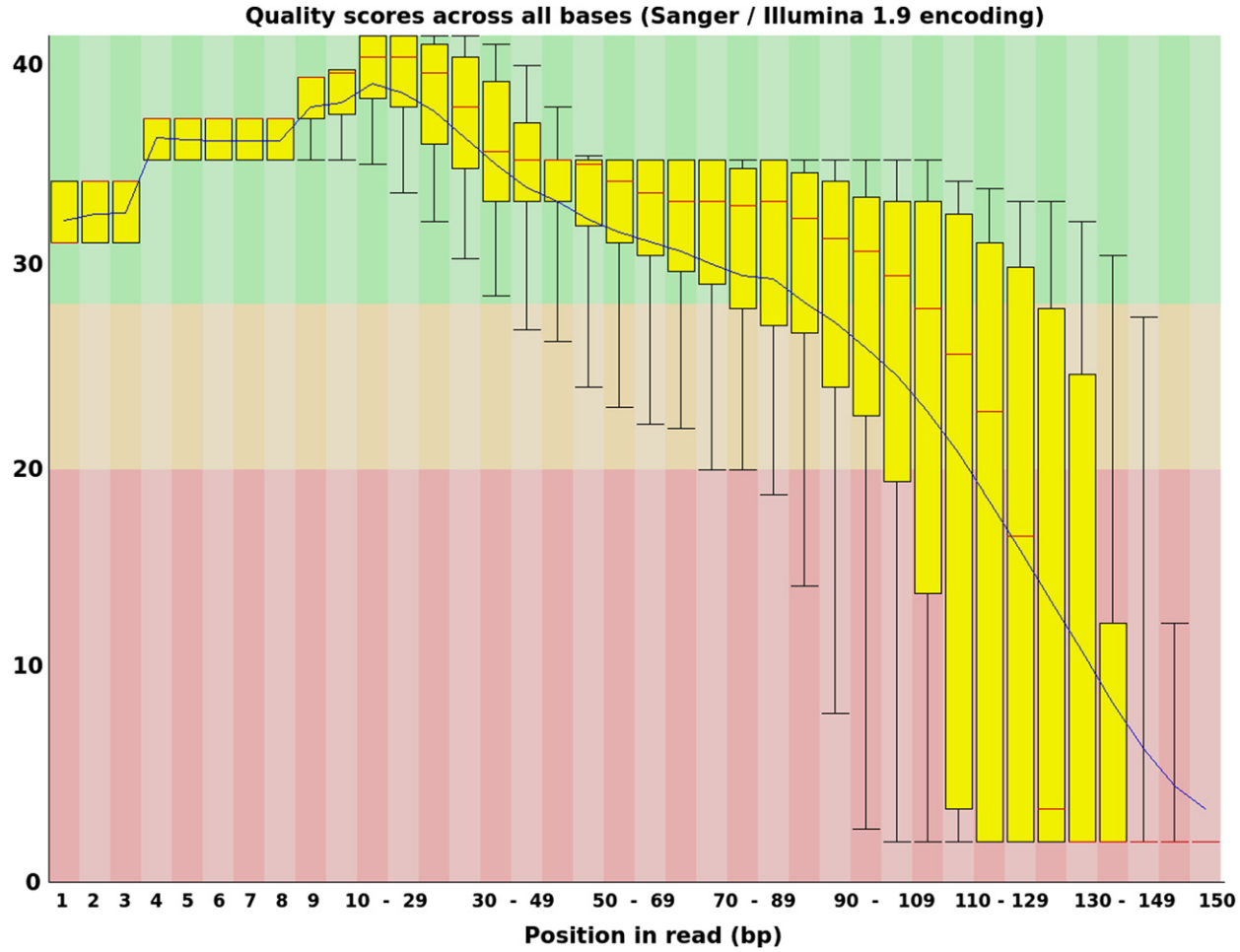

Fig. 4 Per-base sequence quality plot for low-quality Illumina reads.

The problem of read alignment is further complicated by the differences between the reference genome and the genome being analyzed, sequencing errors and genomic repeats. Reads corresponding to repetitive regions can be mapped to several genomic locations and are typically ignored.

In this section we discuss several approaches and tools that are designed for the purpose of aligning reads obtained with different sequencing technologies.

Alignment file formats

Files containing information about the alignment of reads to the reference sequence are typically stored using a specific file format called SAM. SAM files start with a header, which contains all the necessary supplementary information (e.g., lengths of chromosomes in the reference genome). Information about each read alignment is written as a separate line and contains such information as: The position and the strand of the read on the chromosome, name of the chromosome, read ID, the read sequence itself and the corresponding quality string, the information about the mate read (for paired reads) and other supplementary data. Complete information can be found in the SAM format specification.

Since SAM is a text format, it may require a huge amount of disk space, especially when the input files contain hundreds of millions of reads. To reduce disk usage, the binary BAM format was implemented. Typically, BAM files occupy up to 5 times less spaces than SAM files with the same amount of information. SAM/BAM files can be manipulated, converted and processed using a popular package called samtools (Li *et al.*, 2009).

Short read alignment

Modern short read alignment software is based on the same principles as internet search engines – Indexing the content to enable faster query search. Preprocessing of long genomic sequence allows to rapidly search for any of the reads. The genome is often indexed using a suffix array, the Burrows-Wheeler transform, the *FM-index* (Ferragina *et al.*, 2004) or a variation of these algorithms. Since the size of a genome may be large, the preprocessing step may take a significant amount of time, but on the up side, it

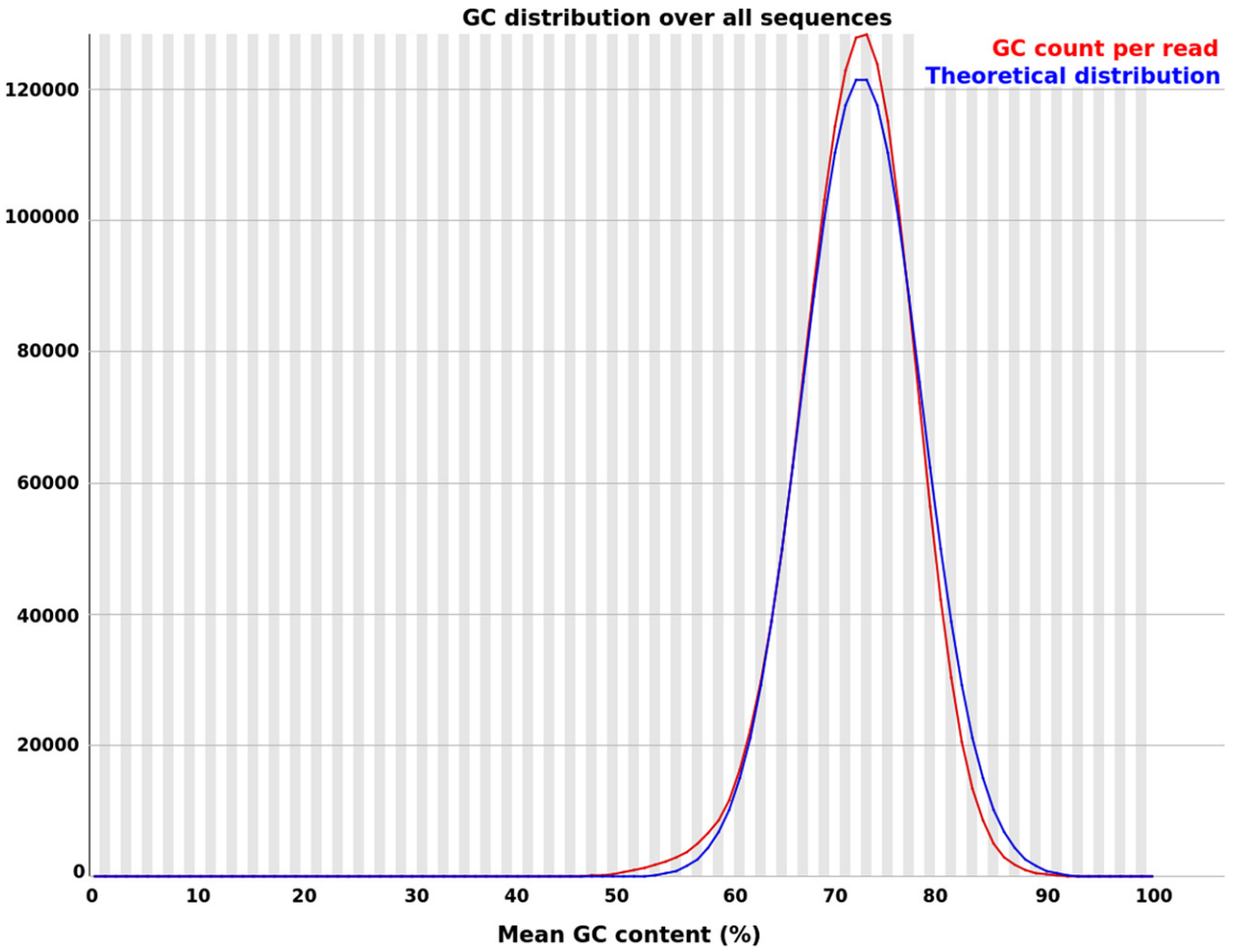

Fig. 5 GC content plot for normal Illumina reads.

needs to be computed only once. Once the index is computed, the read alignment itself can be performed quickly and in parallel, which allows to process reads simultaneously using multiple threads.

The most popular tools for aligning genomic reads are Bowtie2 (Langmead and Salzberg, 2012), various bwa algorithms (Li and Durbin, 2009; Li, 2013), bbmap (Bushnell, 2014) and others. In the case of Ion Torrent sequencing technology TMap can also be used as an alternative. Although comparisons demonstrating the differences in speed and accuracy between a number of popular aligners were published, the final selection of the read alignment software typically comes down to a personal preference based on individual experience.

Variation calling

The main application of short read alignment is detecting different types of genomic variations: *single nucleotide polymorphism* (SNP) sites, small insertions and deletions (*indels*), *copy number variations* (CNV). These kinds of events may have significant biological meaning and thus are of high interest as topics of investigation. For example, a change of single nucleotide in rpoB gene of *Mycobacterium tuberculosis* makes it resistant to Rifampicin.

To perform a variation calling, read alignments are sorted, the BAM files are then indexed, the reads are piled up and the variations are detected by comparing them to the reference sequence. Discovered variations are then filtered based on the specific parameters of the final objective. Filtering criteria may depend on such factors as, for example, the average coverage depth of the sequencing library or the organism ploidy. Details on the different variation calling pipelines can be found in the documentation for the Genome Analysis Toolkit (McKenna *et al.*, 2010).

Discovered genomic variations are typically stored in the VCF file format, or in the compressed BCF binary format. The vcftools toolkit is often used to filter and process these files (Danecek *et al.*, 2011). However, the process of calling SNPs on its own does not provide any biologically meaningful insights. In order to understand the effect of the discovered mutation, SNPs need to be annotated. Annotation of the genomic variations can be performed, using several existing databases (e.g., dbSNP, Ensembl Variation) and tools (e.g., SNPeff (Cingolani *et al.*, 2012), ANNOVAR (Wang *et al.*, 2012)), the selection of which will depend on the types of mutation discovered and its expected effect. *De novo* annotation is also possible, but typically requires a large number of samples to be examined in order to determine the associations between the particular variation and the studied effect (e.g., antibiotic resistance).

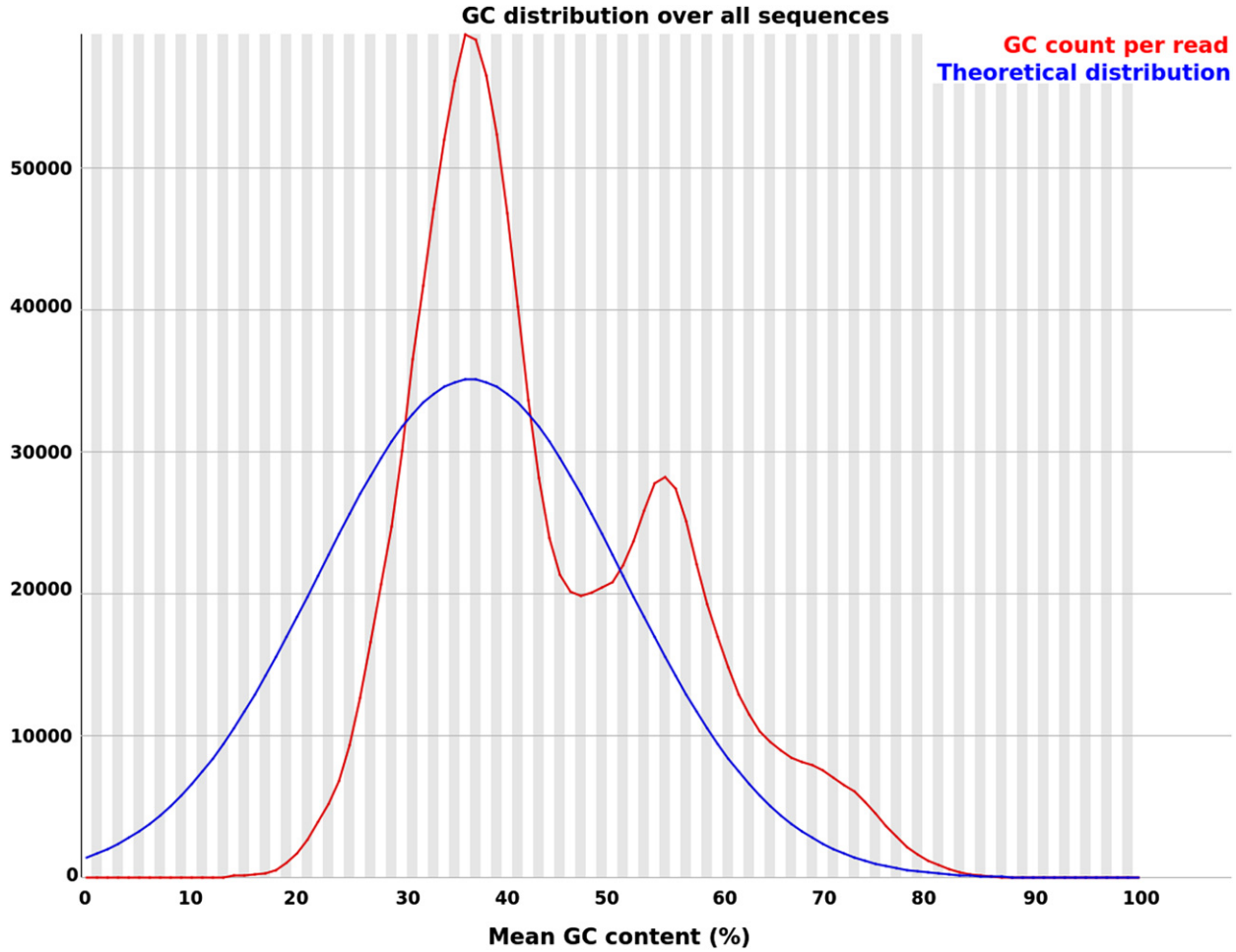

Fig. 6 GC content plot for contaminated Illumina reads.

In addition, once the variations are called, it may be useful to visualize them in a genome browser. Genome browsers are designed to display various genomic features along the chromosomes, such as genes and isoforms. Each genome browser is typically developed to address one particular need and is not universal. For example, to view and examine SNP sites a simple browser called Tablet (Milne *et al.*, 2009) may be used. Alternatively, more sophisticated browsers, such as IGV (Thorvaldsdóttir *et al.*, 2013) or UCSC (Kent *et al.*, 2002) genome browsers can be used.

QC using read alignment

Another possible application of read mapping is an additional QC of sequencing library and calculation of various statistics that cannot be computed without a reference genome. For, example, the number of unaligned, uniquely aligned and multiply aligned reads, faction of the genome covered and substitution error frequencies.

Read alignments may also be used to estimate coverage depth along the genome. **Figs. 10** and **11** show two coverage plots for different E.coli sequencing datasets. On both plots each dot represents an average coverage depth in the window of 1 kbp. **Fig. 10** demonstrates the coverage distribution for a conventional isolate dataset (uniform coverage distribution and more than 99% of genome covered), while plot on **Fig. 11** is constructed using the single-cell MDA-amplified sequencing data (only 96% of genome is covered and coverage depth is highly uneven due to non-uniform amplification process).

Another useful statistic that can be obtained via read alignment is the information about the insert size distribution. **Fig. 12** demonstrates the insert size distribution of an E.coli dataset with an average insert size of 215 bp and a standard deviation of 10 bp.

Plots and statistics shown above may be used to identify problems that cannot be detected using basic QC analysis, such as, for example, problems with size selection and coverage gaps.

Long error-prone read alignment

Reads generated by PacBio and Oxford Nanopore technologies significantly differ from those generated by the previous NGS technologies in terms of read length and accuracy. Alignment tools listed in the previous sections were designed especially for short reads with a rather low error rate and did not perform well out-of-the-box.

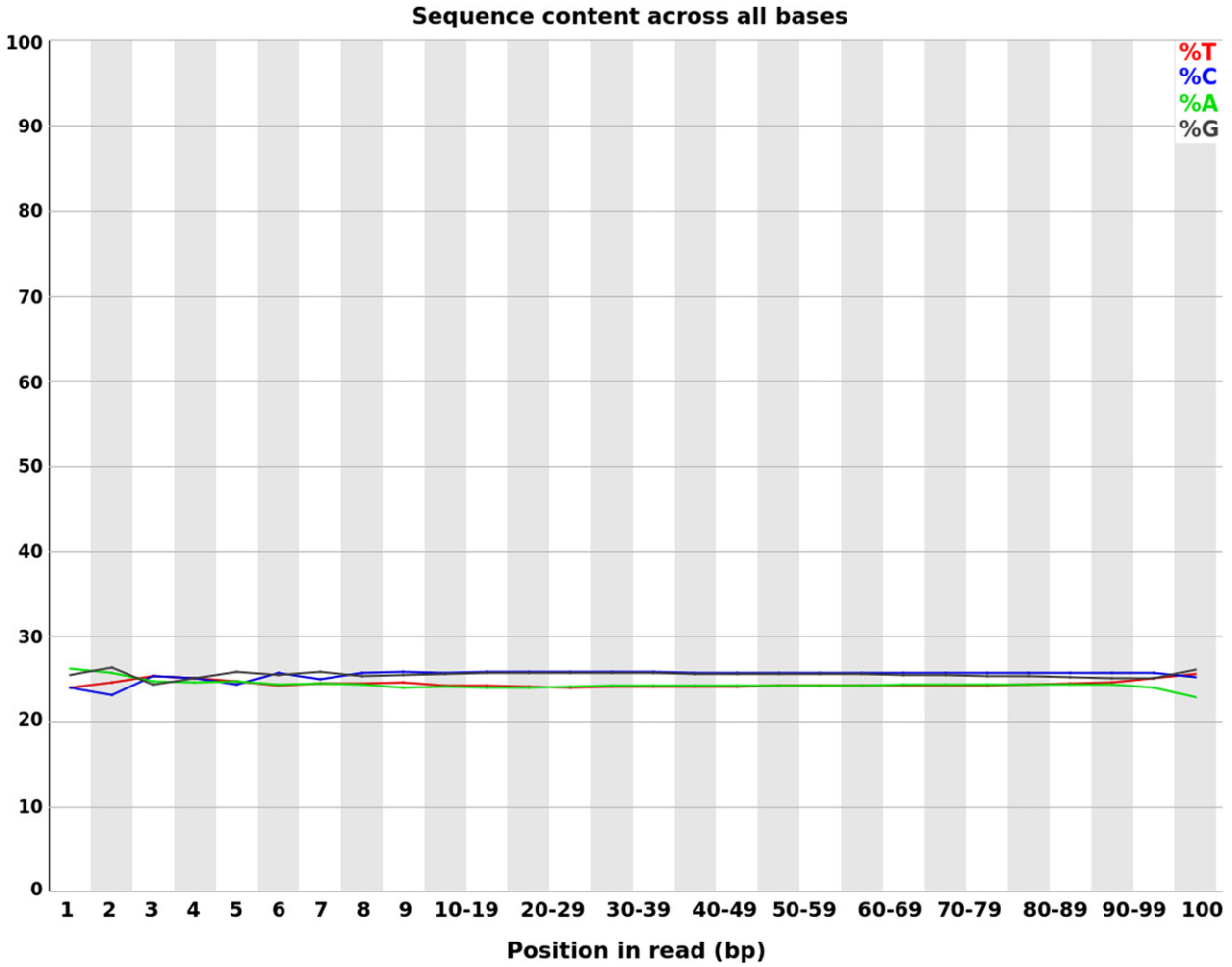

Fig. 7 Per-base sequence content for normal Illumina reads.

Therefore, several new tools were developed for alignment of long error-prone reads, e.g., GraphMap (Sović *et al.*, 2016) and minimap (Li, 2016). The later one is known for mapping reads without outputting actual alignments, which dramatically improves its performance. In addition, the bwa mem algorithm was modified and tuned to support the new family of sequencing technologies. However, due to an extremely high error rate, PacBio and Oxford Nanopores are not the best pieces of technology for calling SNPs and other small variations.

De Novo Genome Assembly

One of the first applications that genome sequencing was intended for is the *de novo genome assembly*. Currently, the problem of assembling a genome from scratch still cannot be considered as completely solved. Even now, in the age of biotechnological and computational progress, a path from the raw biological sample to the finished genome may require years of work and substantial financial resources.

The problem of *de novo* assembly is one of the most popular and algorithmically challenging in computational biology. Dozens of tools were developed during the sequencing era, and yet none of those tools can be named as the best genome assembler in the world for all cases. It is hardly possible to create a universal genome assembler due to the differences in the genomes being sequences, research goals and sequencing technologies.

Despite all recent advances in biotechnology and bioinformatics, it is rarely possible to generate a complete assembly of all chromosomes from raw reads. The key reasons are (i) the presence of genomic repeats and (ii) various sequencing errors and biases. Assembly tools typically generate long genomic regions called *contigs* (stored in FASTA format). The contigs can be further ordered and joined into *scaffolds*, even when the exact genomic sequence and distance between them is unknown. The unspecified nucleotide is usually denoted as an N symbol in the output FASTA file. The scaffolding step is typically performed by long-range sequencing libraries, e.g., mate-pairs.

De novo assembly using long reads

The most intuitive way of solving de novo genome assembly problem is to join overlapping reads with each other one by one – in the same way as jigsaw puzzles are assembled. The approach that implements this simple idea is called *Overlap-Layout-Consensus*

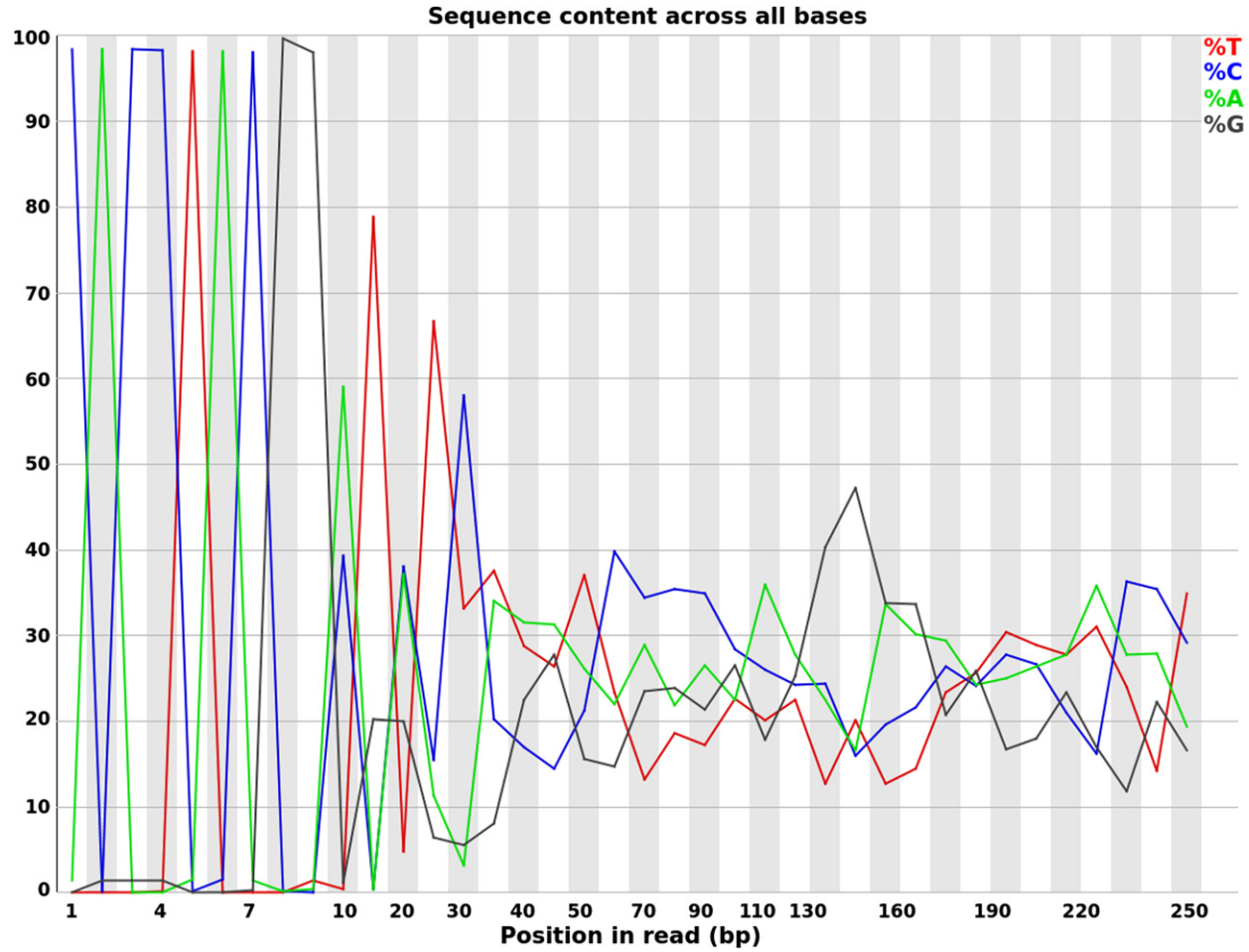

Fig. 8 Per-base sequence content for Illumina reads containing adapters.

(OLC), because it consists of the three consecutive stages. During the first step possible overlaps between all reads are detected, which is typically done using BLAST or similar algorithms. Afterwards, the algorithm constructs an overlap graph in order to determine the layout of the reads, which is then used to compute the consensus sequence and to obtain the resulting contigs. The OLC approach was implemented in multiple popular assembly tools, e.g., Celera (Myers *et al.*, 2000).

As was previously described, Sanger sequencing yields accurate reads up to 1 kbp, typically with low coverage. These characteristics allow to assemble Sanger reads using the OLC approach, which was successfully carried out in multiple groundbreaking genome assembly projects, such as the human genome project. Currently, genomes assembled from Sanger reads with the OLC approach remain the golden standard of genome assembly.

However, this algorithm requires significant computational resources for detecting overlaps between all reads. Although several optimizations can be done to speed up this step, it is still hardly possible to calculate all overlaps for high-covered NGS datasets.

The OLC approach was brought back to life by the recent emergence of the third-generation sequencing technologies – PacBio and Oxford Nanopore. Although Sanger sequencing differs significantly from these novel technologies in terms of error rate and read length, a modified OLC approach appeared to be very useful for their assembly. The main modification compared to the previous implementations of OLC was introduced in the overlap stage, which now implies the alignment of the reads with an extremely high error rate. OLC was successfully reused in such *de novo* assemblers as Canu (Koren *et al.*, 2017) and miniasm (Li, 2016).

De novo assembly using short reads

During the NGS era, sequencing became cheaper, and therefore available to more researchers worldwide. At the same time, it brought multiple algorithmic challenges to computational biology. The problem of *de novo* genome assembly was not an exception. The OLC approach appeared to be impractical for NGS data due to the extreme amount of reads and their short length.

Pevzner *et al.* (2001) proposed to apply the *de Bruijn graph* to the problem of *de novo* genome assembly from short reads. To construct a de Bruijn graph the reads are shredded into a set of smaller sequences of a fixed length k (called *k-mers*), which represent vertices of the graph. In comparison to the overlap graph, the process of constructing a de Bruijn graph does not include

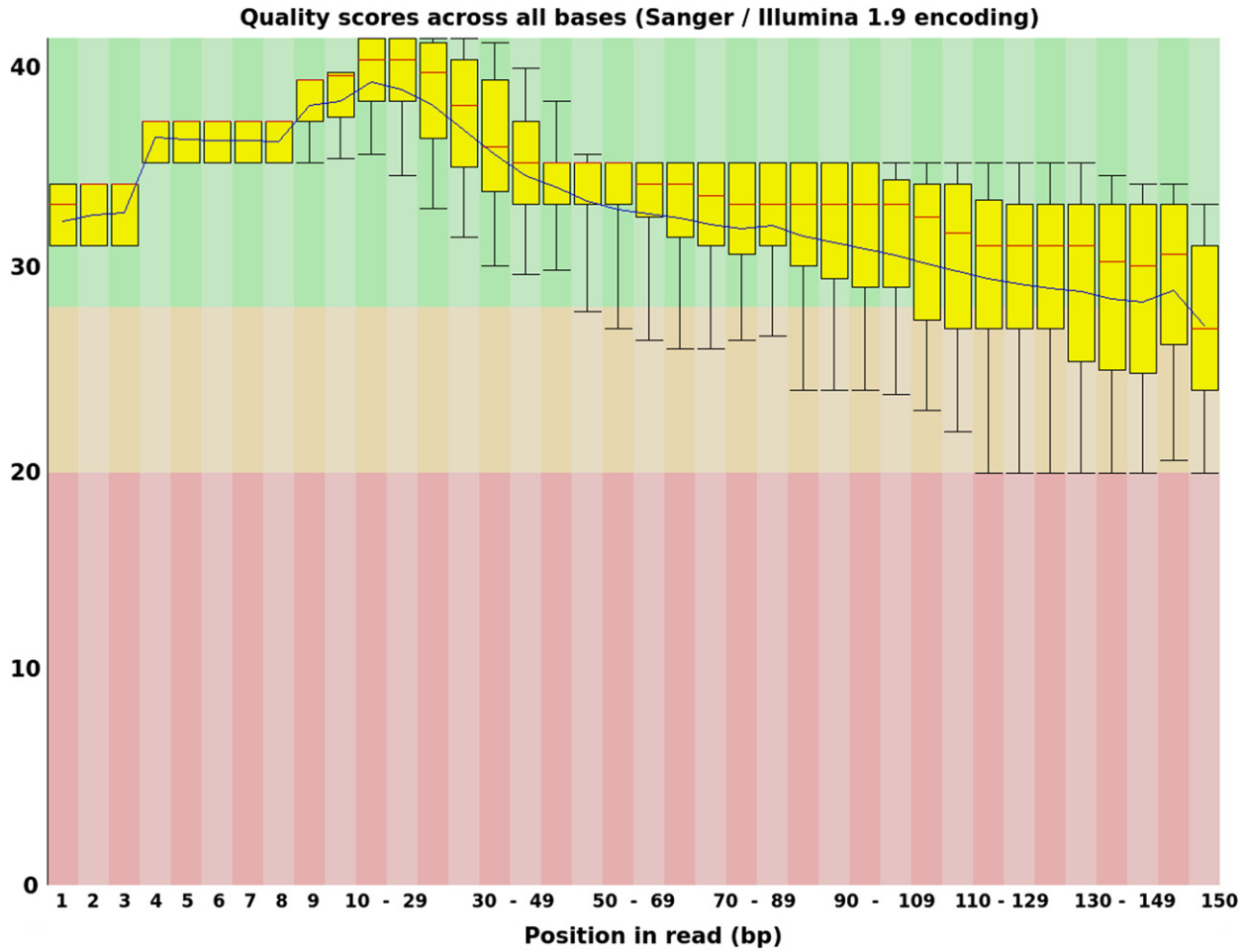

Fig. 9 Per-base sequence quality plot for quality-trimmed Illumina reads.

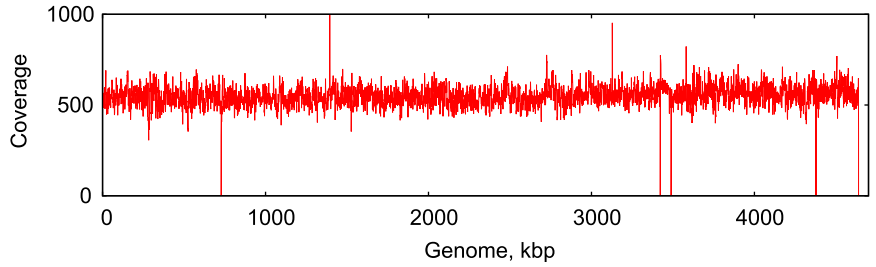

Fig. 10 Coverage depth along the genome for isolate *E.coli* dataset.

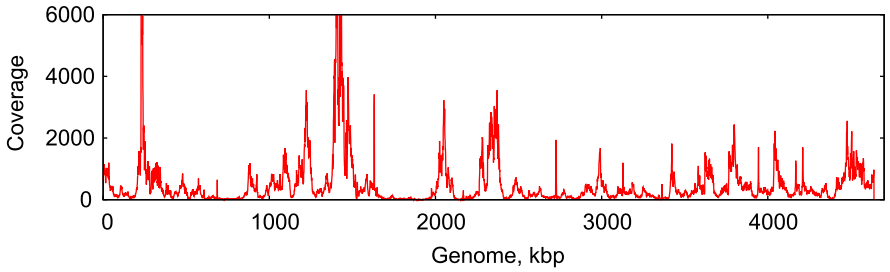

Fig. 11 Coverage depth along the genome for MDA-amplified *E.coli* dataset.

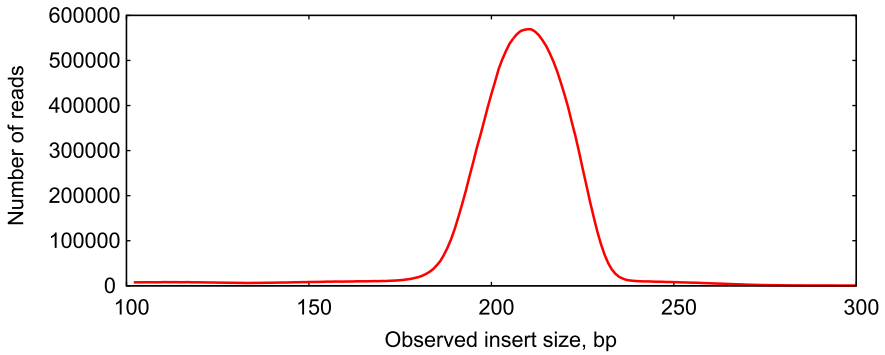

Fig. 12 Insert size distribution for isolate *E.coli* dataset.

read alignment stage, which greatly reduces running time. Since each k-mer is stored only once, the de Bruijn graph approach appears to be more memory-efficient than OLC and therefore suitable for high-coverage datasets with millions of reads.

Although the idea of using the de Bruijn graph was a breakthrough and revolutionized the field of *de novo* genome assembly, its application to real data necessitated the development of several additional algorithms and heuristics. For example, the de Bruijn graph approach appeared to be sensitive to sequencing errors. Each substitution error in a single read creates a number of erroneous non-genomic k-mers, which have to be removed. The process of cleaning the graph from these erroneous k-mers is called *graph simplification*. Once the graph is simplified, *repeat resolution* and *scaffolding* steps are performed to extract contigs and scaffolds from the graph with the help of, for example, read-pairs (Bresler *et al.*, 2012; Prjibelski *et al.*, 2014). The de Bruijn graph approach is implemented in many state-of-the-art genome assembly tools, such as Velvet (Zerbino and Birney, 2008), ABySS (Simpson *et al.*, 2009), SOAPdenovo (Luo *et al.*, 2012) and SPAdes (Nurk *et al.*, 2013).

It is worth noting, that later, in 2005, another similar concept called *String graph* was proposed by Myers (2005). In 2010 it was successfully adapted for use with real data and implemented in the SGA tool (Simpson and Durbin, 2012).

Hybrid assembly

Emergence of novel sequencing technologies also brought the possibility to perform *hybrid assembly* – A process of assembling genomes using reads from different sequencing technologies simultaneously. Hybrid assembly may be useful when long-read libraries have low coverage depth (e.g., less than 20x), which may be insufficient for constructing a reliable consensus string using the OLC approach due to the high error-rate of the reads.

Currently, there are two distinct approaches for utilizing short and long reads simultaneously. In the first approach, short accurate NGS reads are assembled using existing assembly tools to obtain accurate contigs with a small number of substitution and indel errors. Afterwards, long error-prone reads are mapped to these contigs and used for repeat resolution and scaffolding. Since PacBio and Oxford Nanopore technologies are capable of yielding reads up to hundreds of kilobases long, they can be useful for resolving long repeats and closing gaps between Illumina contigs. Some algorithms for hybrid assembly also consider aligning long reads directly to the edges of the de Bruijn graph and perform repeat resolution without explicitly outputting the contigs (Antipov *et al.*, 2015; Wick *et al.*, 2017).

Another approach consists of mapping short accurate reads (typically Illumina) onto long error-prone reads in order to correct sequencing errors in the long reads. Once errors are corrected, the long reads can be fed into a conventional OLC assembler to generate long and accurate contigs (Koren *et al.*, 2012).

Genome assembly evaluation

Assessing the quality of the assembled contigs and scaffolds is an important step for both assembly software developers and their users. By evaluating assemblers on various datasets the developers may gain a better understanding of the disadvantages and weak points of their software, improve it and benchmark against other tools to convince the scientific community that their assembler performs better on a specific variety of datasets. This also means that researchers involved in projects requiring genome assembly now face the problem of having to choose the best assembly tools for their particular needs.

One of the first attempts to benchmark genome assembly tools was made by the organizers of the Assemblathon competitions, who suggested the idea of assembling several datasets and then comparing the results submitted by the different teams. Assemblathon 1 offered simulated data, while Assemblathon 2 provided real sequencing data from three vertebral genomes. The resulted contigs were compared using multiple different metrics and the results were then published (Earl *et al.*, 2011; Bradnam *et al.*, 2013).

Benchmarks of various assemblers were also performed by the developers of the assembly tools. However, since until recently no standard tool for evaluating genome assembly quality existed, most of these comparisons were made using custom in-house scripts. In addition, to justify the research, in most cases dataset selection was subjective and the benchmarks typically revealed that the assembler described in the paper outperforms all other tools. However, several benchmarks were performed and published by independent research groups with the goal of identifying the strong and weak points of different genome assembly software (Salzberg *et al.*, 2012; Magoc *et al.*, 2013).

In 2012 GAGE (Salzberg *et al.*, 2012) and QUAST (Gurevich *et al.*, 2013) tools were developed to allow researchers to perform genome assembly quality evaluation on their own. These tools are mostly designed for assessing assemblies of model organisms, i.e., when the reference genome is known. Reference-based quality evaluation allows to perform an excessive analysis of the assemblies by providing a number of informative metrics, such as the fraction of the genome covered, *misassemblies* statistics as well as mismatch and indel error rates.

While reference-based assembly evaluation is useful for testing and comparing genome assemblers, *de novo* evaluation is an integral part of the genome assembly projects. In general, there are three different approaches for reference-free assembly quality assessment. The first one includes calculation of basic statistics (e.g., total assembly length, *N50*) and is useful for initial quality control of the assembly. Another method relies on mapping the raw reads back to the assembly and estimating the likelihood of this particular assembly being generated by the given reads (Hunt *et al.*, 2013). The last group of methods involve counting genes in the assembled sequences. The BUSCO tool (Simão *et al.*, 2015) is designed for detecting the presence of *universal single-copy orthologs*, typical for the specified kingdom, phylum or class. Such tools as GeneMark (Lukashin and Borodovsky, 1998) and Glimmer (Delcher *et al.*, 1999) identify genes *de novo* using Hidden Markov Models (see Section "Genome Annotation" for more information).

Assembly of microbial genomes

Microbial genomes are known for their rather simple repeat structures (compared to the eukaryotic genomes) and therefore are easier to assemble. Since most of the repeats in bacterial genomes do not exceed 7 kbp in length, it is usually possible to assemble a complete bacterial genome with the help of either a mate-pair library (Vasilinetc *et al.*, 2015) with a large insert size or long-read sequencing data (Antipov *et al.*, 2015; Wick *et al.*, 2017). Current Illumina Nextera Mate-Pair protocol allows to obtain a complete bacterial genome using only a single short-read sequencing library (Vasilinetc *et al.*, 2015). At the same time, a single PacBio or Oxford Nanopore library with a decent coverage depth (e.g., higher than 20x–30x) also allows to obtain a complete bacterial genome in most cases.

To generate a sequencing library for a prokaryotic organism millions of identical cells are needed. However, nowadays most of known bacteria cannot be cultivated in the lab. To generate enough genomic material for the sequencing experiment, bacterial genome from a single cell can be amplified using for example the Multiple Displacement Amplification method (Lizardi and Yale University, 2000). Compared to the conventional isolate datasets, the reads obtained via the MDA technology typically have a highly uneven coverage depth across the genome, which complicates the assembly process. To address the problem of single-cell bacterial assembly SPAdes (Nurk *et al.*, 2013) and IDBA-UD (Peng *et al.*, 2012) assemblers were developed. Recently, a first bacterial genome was completely assembled and finished using only single-cell sequencing data (Woyke, 2010).

Assembling metagenomes

In real life, bacteria typically exist within large communities that may contain thousands of different species. Sequencing and studying the whole community at once is called *metagenomics*. Until recently, metagenomic studies mostly included such research goals as detecting the variety of species (e.g., using 16s rRNA genes) and their abundances within the specific community. The development of MEGAHIT (Li *et al.*, 2015) and metaSPAdes (Nurk *et al.*, 2017) assemblers allows to obtain decent bacterial assemblies from metagenomic datasets.

Metagenome assembly is characterized by such challenges as the presence of related strains within the community, conservative genomic regions shared between different species and uneven coverage depth due to variations in the abundances of the different bacteria in the sample. In addition, metagenomic datasets typically contain enormous amount of reads to capture genomic information for underrepresented bacteria. Since the total length of all genomes within the community may be similar to the length of a mammalian genome, metagenome assembly appears to be an extremely challenging problem not only regarding the quality of the resulting contigs, but in terms of computational resources required as well.

Assembly of eukaryotes

Eukaryotic genomes are known for their complex repeat structures, which complicate the problem of genome assembly by a margin. For example, the human genome contains various repeat families, such as ALUs (short 300 bp repeats with over a million copies in the genome), other short interspersed nuclear elements (SINEs), long interspersed nuclear elements (LINEs, typically about 7 kbp long), terminal repeats and various tandem repeats. To resolve such repeats eukaryotic genomes are assembled using a large variety of sequencing data: mate-pair libraries with different insert sizes, fosmid mate-pairs and genomic maps.

Genome Annotation

Assembled from short sequenced fragments genomic DNA contains all of the information about the metabolism of the organism, its development and responses to external stimuli. Most of these functions are carried out by proteins encoded within the genomic sequence. This means that the primary goal of any genome analysis is to identify protein encoding genes, determine their functions and then define and characterize the sections of DNA that regulate the amounts of these proteins being synthesized based on an exact point in time and conditions. This process is called *genome annotation*. At the start of the annotation process all possible start and end codons that delimit the parameters of protein encoding genes are identified. This stage is called *ORF calling* and will be described in more detail in the "Markov models and gene prediction" section.

It is important to keep in mind that ORF calling methods used for bacterial genomes cannot be applied to eukaryotic genomes, since the latter contain both coding (exons) and non-coding regions (introns). Introns are also significantly longer than exons (e.g., exons take up about 2% of the entire length of the human genome). Information, such as the frequency of encountering particular amino acids and synonymous codons and the knowledge of the length of the exons and introns is used in combination with other statistical data to identify genes in eukaryotic genomes. Despite the fact that genome annotation using statistical identification rules yields only very approximate data, it is still of great value as part of the whole process.

The best way to tackle this task is to compare the sequences of several different, but related genomes. Since exons and introns evolve with different speed (exons, evolve comparatively slower than non-coding segments, aka introns), we can use this information to highlight similar regions by comparing genome sequences to each other and as such, identify the overall location of the exons. The delimitations of these coding regions are then further refined via statistical methods.

Comparative analysis is also of help when trying to predict the function of the particular proteins encoded by the genes. If the level of similarity between the new protein and a previously researched and well described one (studied in a laboratory setting) is high, they can be considered as similar and are likely to have identical functions. The lower the degree of similarity between proteins, the more different their functions will be. To make sure that two protein coding genes from two different genomes are in fact the same gene, we must make sure that neither of these genes has a closer relative in a different genome. This stands true even if the function of these proteins is extremely similar. The quality and accuracy of such analysis is directly dependent on the availability of a large amount of high quality whole genome assembly reference data.

High quality predictions of protein functions play a crucial role in a large number of theoretical and experimental studies, such as for example research aiming to gain a deeper understanding of the underlying biochemical conditions related to numerous genetic diseases.

Not all genes in a genome encode proteins. Thus, instead of synthesizing protein, non-coding RNA genes are responsible for the production of functional RNA products. Non-coding RNA genes can be predicted by using various programs tRNAscan-SE (Lowe and Eddy, 1997), RNammer (Lagesen *et al.*, 2007) and BLAST (Altschul *et al.*, 1990) and data sources, as well as performing homology-based searches in such databases as Rfam (Burge *et al.*, 2013), the plant snoRNA database (Brown *et al.*, 2003) (for plant genomes), GenBank (Benson *et al.*, 2004) and ASRG (Wang and Brendel, 2004). RNA sequencing reads (if available) can be further mapped against the appropriate databases to strengthen the generated predictions.

The comparative sequence analysis approach coupled with functional genomics experiments allows annotating non-protein coding genomic regions.

A number of comprehensive annotation systems are available for carrying out functional annotation (see Further reading) for the interested reader to consider and use.

Markov models and gene prediction

Knowledge of the intrinsic gene structure is the main mechanism used in gene prediction. Indeed, the majority of genes have ATG (methionine) as a starting codon, though sometimes GTG and TTG are used as alternative starting codons. Certainly, there may be multiple ATG, GTG, or TGT codons in a frame, which means that the presence of these codons at the beginning of the frame does not necessarily give a proper indication of the translation initiation site. Therefore, another type of DNA structure, also associated with the translation initiation event, has to be used in addition to nucleotide content. One of such features is the ribosomal binding site (RBS) located upstream of the translation start codon. Knowing the ribosome binding site can help to determine the start codon. Each protein coding region ends in a stop codon that causes translation to stop. There are three possible stop codons, namely, TAG, TAA, and TGA, and all three of them are fairly easy to identify.

Manual prokaryotic gene identification relies on brute-force determination of ORFs and major features related to prokaryotic genes. As a part of this approach the DNA is first translated in all six possible frames (three frames on one strand and three frames on the opposite strand). It is worth mentioning that in a noncoding region a stop codon occurs on average every twenty codons, which means that a frame longer than thirty codons without any stop codons could be considered as putative gene coding region. The putative frame is further manually confirmed by the presence of other features such as a start codon and the RBS sequence.

Markov models can be very helpful in providing a probabilistic description of a gene sequence. A *Markov model* describes the probability distribution of nucleotides in a DNA sequence, where the conditional probability of a nucleotide occurring in a particular sequence position depends only on the nucleotides at k previous positions. The value k is called the *order* of a Markov model. This mean that a zero-order Markov model implies that each of the bases occur independently with a given probability. This is a typical situation for noncoding parts of a genome. A first-order Markov model (or *Markov chain*) stipulates that the probability of the occurrence of a particular base will depend only on the base preceding it. A second-order model looks at the preceding two bases to determine the probability of occurrence of the following base. This model is a better fit for the codons in a coding sequence.

The use of the Markov models in gene finding exploits the idea that oligonucleotide distributions in the coding regions are different from those for the noncoding regions. Indeed, having the Markov models that describes the structure of coding and non-coding regions (these can be represented with Markov models of different orders) we can calculate the probabilities of occurrence of a particular sequence under these models. Comparing these probabilities or, better yet, calculating the likelihood ratio we could decide whether the particular sequence is a part of a coding region, or not. Since a fixed-order Markov model describes the probability of a particular nucleotide that depends on previous k nucleotides, the longer the nucleotide sequence, the more accurately the internal structure can be described for the coding region. Therefore, the higher the order of a Markov model, the better it can predict genes.

A protein-encoding gene is formed by nucleotides organized in triplets as codons, hence the more effective Markov models are constructed from the sets of three nucleotides, explaining the observed distributions of trimers or hexamers, and so on. The parameters of a Markov model have to be trained using a set of sequences with known gene locations. Once the parameters of the model are established, the model can be used to estimate the probabilities of trimers or hexamers in a new sequence.

Under normal circumstances the pairs of codons tend to correlate, and as a result, the frequency of a particular six nucleotides appearing together in a coding region can be much higher than occurring by random chance. As a result, a fifth-order Markov model, which calculates the probability of hexamer bases, can detect nucleotide correlations typical for coding regions more accurately and is in fact most often used (Wang *et al.*, 2004).

A potential problem of using a high-order Markov chain is that if there are not enough hexamers, which happens in short gene sequences, the method's efficacy might be limited. One way to deal with this limitation is a variable-length Markov model also known as the *interpolated Markov model* (IMM). The IMM samples the largest number of sequence patterns with Markov model orders (k) ranging from 1 to 8 and uses a weighting scheme, placing less weight on rare k-mers (outlining the worse accuracy of corresponding probability estimation) and more weight on more frequent k-mers. The probability produced by the final model is the sum of probabilities of all weighted k-mers. In other words, this method has more flexibility in using Markov models depending on the amount of data available. Higher-order models are used when there is an abundant amount of data and lower-order models are used when the amount of data is smaller (Salzberg *et al.*, 1998).

Surprisingly, the gene content and length distribution of prokaryotic genes can vary a lot. The majority of genes are in the 100–500 amino acids range with a nucleotide distribution derived from the GC content of the organism. However, there are atypical genes that are shorter or longer and have a variety of nucleotide frequency profiles. These genes tend to escape detection using the gene models trained on the typical data. Moreover, the models trained on the whole spectrum of the genes tend to produce worse results due to reduced precision. Ideally, to describe all genes in a genome fully more than one Markov model is needed. The standard way to combine different Markov models (that would represent typical and atypical nucleotide distributions) is via *Hidden Markov Models* (HMM).

A HMM combines two or more Markov models within only one consisting of observed states and others representing unobserved (or "hidden") states that influence the outcome of the observed states (such unobserved state could be, for example, the type of the gene, or a particular part in the gene the nucleotide composition of which would differ from the other parts). In an HMM, as in a Markov model, the probability of going from one state to another state is the transition probability. Each state may be composed of a number of elements. For nucleotide sequences, there are four possible symbols, namely A, T, G, and C in each state. For amino acid sequences, there are twenty possible symbols. The probability value associated with the occurrence of each of the symbols in each state is called the emission probability. To calculate the total probability of a particular path of the model, both transition and emission probabilities linking all the "hidden" as well as observed states need to be taken into account.

The most popular prokaryotic gene prediction/ORF calling software are: Glimmer is based on interpolated Markov models (Salzberg *et al.*, 1998). GeneMark (Lukashin and Borodovsky, 1998) uses inhomogeneous three-periodic Markov chain models of protein-coding DNA sequences and could be viewed as an approximation of an HMM approach (Azad and Borodovsky, 2004).

Analyzing Transcriptomic Sequences

Quality Control of Rna-Seq

Initial quality control of RNA-Seq data can be performed using the same FastQC tool (Andrews, 2010), as used for genomic data. However, the FastQC reports for RNA-Seq data should be interpreted somewhat differently, compared to the genomic data. The main differentiating criteria for RNA-Seq data is the uneven coverage depth and the presence of ribosomal RNA sequences. Due to their high abundance, rRNA reads may introduce additional peaks into the GC content plot (**Fig. 13**), as well as create non-uniform k-mer content (**Fig. 14**).

The presence of unwanted rRNA reads is a typical problem for RNA-Seq experiments. Some sequencing protocols aim to filter out rRNAs during sample preparation, but such approach, however, may only remove a fraction of the rRNA reads from the dataset. Reads from rRNAs can be removed as contamination using, for example, the bbduk script from the bbtools package (see Section "Sequencing Data Formats". for the details), with known rRNA sequences provided as the database.

Aligning RNA Sequences

While alignment of prokaryotic RNA sequences to the reference genome can be performed with the same algorithms and methods used for DNA sequence alignment, the process of mapping eukaryotic RNA differs significantly due to the presence of splicing events. To support long insertions and skip intronic sequences splicing alignment algorithms are required. The most popular software tools for RNA to DNA mapping are BLAT (Kent, 2002), GMAP (Wu and Watanabe, 2005) and Splign (Kapustin *et al.*, 2008).

The problem of mapping eukaryotic RNA-Seq reads to the reference genome is further exacerbated by their short length and the presence of sequencing errors. One of the first tools for mapping RNA-Seq reads – TopHat – was based on the Bowtie aligner for genomic reads (Trapnell *et al.*, 2009) Later, STAR (Dobin *et al.*, 2013) and Hisat (Kim *et al.*, 2015) aligners were developed specifically to address the problem of rapid and accurate mapping of RNA-Seq data. All listed tools can take the gene database in

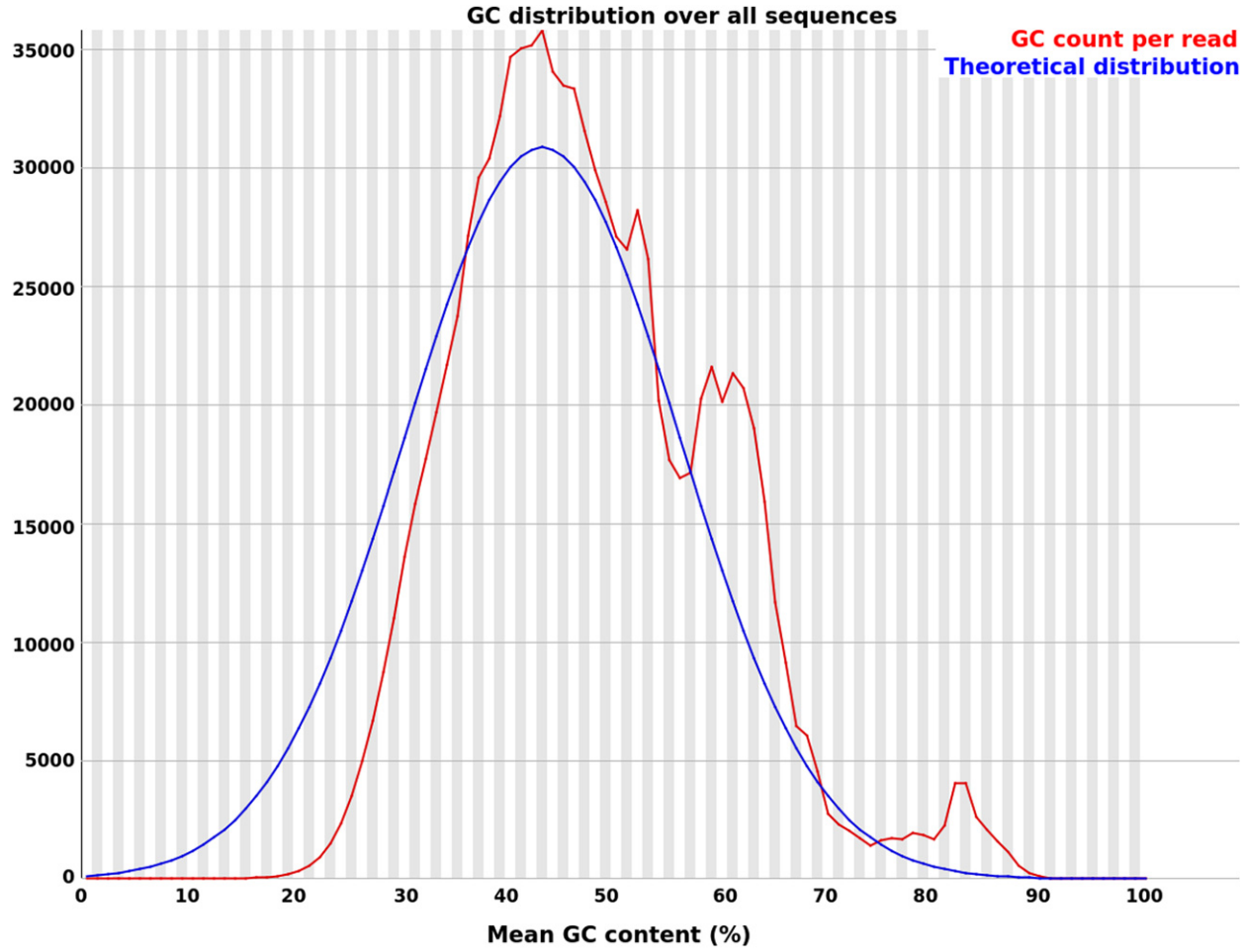

Fig. 13 GC content plot for normal Illumina RNA-Seq reads.

GFF format as input in addition to the reference genome, which increases the accuracy of the alignments and is especially true in the case of reads containing the splice junctions.

Mapping RNA-Seq reads can be of use as an additional reference based quality control. For example, the qualimap (Okonechnikov *et al.*, 2015) tool calculates such statistics as fractions of exonic, intronic and intergenic reads, gene coverage profile and splice junction distribution. The RSeQC (Wang *et al.*, 2012) can also be used as an alternative tool. In addition to the statistics listed above, it is capable of determining the insert size for paired-end RNA-Seq reads, estimate mismatch content and detect whether RNA-Seq data is *strand-specific* or not. In strand-specific RNA-Seq experiment the reads are sequenced either from the same or from the opposite strand of the gene that produces the RNA, which then allows to detect actual gene strand. RNA-Seq alignment can also be used to detect fusion genes – chimeric genes that can appear during genomic rearrangements in cancer cells (Kim and Salzberg, 2011).

Gene Counting and Differential Expression

The main application of RNA-Seq is the estimation of gene expression levels and *differential expression analysis* between multiple samples. Differential expression analysis implies the comparison of expression levels of the same genes/isoforms between samples under different conditions, such as cells under different treatments, healthy and ill patients, patients before and after medical treatment. The number of samples analyzed is the key to a successful analysis as only a correct choice will fully take advantage of RNA-Seq possibilities. To perform the differential expression analysis, a study requires at least three biological samples for each condition analyzed. This is because, when comparing gene expression between the different conditions, sample specific differences may affect the analysis and thus lead to inadequate conclusions, since these differences may not be related to the subject of research. Having an appropriate number of samples for each condition will reduce the impact of sample specific variability on the analysis. Both biological and technical variabilities within the samples can be reduced by normalization. The number of samples analyzed is also essential for the statistical tests required by any differential expression analysis. Most of them need at least three or more samples per condition to provide a statistically sound result. Less samples will suffice only for obtaining exploratory results, while more samples will increase the power of the analysis.

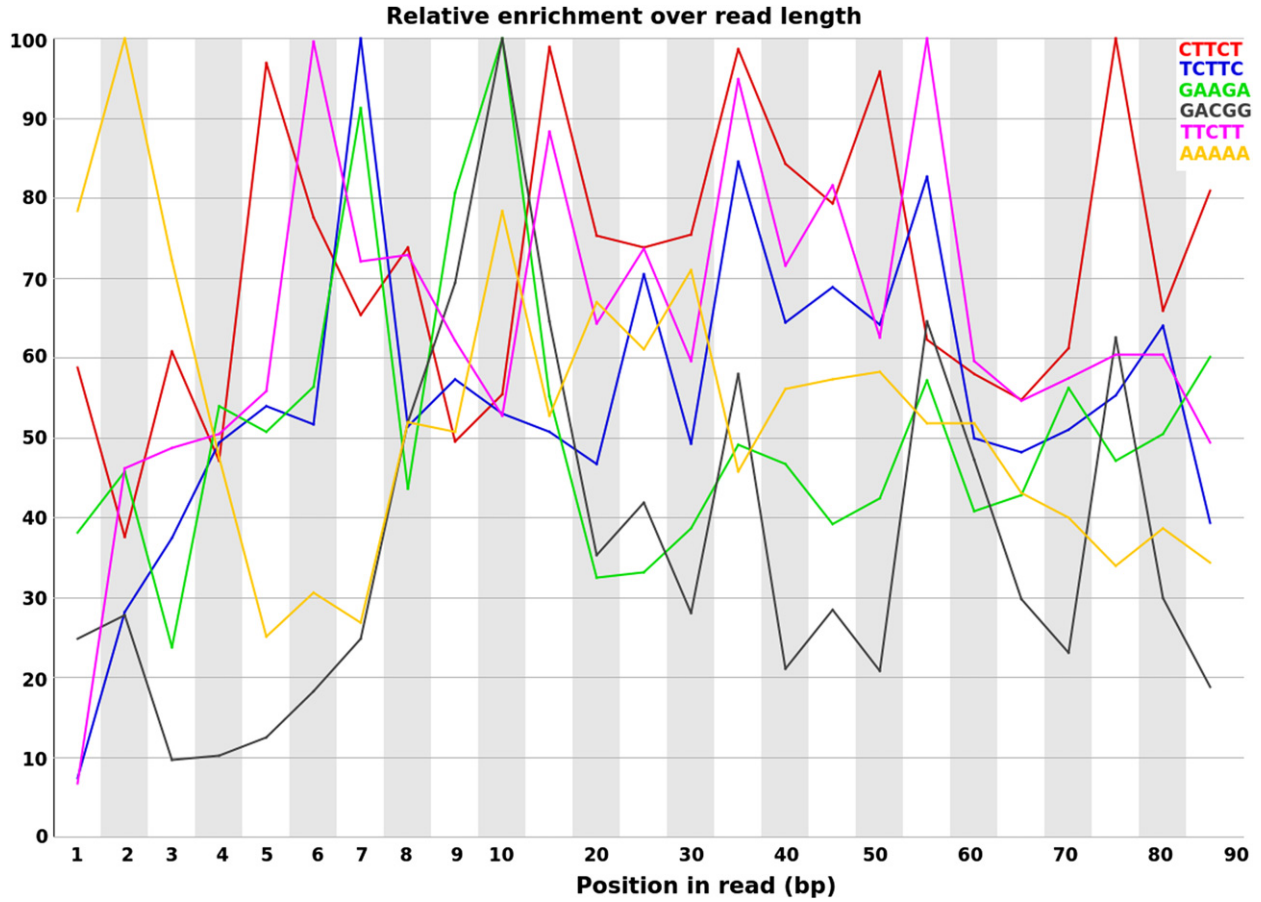

Fig. 14 k-mer distribution for normal Illumina RNA-Seq reads.

Since the isoforms of a gene contain the same exons, assigning each read to a specific isoform is much more complicated task than assigning those reads to the corresponding gene. Therefore, most RNA-Seq pipelines focus on gene level analysis, consisting of several steps.

Firstly, RNA-Seq reads are mapped to the reference genome ignoring reads that map equally well to more than one genomic location (multi-mappers) and assigned to the genes from the database. Afterwards, the number of reads assigned to each gene are counted, e.g., using htseq-count (Anders *et al.*, 2015). Some tools also allow to perform quantification without actually mapping the reads to the reference genome (Bray *et al.*, 2016). The gene counts can be normalized, usually by sequencing depth and gene length. The normalized counts are used to estimate relative change of the expression levels between different samples with a statistical test, e.g., a t-test or a test created specifically for RNA-Seq data. State-of-the-art software tools such as DESeq2 (Love *et al.*, 2014) and edgeR (Robinson *et al.*, 2010) require unnormalized raw counts and perform themselves the normalization step.

The results of the differential analysis are typically represented as a table with genes as rows and several values for each gene. Among these numbers, the most important are the measure of the change in gene expression between the conditions (usually a log fold-change) and the result of the statistical test, usually an *adjusted p-value*. Since thousands of tests are made, the adjusted p-value indicates the significance of the difference in expression (p-values are calculated for null-hypotheses stating that the expression level is the same for all conditions). The decision for calling a gene differentially expressed is not straightforward: it depends on arbitrary threshold values for both log fold-change (how much the expression has to change) and adjusted p-value (how many false positives can be tolerated).

Another way of exploring differential expression analysis results is using a heat map (see example in **Fig. 15**). Table rows correspond to the genes being examined, and each column corresponds to a single sample. Cells color represents estimated genes expression levels. The table is usually sorted according to the change in expression level between different conditions.

One of the possible biases that can significantly affect the results of differential expression analysis is the so called batch effect. The batch effect takes place when the difference between samples is caused by sample preparation or sequencing protocols, flowcell inconsistencies, variations between sequencer runs (Dündar *et al.*, 2015). These factors may lead to inappropriate calculation of expression levels and therefore, misleading interpretation of the results. *In silico* removal of the batch effect is implemented in various differential expression analysis toolkits, e.g., Deseq2 (Love *et al.*, 2014).

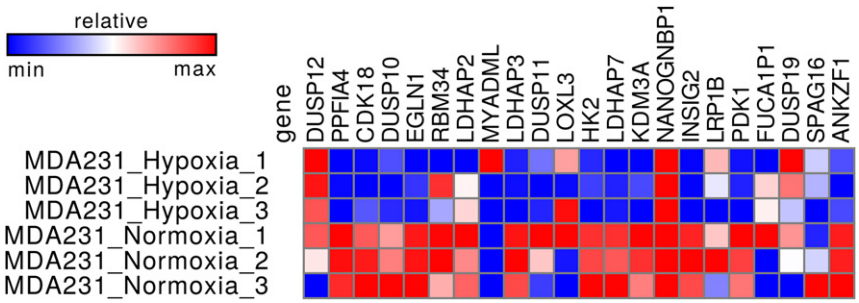

Fig. 15 An example of gene expression heat map for 6 samples under 2 different conditions and 22 genes.

De Novo Transcriptome Assembly

The *de novo* transcriptome assembly may appear to be a simpler problem than genome assembly due to the absence of long complex repeats and a smaller transcriptome total length. However, uneven coverage depth caused by varying expression levels, presence of isoforms from the same gene and paralogous genes complicate the process of transcriptome assembly. Most of the modern de novo transcriptome assembly tools, such as Trans-ABySS (Robertson *et al.*, 2010) and SOAPdenovo-Trans (Xie *et al.*, 2014), are based on the existing de Bruijn graph genome assemblers. Trinity is the only assembler that was initially developed as stand-alone RNA-Seq assembler (Grabherr *et al.*, 2011).

As metagenomic studies became more popular, metatranscriptomics sequencing experiments have also become more frequent. Sequencing RNA of the bacterial community helps gain an understanding of its functionality. In some studies the usage of metatranscriptomic data is limited to the estimation of expression levels for known genes and tracking the signal pathways in the community under different conditions. In some projects, however, the *de novo* metatranscriptome assembly is also performed, although only a few tools are available now (Leung *et al.*, 2015; Ye and Tang, 2015).

Since the result of the *de novo* transcriptome assembly is completely different from the genomic contigs, other methods for quality evaluation are also needed. However, the same three distinct approaches can be applied for the assessment of assembled transcripts: Reference-based, based on gene content and *de novo*.

Reference-based approach includes alignment of the transcripts either to the reference genome with a known gene database via a spliced alignment method or to the isoform sequences themselves. The resulting alignments can be used to compute various statistics and metrics, such as *misassembled* transcripts, number of assembled genes and isoforms, duplication ratio along with mismatch and indel error rate. Recently developed rnaQUAST (Bushmanova *et al.*, 2016) and Transrate (Smith-Unna *et al.*, 2016) tools are designed specifically for the reference-based quality evaluation of *de novo* transcriptome assemblies.

Similar to the genome assemblies, the gene content can be estimated using BUSCO (Simão *et al.*, 2015), which allows to search for single-copy orthologs that are universal for the specific group of organisms. *De novo* gene identification can be performed with a modified version of a popular tool GeneMark, called GeneMarkS-T (Tang *et al.*, 2015). *De novo* quality evaluation based on raw reads for RNA-Seq assemblies is implemented in such tools as Transrate (Smith-Unna *et al.*, 2016) and Detonate (Li *et al.*, 2014).

Protein Data Analysis

All living cells contain proteins that are of a great importance for all biological objects. The structure of proteins is very complex and includes several levels of organization. The primary structure of a protein is usually represented as a string of amino acids starting from the amino-terminal (N-terminal) to the carboxyl-terminal (C-terminal) of the protein. It can be obtained as a result of the direct sequencing of the protein or deduced from the corresponding DNA or mRNA, since the order of bases on a DNA strand specifies the order in which the amino acids are linked together during translation. The amino acid sequence of proteins or peptides is basic information to understand proteins or peptides and predict its post-translational modifications and functions.

Protein Sequencing Technologies

The process of determination of the amino acid sequence is known as protein sequencing. Amino acids may be represented by a single- or three-letter codes. It is worth to mention that sequencing methods were originally developed specifically with proteins in mind, since people thought that proteins were the main genetic material. Researchers believed that the structure of DNA was too simple (composed of only four nucleotides) to play this role comparing to proteins, that can be built of 20 different amino-acids and thus have a higher heterogeneity as molecules. All protein sequencing methods can be divided into two groups: one group provides N-terminus sequence of a protein, while the second group of methods is used to sequence and identify the entirety of the protein. A label-cleavage method for protein sequencing was developed by Pehr Edman in 1950s (Edman *et al.*, 1950).

Edman degradation procedure (**Fig. 16**) is based on a three-stage reaction that labels and removes the N-terminal residue of a polypeptide, which can then be identified as a phenylthiohydantoin (PTH – Edman reagent) derivative.

Phenyl isothiocyanate, $C_6H_5N{=}C{=}S$, is a reagent that permits the progressive removal and identification of the N-terminal amino acid, leaving the rest of the chain unaffected. The truncated peptide after the first degradation has a new N-terminal amino acid that can again react with phenyl isothiocyanate and thus makes progressive removal of the subsequent amino acids possible. This technique cannot be applied if the N-terminus is modified. Removed amino-acid residues are then identified by chromatography (Berg *et al.*, 2002).

Edman sequencing is automated (Niall, 1973) and uses a protein sequenator that allows sequencing peptides of about 5–50 amino acids long. Polypeptides longer than 50–70 amino acids must be cleaved into smaller peptides before sequencing. Use of multiple endopeptidases for the same protein to break up long peptides allows obtaining overlapping fragments of peptide that can be used to produce final sequence. To be sequenced the peptide is adsorbed onto a solid surface.

Edman approach is known for impurities that commonly occur in the first round of degradation. Despite this fact, it remains a valuable tool for characterizing a protein's N-terminus and is still the most reliable, and accurate protein and peptide sequencing method that provides an unambiguous sequence read and is widely used for quality control purposes.

Another method for peptide end-group analysis was developed by Fred Sanger (Ryle *et al.*, 1955) who was awarded the 1958 Nobel Prize for Chemistry for sequencing insulin that was performed with the developed approach. Sanger method uses chemical derivatives to selectively label the N-terminus of a protein with the yellow dye fluorodinitrobenzene, followed by hydrolysis, and electrophoretic or chromatographic separation of the labeled N-terminal amino acid residue. Multiple rounds of partial protein hydrolysis, fractionation, and terminal amino acid determination lead to complete sequence of a protein.There is no robust method to sequence a protein or peptide from its C-terminus because carboxypeptidases – enzymes that are used as part of these methods – can remove only specific individual C-terminal amino acids (Bergman *et al.*, 2003).

Reconstructing Proteins From Mass-Spectra

Mass spectrometry (MS) methods are now the most widely used for protein sequencing and identification. Mass spectrometry is an analytical method that measures mass-to-charge ratio for ions in gas phase. MS breaks up peptides into individual amino acids using electric current, collects the released amino acids in mass spectrometer detector to be individually identified by their unique mass. The development of MS technology allowed to sequence the entire set of proteins of a living organism and thus became the trigger for the emergence of proteomics. (Tran *et al.*, 2017).

There are currently two major types of MS approaches to protein sequencing (**Fig. 17**): the *bottom-up* approach for the analysis of peptide mixtures from digested proteins and *top-down* mass spectrometry that is used to analyze intact proteins.

The bottom-up approach starts from the proteolytic digestion of proteins that creates complex peptide samples, which are then used in combination with high-throughput liquid chromatography (LC) and tandem MS* (LC-MS/MS; LC–MALDI MS/MS,

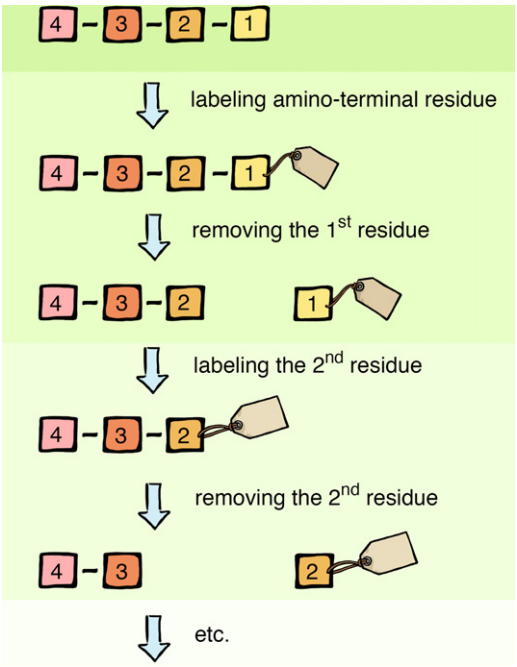

Fig. 16 Edman degradation procedure.

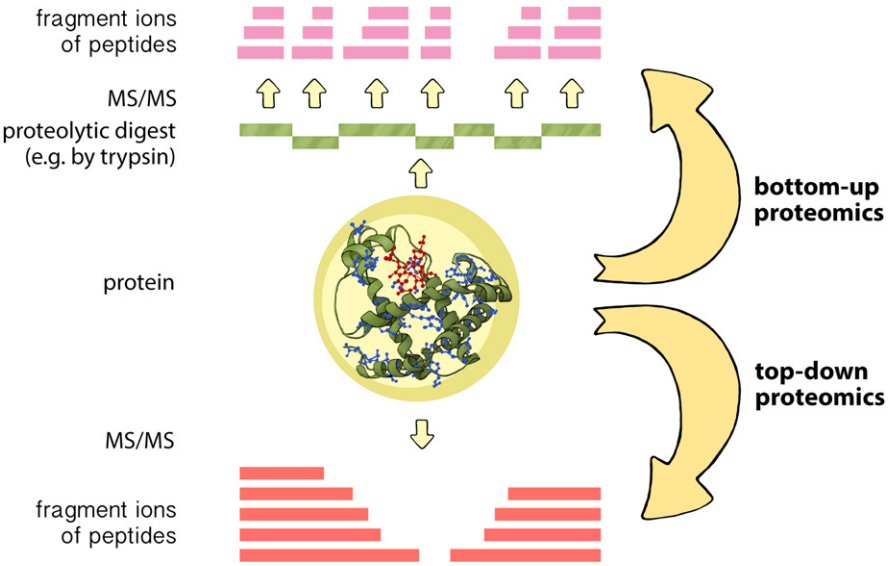

Fig. 17 MS approaches to protein sequencing.

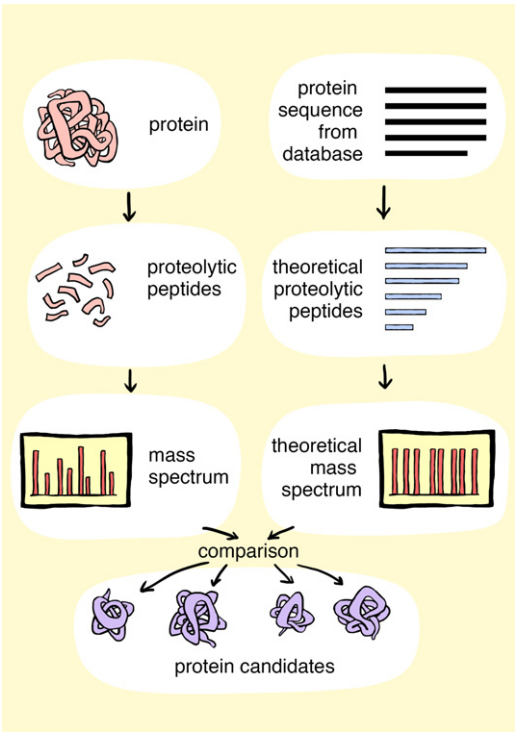

Fig. 18 Theoretical and experimental spectra for the peptide sequence.

MALDI TOF/TOF) to enable large-scale protein characterization in proteomics (Fernández-Puente *et al.*, 2014). This approach should not be applied for small proteins, as they have less cleavage sites, which will lead to an insufficient amount of peptides.

The top-down MS uses liquid chromatography or 2-D gel electrophoresis to separate intact proteins from complex samples. This approach is successfully used to characterize the structures of large proteins and to detect post-translational modifications (Aken *et al.*, 2016).

To analyze an experimental MS spectrum it needs to be compared with theoretical spectra for each of the peptides in a database (see "Relevant Websites" section). Theoretical peptides with the best fit help reconstruct the sequence of the experimental peptide (**Fig. 18**). Numerous post-translational modifications of proteins make them hard to be identified in databases and create a challenging computational problem.

The main goal of peptide sequencing is to reconstruct the amino acid sequence of a peptide. This goal can be achieved by using data of MS/MS spectrum and the peptide mass. Each tandem mass spectrum covers only a short part of the protein and to reconstruct the primary structure one has to assemble tandem mass spectra of overlapping peptides generated from multiple proteolytic digestions of the protein. Sounds like a DNA assembly task. Doesn't it? Here too we are trying to reconstruct a long sequence from many small fragments. In the case of protein assembly this approach is called top-down MS. To continue the analogy with genomic assembly, we should mention that top-down tandem mass spectra rarely provide full coverage of a protein. Bottom-up tandem mass spectra are used to increase sequence coverage (Liu *et al.*, 2014).

Experiments using MS technologies produce huge amount of very complex data and require specialised software for their analysis (for example: PepNovo, PEAKS, NovoHMM, MSNovo, pNovo, UniNovo, Novor and DeepNovo (Tran *et al.*, 2017)). Noisy and ambiguous MS/MS spectra cause additional computational challenges and require improved algorithms.

To find out more about the current tools that help retrieve long sequence fragments of the target proteins from sets of bottom-up and top-down MS/MS spectra see publications in further readings.

Mapping Proteins to Genome

Proteomics is a science that studies the protein composition of biological objects, as well as the modifications and the structural and functional properties of proteins. The uniqueness of the mass spectrometry pictures of products produced by proteolytic protein hydrolysis for each protein is used to analyze them against theoretical mass spectrum collected in databases.

One of the main goals when studying protein is to match the protein to the gene its coding. Knowledge of the mass of the whole protein does not provide reliable information due to the insufficient accuracy of the measurement or the presence of modifications. Therefore, researches use a mass spectrometry peptide map of a protein that consists of mass values of peptides obtained as a result of highly specific hydrolysis (chemical or enzymatic) of a protein. Such sets are called "MS-peptide fingerprints".

The main approach of protein identification by mass-spectrometry is to compare the real, experimentally obtained mass spectrum with the theoretical ones from the database that correspond to the known proteins.

Existing algorithms and corresponding programs that allow the identification of proteins by their mass spectra can be divided into three main groups:

1. Programs using the protein proteolytic peptide fingerprint (MASCOT (Perkins *et al.*, 1999). ProFound (Zhang and Chait, 2000)),
2. Programs that work with the peptide fingerprint and MS/MS spectra (MASCOT, MS-Fit (see "Relevant Websites" section),
3. Programs that work with the MS/MS spectra only (SEQUEST (see "Relevant Websites" section), PepFrag (see "Relevant Websites" section), MS-Tag (see "Relevant Websites" section), Sherpa (Taylor *et al.*, 1996).

All programs mentioned above use publicly available databases of amino acid sequences of proteins or nucleotide sequences of genes. Some of the most popular programs that contain protein sequences are: UniProt, a powerful database of annotated proteins (see "Relevant Websites" section); UniProtKB (see "Relevant Websites" section), a database that consists of Swiss-Prot (a collection of manually annotated and curated sequences) and TrEMBL (automatically annotated protein sequences translated from the nucleic acid sequences stored in EMBL) (Bairoch and Apweiler, 2000); UniRef, which contains sequence clusters for fast sequence similarity searches (see "Relevant Websites" section) and UniParc, a sequence archive of sequences and their identifiers (see "Relevant Websites" section); Protein Sequence Database (PSD) in the Protein Identification Resource (PIR, Barker *et al.*, 1999), containing annotated protein sequences; NCBI nr protein database, including entries from the non-redundant GenBank translations (see "Relevant Websites" section), UniProt, PIR (Barker *et al.*, 1999), Protein Research Foundation (PRF) in Japan (see "Relevant Websites" section), and the Protein Data Bank (PDB, see "Relevant Websites" section). Only entries with absolutely identical sequences are merged (Xu, 2004). Most of these databases also provide a sequence search tools.

Protein-Protein Alignment

Pairwise sequence alignment is a fundamental component of many bioinformatics problems. It is extremely useful in structural, functional, and evolutionary analyses of sequences since it provides information about the level of similarity between two sequences.

The overall goal of pairwise sequence alignment is to find the best "pairing" of two sequences of DNA, RNA, or protein, maximizing the number of similar characters. To achieve this, one sequence needs to be spread relative to the other to find the positions where maximal correspondence can be found. There are two main alignment strategies that are most oftenly used: global alignment and local alignment.

In *global alignment*, the two sequences to be aligned are assumed to be generally similar over the entirety of their length. Alignment is carried out from the start and till the end of both sequences to find the best possible alignment across the entire length between the two sequences. This method is more applicable for aligning two closely related sequences of (roughly) the same length. Note that this method is not suitable for identifying the highly similar local regions between the two sequences.

Local alignment, on the other hand, does not assume that the two sequences in question are similar over their entire length. In fact, it only determines regions with the highest level of similarity between the two sequences and aligns only these regions without taking into account the alignment of the rest of the sequences. This approach can be used for aligning more divergent sequences with the goal of searching for conserved patterns in DNA or protein sequences.

The way the alignment algorithms (both global and local) are implemented is fundamentally similar, with the exception of the optimization strategy used in aligning similar residues. Typically the alignment is performed by means of *dynamic programming*.

Dynamic programming is a method that determines the optimal alignment by matching two sequences against each other to identify for all possible pairs of matching characters between them. During this process a two-dimensional matrix is created and the two sequences to be compared are both laid out of the two axes. The symbols are then matched in according to the particular *scoring matrix*. Alignment scores are calculated one row at a time. The process starts with the first row of one sequence, which is used to scan through the entire length of the other sequence. The matching scores are calculated. The scanning of the second row takes into account the scores previously obtained during the first round. The best score is put into the bottom right corner of an intermediate matrix. This process is iterated until values for all the cells are filled. The scores are thus accumulated along a diagonal going from the upper left corner to the lower right corner. Once the matrix with the scores has been computed, the next step is to find the path that represents the optimal alignment. This is done by tracing back through the matrix in reverse order from the lower right-hand corner toward the origin of the matrix in the upper left-hand corner. The best matching path is the one that has the maximum total score. If two or more paths reach the same highest score, one is chosen arbitrarily to represent the best alignment. The path can also move horizontally or vertically at a certain point, which corresponds to the introduction of a gap or an insertion or deletion for one of the two sequences.

Gap penalties

Alignment between sequences often involves adding *gaps* that represent insertions and deletions. Note that special care should be taken when allowing gaps within the scoring scheme: if the gap penalty values are set too low, the gaps can become too numerous to allow even non-related sequences to be matched up with high similarity scores. If the penalty values are set too high, gaps may become too difficult to appear, which would stand in the way of the creation of a reasonable alignment.

Another factor to consider is the cost difference between opening a gap and extending an existing gap. Normally a gap opening should have a much higher penalty than a gap extension. This is based on the rationale that if insertions and deletions ever occur, several adjacent residues are likely to have been inserted or deleted together. These differential gap penalties are also referred to as affine gap penalties. Under normal circumstances separate gap penalty values are assigned for introducing and extending gaps. For example, one may use a $-12/-1$ scheme, in which the gap opening penalty is equal to -12 and the gap extension penalty is given the value of -1. The total gap penalty W is a linear function of gap length, which is calculated as $W=\gamma+\delta\times(l-1)$, where γ is the gap opening penalty, δ is the gap extension penalty, and l is the length of the gap (Gotoh, 1982). Besides the affine gap penalty, a constant gap penalty is sometimes also used, which assigns the same score for each gap position regardless whether it is an opening or an extension. In addition to that, end gaps can be allowed to be free (their inclusion will have zero penalty) to avoid getting unrealistic alignments.

Dynamic programming for global alignment

The classical global pairwise alignment algorithm using dynamic programming is known as the Needleman – Wunsch algorithm (Needleman and Wunsch, 1970). The aim this algorithm is to identify the optimal alignment is obtained over the entire lengths of the two sequences. It must extend from the beginning to the end of both sequences to achieve the highest total score. In other words, the alignment path has to go from the bottom right corner of the matrix to the top left corner. The possible drawback of focusing on trying to get a maximal score for the full-length sequence alignment is the risk of missing the best local similarities, which means that this strategy is only suitable for aligning two closely related sequences that are of the same length.

Dynamic programming for local alignment

In a regular sequence alignment, the level of divergence between the two sequences is not known in advance. The lengths of the two sequences may also differ. In cases such as these, it is of more value to identify the regional similarities between the two sequences than to try finding a match that would include all of the symbols. The dynamic programming approach to local alignment is known as Smith – Waterman algorithm (Smith and Waterman, 1981). In this algorithm, positive scores are assigned to matching symbols and zeros to mismatches. No negative scores are used. Occasionally, several optimally aligned segments with best scores are obtained. As in the global alignment, the final result is influenced by the choice of the scoring systems used. The goal of a local alignment is to get the highest alignment score locally, which may be at the expense of the highest possible overall score for a full-length alignment. This approach may be suitable for aligning divergent sequences or sequences with multiple domains that may be of different origins.

Scoring matrices

In the dynamic programming approach the alignment procedure has to make use of a scoring system, which represents a set of values used for the purpose of quantifying the likelihood of one symbol being substituted by another in an alignment. The scoring system is called a *substitution matrix* and is derived from the statistical analysis of symbol substitution data from sets of reliable alignments of highly related sequences. Scoring matrices for nucleotide sequences are relatively simple: a positive value or high score is given for a match and a negative value or low score for a mismatch. This assignment is based on the assumption that the frequencies of mutation are equal for all bases. However, this assumption may not be realistic; typically transitions (substitutions between purines and purines or between pyrimidines and pyrimidines) occur more frequently than transversions (substitutions

between purines and pyrimidines). Therefore, a more sophisticated statistical model with different probability values to reflect the two types of mutations is needed.

Scoring matrices for amino acids are much more complicated because the scoring systems have to reflect both the properties of amino acid residues, as well as the likelihood of certain residues being substituted among the homologous sequences. Certain amino acids with similar chemical properties can be more easily substituted than those having vastly different characteristics. Substitutions among similar residues are likely to preserve the essential functional and structural features. Substitutions between residues of different chemical properties, however, are more likely to cause changes (and even disruptions) to the structure and function. Such type of substitution is less likely to be favored by evolution since it is more likely to yield nonfunctional proteins.

For example, phenylalanine, tyrosine, and tryptophan all share aromatic ring structures. Because of their chemical similarities, they can be easily substituted for one another without changing the regular function and structure of the protein. Similarly, arginine, lysine, and histidine are all large basic residues and there is a high probability of them being freely interchanged. The hydrophobic residue group includes methionine, isoleucine, leucine, and valine. Small and polar residues include serine, threonine, and cysteine. Residues within these groups have high a likelihoods of being substituted for each other. However, cysteine contains a sulfhydryl group that plays a role in metal binding, as well as active site, and disulfide bond formation. Substitution of cysteine with other residues therefore often reduces the enzymatic activity or destabilizes the protein structure. This residue is thus very rarely substituted. Small and nonpolar residues such as glycine and proline are also unique since their presence often disrupts regular protein secondary structures and therefore the substitutions with these residues do not frequently occur.

Amino acid scoring matrices

Amino acid substitution matrices are 20×20 and have been developed to reflect the likelihood of residue substitutions. There are two possible types of amino acid substitution matrices: one type is based on the interchangeability of the genetic code or amino acid properties, and the other is derived from empirical studies of amino acid substitutions. Although the two different approaches have a certain degree of overlap, surprisingly, the first approach, based on the genetic code or the chemical features of amino acids, has been shown to be less accurate than the second approach, which is based on the comparison of actual amino acid substitutions among related proteins.

The empirical amino acid scoring matrices, which include PAM (Dayhoff *et al.*, 1978) and BLOSUM (Henikoff and Henikoff, 1992) matrices, are derived from actual alignments of highly similar sequences. By analyzing the frequencies of amino acid substitutions in these alignments, a scoring system can be developed by giving a higher score to more likely substitutions and correspondingly lower scores for rare ones.

Statistical significance of a sequence alignment

Once a sequence alignment showing some degree of similarity is achieved, it is important to double check whether the observed sequence alignment is indeed sound or could have been occurred by pure chance. A truly statistically significant sequence alignment might be able to provide some results connected with the homology between the sequences being aligned.

To solve this problem one must make use of a procedure designed to assist with the estimation of the probabilistic distribution of the alignment scores of the two unrelated sequences of the same length. By calculating alignment scores of a large number of unrelated pairs of sequences, a distribution model of the randomized sequence scores can be derived. It turns out that the distribution in question is the so-called Gumbel distribution (or Generalized extreme value distribution of type I) for which a mathematical expression is available. This means that given a sequence similarity value the statistical significance can be accurately estimated (Altschul *et al.*, 1990).

Multiple sequence alignment

A natural extension of pairwise alignment is multiple sequence alignment, which aims to align multiple related sequences to achieve their optimal matching. Usually such related sequences are identified through a database similarity searching. As the process generates multiple matching sequence pairs, it is often necessary to somehow turn separate pairwise alignments into a single alignment, which arranges sequences in such a way that evolutionarily equivalent positions across all sequences are matched.

The multiple sequence alignment has the unique advantage of being able to reveal more biological information than many of the pairwise alignments. For example, it allows to identify conserved sequence patterns and motifs in the whole sequence family, which would not be easy to detect when comparing only two of the sequences. Many conserved and functionally critical amino acid residues can be identified in a protein multiple alignment. Multiple sequence alignment is also an essential prerequisite to carrying out phylogenetic analysis of sequence families and prediction of protein secondary and tertiary structures.

It is theoretically possible to use dynamic programming to align any number of sequences in the similar way as for pairwise alignment. However, it is worth keeping in mind that the amount of computing time and memory required will increase exponentially the more sequences are being compared. As a consequence, full dynamic programming usually cannot be applied for datasets of more than 10–25 sequences. In practice, heuristic approaches are most often used that provide suboptimal, but still extremely useful solutions. There are different approaches that could be used to perform multiple sequence

alignment. For the sake of the simplicity we will describe only one, namely, the *progressive alignment* approach (Feng and Doolittle, 1996).

Progressive alignment depends on the stepwise assembly of multiple alignment and is heuristic in nature. It speeds up the alignment of multiple sequences through a multistep process. It first conducts pairwise alignments for each possible pair of sequences using the Needleman–Wunsch global alignment method and records these similarity scores from the pairwise comparisons. The scores can be, for example, a percent identity or similarity scores based on a particular substitution matrix. Both do correlate with the evolutionary distances between the sequences. The scores are then converted into evolutionary distances to generate a distance matrix for all the sequences involved. A simple phylogenetic analysis is then performed based on the distance matrix to group sequences based on pairwise distance scores. As a result, a phylogenetic tree is generated using the neighbor-joining method. The tree reflects the evolutionary proximity among all the sequences.

Despite the fact that the resulting tree will only be an approximation, it still can be used as a guide for directing realignment of the sequences. For that reason, it is often referred to as a *guide tree*. According to the guide tree, the two most closely related sequences are first re-aligned using the Needleman – Wunsch algorithm. To align additional sequences, the two already aligned sequences are converted to a consensus sequence with fixed gap positions. The consensus is then treated as a single sequence in the subsequent step. In the following step, the next closest sequence based on the guide tree is aligned with the consensus sequence using dynamic programming. More distant sequences or sequence profiles are subsequently added one at a time in accordance with their relative positions on the guide tree. After realignment with a new sequence using dynamic programming, a new consensus is derived, which is then used for the next round of alignment. The process is repeated until all the sequences are aligned.

See sections "Further readings" and "Software" for the list and details of software tools and web portals used in nucleotide and protein sequence alignment.

Metabolic Pathways Analysis

The functions of about half of the genes in a bacterium can be reliably insinuated by comparing them to reference databases. This approach allows to compile an overall approximation of the metabolic map of the organism, which describes the substances the bacterium is able to synthesize on its own, the substances it needs to absorb from its environment, what constitutes its source of energy etc. These maps can then be further refined and cleaned of any possible errors by adjusting the similarity and other criteria.

The results of these studies can be used in such fields as biotechnology, agronomy, pharmaceutics and medicine. The analysis of the metabolic pathways of commercial strains helps optimize the release of the intended product of interest. For example, enzymes that trigger key reactions in pathogens can be used as potential targets for new types of antibiotics.

Metabolic Pathways

Series of interactions between molecules in a cell that lead to assembly of new molecules in a cell, turn genes on and off, maintain and control the flow of information, energy and biochemical compounds are called biological pathways.

The most common types of biological pathways are involved in the metabolism of a cell (metabolic pathways), gene expression regulation (genetic pathways) and signal transmission (signal transduction pathways).

Metabolism can be defined as a series of chemical transformations of matter and energy taken from the environment aimed to maintain an organism's life. Metabolic pathways in their turn are a series of biochemical reactions that convert substrates into products.

Metabolism can be conditionally divided into two opposite streams of transformations: anabolism and catabolism.

Anabolism refers to all processes involved in the creation of new substances, cells and tissues in the body. Examples of anabolism include synthesis of proteins and hormones, creation of new cells, accumulation of fats, and creation of new muscle fibers. The sum of all processes in the body that lead to the creation of any new substances and tissues is called anabolism.

On the flip side we have catabolism, which encompasses the processes involved in splitting complex substances into simpler ones, as well as decay of old cell parts and tissues of the body. An example of catabolism is the breakdown of fats and carbohydrates to produce energy, which in turn can be used for the synthesis of the substances of use to the organism, the creation of cells and the overall renewal of the body. It follows that both anabolism and catabolism are very closely interrelated to each other. The most important metabolic pathways in humans are: glycolysis, citric acid cycle (Krebs' cycle), oxidative phosphorylation, pentose phosphate pathway, urea cycle, fatty acid β-oxidation, gluconeogenesis.

The most convenient way to investigate metabolic pathways is by analyzing metabolic models. They contain all the information about the reactions that can occur in a cell, how the reactions are related to each other, the activity of which genes can regulate the reactions, which of the reactions relate to standard metabolic pathways, and which are the most important, how they are combined into metabolic pathways and how they interact with each other.

Regulation of metabolic pathways is achieved through complex interactions of a large number of factors. As a result of the proper functioning of all metabolic pathways, the cell can maintain a stable homeostasis and properly respond to changes in the external environment. The availability of a large volume of genomic information (assembled genomes) facilitates the study of metabolism regulation. Based on this information, whole genome metabolic models generalizing the available ideas about

metabolism and its regulation are constructed. Such models include a list of all possible reactions, a set of rules for the regulation of individual reactions, thermodynamic limitations that determine possible directions of reactions.

Methods of metabolic models analysis can be subdivided into two groups: qualitative (topological analysis, flux balance analysis, structural kinetic models) and quantitative (structural kinetic models, kinetic models) (Tomar and De, 2014). A number of methods also use information about the atomic structure of metabolites. They allow to analyze the transition from individual metabolites to biochemical reactions.

The development of bioinformatics methods to interpret data led to the development of metabolic pathway analysis methodologies, including structural and stoichiometric analysis, metabolic flux analysis, metabolic control analysis, and several kinetic modeling based analysis. The article written by Dr. Namrata Tomar and Dr. Rajat K. (Tomar and De, 2014), provides the comprehensive survey on the existing metabolic pathway analysis methodologies.

A review of the most common approaches for the construction of computational models of metabolic systems can be found in (Rice *et al.*, 2008).

Metabolic Databases

Metabolic models can be obtained from different sources.

The KEGG database contains a large amount of information on metabolic reactions. It includes several separate databases, such as: KEGG REACTION – with information on biochemical reactions; KEGG COMPOUND and KEGG GLYCAN – about metabolites; KEGG ENZYME – about enzymes; KEGG GENES – about the genes; KEGG PATHWAY and KEGG MODULES – about metabolic pathways.

This database was developed for bioinformatics analysis in genomics, metagenomics, metabolomics, modeling and simulation in systems biology, and translational research in drug development. It contains about 4000 organisms, including 123 animals. The database is curated and requires a paid license for full access. Some information is available for free via the web interface and the program interface (Kanehisa *et al.*, 2008, 2012, 2014, 2016, 2017). One can analyze new sequenses (DNA or protein) against metadata stored in the KEGG database by mapping them to the known pathways, models, orthologs and so on (See "Relevant Websites" Section) by linking new genomes to well curated known pathways.

The BioCyc database is similar to the KEGG database. It contains about 7600 organism-specific bases. Some of them are manually curated, including the base containing human data. Another ones, including the mouse DB, are partially supervised.

The metabolic models are also stored in the EBI BioModels Database. This free to use DB contains about 2500 whole genome metabolic models in SBML format, but all of them are generated automatically from KEGG and MetaCyc databases.

The Reactome database is another free database with information on metabolic and molecular pathways. This carefully curated database focuses mainly on human biology. Data from the database can be downloaded in SBML and other formats.

There are several end to end data pipelines dedicated to help with the complex task of metabolic pathways analysis (Poskar *et al.*, 2014; Abubucker *et al.*, 2016).

Gene Ontology

Gene ontology (GO) is a major bioinformatics initiative to unify the representation of gene and gene product attributes across all species (see "Relevant Websites" section).

Because the complexity of biological systems and the sizes of the datasets that need to be analyzed continue to grow, biomedical research is becoming increasingly dependent on knowledge stored in computable form. The Gene Ontology (GO) project is a collaborative effort to address the need for consistent descriptions of gene functions and gene products across different databases. The GO knowledgebase is composed of three primary components:

1. *Gene Ontology*, which provides the logical structure of the biological functions ('terms'). It is structured as a directed acyclic graph where each term has defined relationships to one or more other terms in the same domain and sometimes to other domains. The ontology includes data on: *cellular components*; *molecular functions* and *biological processes* (see "Relevant Websites" section).
   Molecular function corresponds to activities that can be performed by the individual gene products or by the assembled complexes of gene products.
   A biological process in GO is not the same as a pathway since GO does not include information about the dynamics or the dependencies, both of which would be required to fully describe a pathway.
   The GO vocabulary is species-neutral, and includes terms applicable to prokaryotes and eukaryotes, single or multicellular organisms.
2. *GO annotations* are evidence-based statements relating a specific gene product (a protein, non-coding RNA, or macromolecular complex, which we referred as 'genes' for simplicity) to a specific ontology term.
3. *Tools* that facilitate the creation, maintenance and use of ontologies. Software being developed aims to help edit and perform logic based analysis of ontologies, provide web access to the ontology and its annotations, and analyze data using the information in the GO knowledgebase to support biomedical research.

Ontologies can be browsed using a range of web-based browsers (see "Relevant Websites" section).

The overall goal of GO is to create a comprehensive model of biological systems. Currently, the GO knowledgebase includes experimental findings from almost 140,000 published papers (see "Relevant Websites" section), represented by over 600,000 experimentally-supported GO annotations, on the basis of which an additional 6 million functional annotations for a diverse set of organisms spanning the tree of life can be inferred.

That said, it is worth mentioning that GO does not store gene sequences and is not a catalog of gene products. GO describes how gene products behave in a cellular/organism context. The use of GO terms by the collaborating databases provides controlled vocabularies that are structured to facilitate querying at different levels and also to allow annotators to assign properties to genes or gene products at different levels, depending on the depth of knowledge about the particular entity.

Conclusions

Due to the rapid evolution of new methodologies in molecular and computational biology evolve, the aim of this article was to only provide a general overview of sequence analysis at the DNA, RNA, and protein and metabolic pathways level.

We have attempted to briefly cover both the general approaches, and the most popular and powerful tools that are currently being used to analyze large amounts of data from labs around the world.

Acknowledgement

Authors are grateful to our colleague Elena Strelnikova for her help with **Figs. 16**, **17** and **18**. This work was supported by Russian Science Foundation (grant number 14–50–00069).

See also: Algorithms for Strings and Sequences: Multiple Alignment. Algorithms for Strings and Sequences: Pairwise Alignment. Natural Language Processing Approaches in Bioinformatics

References

Abubucker, S., Segata, N., Goll, J., *et al*., 2016. Metabolic reconstruction for metagenomic data and its application to the human microbiome. PLOS Comput. Biol. 8 (6), e1002358.
Acland, A., Agarwala, R., *et al*., 2014. Database resources of the National Center for Biotechnology Information. Nucleic Acids Res 42, D7–D17.
Aken, B.L., Ayling, S., Barrell, D., *et al*., 2016. The Ensembl gene annotation system. Database J. Biol. Databases Curation 2016, baw093.
Altschul, S.F., Gish, W., Miller, W., Myers, E.W., Lipman, D.J., 1990. Basic local alignment search tool. J. Mol. Biol. 215, 403–410.
Anders, S., Pyl, P.T., Huber, W., 2015. HTSeq – A python framework to work with high-throughput sequencing data. Bioinformatics 31 (2), 166–169.
Andrews S., 2010. FastQC: A quality control tool for high throughput sequence data. Available online at: http://www.bioinformatics.babraham.ac.uk/projects/fastqc.
Antipov, D., Korobeynikov, A., McLean, J.S., Pevzner, P.A., 2015. hybridSPAdes: An algorithm for hybrid assembly of short and long reads. Bioinformatics 32 (7), 1009–1015.
Azad, R.K., Borodovsky, M., 2004. Probabilistic methods of identifying genes in prokaryotic genomes: Connections to the HMM theory. Brief. Bioinform. 5, 118–130.
Barker, W.C., Garavelli, J.S., McGarvey, P.B., *et al*., 1999. The PIR-international protein sequence database. Nucleic Acids Res. 27 (1), 39–43.
Benson, D.A., Karsch-Mizrachi, I., Lipman, D.J., Ostell, J., Wheeler, D.L., 2004. GenBank: Update. Nucleic Acids Res. 32, D23–D26.
Berg, J., Tymoczko, J., Stryer, L., 2002. Biochemistry, 5th ed. New York: W.H. Freeman.
Bergman, T., Cederlund, E., Jörnvall, H., Fowler, E., 2003. Current protocols in protein science. (Chapter 11, Unit 11.8).
Bairoch, A., Apweiler, R., 2000. The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000. Nucleic Acids Res. 28 (1), 45–48.
Bolger, A.M., Lohse, M., Usadel, B., 2014. Trimmomatic: A flexible trimmer for illumina sequence data. Bioinformatics 30 (15), 2114–2120.
Bookstein, A., Kulyukin, V.A., Raita, T., 2002. Generalized hamming distance. Inform. Retr. 5, 353.
Bradnam, K.R., Fass, J.N., Alexandrov, A., *et al*., 2013. Assemblathon 2: Evaluating de novo methods of genome assembly in three vertebrate species. GigaScience 2 (1), 10.
Bray, N.L., Pimentel, H., Melsted, P., Pachter, L., 2016. Near-optimal probabilistic RNA-seq quantification. Nat. Biotechnol. 34 (5), 525–527.
Bresler, M.A., Sheehan, S., Chan, A.H., Song, Y.S., 2012. Telescoper: De novo assembly of highly repetitive regions. Bioinformatics 28 (18), i311–i317.
Brown, J.W.S., Echeverria, M., Qu, L.-H., *et al*., 2003. Plant snoRNA database. Nucleic Acids Res. 31, 432–435.
Burge, S.W., Daub, J., Eberhardt, R., *et al*., 2013. Rfam 11.0: 10 years of RNA families. Nucleic Acids Res. 41, D226–D232.
Bushnell, B., 2014. BBTools: A suite of fast, multithreadedbioinformatics tools designed for analysis of DNA and 1NA sequencedata. Available online at: https://jgi.doe.gov/data-and-tools/bbtools/.
Bushmanova, E., Antipov, D., Lapidus, A., Suvorov, V., Prjibelski, A.D., 2016. rnaQUAST: A quality assessment tool for de novo transcriptome assemblies. Bioinformatics 32 (14), 2210–2212.
Cingolani, P., Platts, A., Wang, L.L., *et al*., 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 6 (2), 80–92.
Danecek, P., Auton, A., Abecasis, G., *et al*., 2011. The variant call format and VCFtools. Bioinformatics 27 (15), 2156–2158.
Dayhoff, M.O., Schwartz, R.M., Orcutt, B.C., 1978. A model for evolutionary change in proteins. In: Dayhoff, Margaret O. (Ed.), Atlas of Protein Sequence and Structure, vol. 5. Washington DC: National Biochemical Research Foundation, pp. 345–352.
Delcher, A.L., Harmon, D., Kasif, S., White, O., Salzberg, S.L., 1999. Improved microbial gene identification with GLIMMER. Nucleic Acids Res. 27 (23), 4636–4641.
Dobin, A., Davis, C.A., Schlesinger, F., *et al*., 2013. STAR: Ultrafast universal RNA-seq aligner. Bioinformatics 29 (1), 15–21.
Dündar, F., Skrabanek, L., Zumbo, P., 2015. Applied Bioinformatics Core/Weill Cornell Medical College. Applied Bioinformatics Core/Weill Cornell Medical College. pp. 1–67.
Earl, D., Bradnam, K., John, J.S., *et al*., 2011. Assemblathon 1: A competitive assessment of de novo short read assembly methods. Genome Res. 21 (12), 2224–2241.
Edman, P., Högfeldt, E., Sillén, L.G., Kinell, P.-O., 1950. Method for determination of the amino acid sequence in peptides. Acta Chem. Scand. 4, 283–293.

Felsenstein, J., 1981. Evolutionary trees from DNA sequences: A maximum likelihood approach. J. Mol. Evol. 17 (6), 368–376.
Feng, D.-F., Doolittle, R.F., 1996. Doolittle progressive alignment of amino acid sequences and construction of phylogenetic trees from them. In: Proceedings of the Methods in Enzymology, 266, pp. 368–382. Academic Press.
Fernández-Puente, P., Mateos, J., Blanco, F.J., Ruiz-Romero, C., 2014. LC-MALDI-TOF/TOF for shotgun proteomics. Methods Mol. Biol. 2014 (1156), 27–38.
Ferragina, P., Manzini, G., Mäkinen, V., Navarro, G., 2004. An alphabet-friendly FM-index. In: Proceedings of the String Processing and Information Retrieval, p. 228. Berlin/Heidelberg: Springer.
Gotoh, O., 1982. An improved algorithm for matching biological sequences. J. Mol. Biol. 162, 705–708.
Grabherr, M.G., Haas, B.J., Yassour, M., *et al.*, 2011. Full-length transcriptome assembly from RNA-Seq data without a reference genome. Nat. Biotechnol. 29 (7), 644–652.
Gurevich, A., Saveliev, V., Vyahhi, N., Tesler, G., 2013. QUAST: Quality assessment tool for genome assemblies. Bioinformatics 29 (8), 1072–1075.
Hannenhalli, S., Pevzner, P.A., 1999. Transforming cabbage into turnip: Polynomial algorithm for sorting signed permutations by reversals. J. ACM (JACM) 46 (1), 1–27.
Heather, J.M., Chain, B., 2016. The sequence of sequencers: The history of sequencing DNA. Genomics 107 (1), 1–8. doi:10.1016/j.ygeno.2015.11.003.
Henikoff, S., Henikoff, J.G., 1992. Amino acid substitution matrices from protein blocks. PNAS 89, 10915–10919.
Hrdlickova, R., Toloue, M., Tian, B., 2017. RNA-Seq methods for transcriptome analysis. WIREs RNA 8 (1), doi:10.1002/wrna.1364. Jan.
Hunt, M., Kikuchi, T., Sanders, M., *et al.*, 2013. REAPR: A universal tool for genome assembly evaluation. Genome Biol. 14 (5), R47.
Kanehisa, M., Araki, M., Goto, S., *et al.*, 2008. KEGG for linking genomes to life and the environment. Nucleic Acids Res. 36, D480–D484.
Kanehisa, M., Goto, S., Sato, Y., Furumichi, M., Tanabe, M., 2012. KEGG for integration and interpretation of large-scale molecular datasets. Nucleic Acids Res. 40, D109–D114.
Kanehisa, M., Goto, S., Sato, Y., *et al.*, 2014. Data, information, knowledge and principle: Back to metabolism in KEGG. Nucleic Acids Res. 42, D199–D205.
Kanehisa, M., Sato, Y., Kawashima, M., Furumichi, M., Tanabe, M., 2016. KEGG as a reference resource for gene and protein annotation. Nucleic Acids Res. 44, D457–D462.
Kanehisa, M., Furumichi, M., Tanabe, M., Sato, Y., Morishima, K., 2017. KEGG: New perspectives on genomes, pathways, diseases and drugs. Nucleic Acids Res. 45, D353–D361.
Kapustin, Y., Souvorov, A., Tatusova, T., Lipman, D., 2008. Splign: Algorithms for computing spliced alignments with identification of paralogs. Biol. Direct 3 (1), 20.
Kent, W.J., 2002. BLAT – The BLAST-like alignment tool. Genome Res. 12 (4), 656–664.
Kent, W.J., Sugnet, C.W., Furey, T.S., *et al.*, 2002. The human genome browser at UCSC. Genome Res. 12 (6), 996–1006.
Kim, D., Salzberg, S.L., 2011. TopHat-Fusion: An algorithm for discovery of novel fusion transcripts. Genome Biol. 12 (8), R72.
Kim, D., Langmead, B., Salzberg, S.L., 2015. HISAT: A fast spliced aligner with low memory requirements. Nat. Methods 12 (4), 357–360.
Koren, S., Schatz, M.C., Walenz, B.P., *et al.*, 2012. Hybrid error correction and de novo assembly of single-molecule sequencing reads. Nat. Biotechnol. 30 (7), 693–700.
Koren, S., Walenz, B.P., Berlin, K., *et al.*, 2017. Canu: Scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. Genome Res. 27 (5), 722–736.
Kukurba, K.R., Montgomery, S.B., 2015. RNA Sequencing and Analysis. Cold Spring Harb Protoc. 11, 951–969.
Kumar, S., Tamura, K., Nei, M., 1994. MEGA: Molecular evolutionary genetics analysis software for microcomputers. Bioinformatics 10 (2), 189–191.
Lagesen, K., Hallin, P., Rødland, E.A., *et al.*, 2007. RNAmmer: Consistent and rapid annotation of ribosomal RNA genes. Nucleic Acids Res. 35, 3100–3108.
Langmead, B., Salzberg, S.L., 2012. Fast gapped-read alignment with Bowtie 2. Nat. Methods 9 (4), 357–359.
Lapidus, A.L., 2015. Genome Sequence Databases: Sequencing and Assembly. Reference Module in Biomedical Sciences. Oxford: Elsevier, Available at: http://www.sciencedirect.com/science/article/pii/B9780128012383024958.
Leung, H.C., Yiu, S.M., Chin, F.Y., 2015. IDBA-MTP: A hybrid metatranscriptomic assembler based on protein information. J. Comput. Biol. 22 (5), 367–376.
Levene, M.J., Korlach, J., Turner, S.W., *et al.*, 2003. Zero-mode waveguides for single-molecule analysis at high concentrations. Science 299, 682–686.
Li, B., Fillmore, N., Bai, Y., *et al.*, 2014. Evaluation of de novo transcriptome assemblies from RNA-Seq data. Genome Biol. 15 (12), 553.
Li, D., Liu, C.M., Luo, R., Sadakane, K., Lam, T.W., 2015. MEGAHIT: An ultra-fast single-node solution for large and complex metagenomics assembly via succinct de Bruijn graph. Bioinformatics 31 (10), 1674–1676.
Li, H., 2013. Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM. Available from: http://arxiv.org/abs/1303.3997.
Li, H., 2016. Minimap and miniasm: Fast mapping and de novo assembly for noisy long sequences. Bioinformatics 32 (14), 2103–2110.
Li, H., Durbin, R., 2009. Fast and accurate short read alignment with Burrows – Wheeler transform. Bioinformatics 25 (14), 1754–1760.
Li, H., Handsaker, B., Wysoker, A., *et al.*, 2009. The sequence alignment/map format and SAMtools. Bioinformatics 25 (16), 2078–2079.
Liu, X., Dekker, L.J., Wu, S., *et al.*, 2014. De novo protein sequencing by combining top-down and bottom-up tandem mass spectra. J. Proteome Res. 13 (7), 3241–3248.
Lizardi, P.M., 2000. Multiple displacement amplification. Yale University, U.S. Patent 6,124,120.
Love, M.I., Huber, W., Anders, S., 2014. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biol. 15 (12), 550.
Lowe, T.M., Eddy, S.R., 1997. tRNAscan-SE: A program for improved detection of transfer RNA genes in genomic sequence. Nucleic Acids Res. 25, 955–964.
Lukashin, A.V., Borodovsky, M., 1998. GeneMark.hmm: New solutions for gene finding. Nucleic Acids Res. 26 (4), 1107–1115.
Luo, R., Liu, B., Xie, Y., *et al.*, 2012. SOAPdenovo2: An empirically improved memory-efficient short-read de novo assembler. Gigascience 1 (1), 18.
Magoc, T., Pabinger, S., Canzar, S., *et al.*, 2013. GAGE-B: An evaluation of genome assemblers for bacterial organisms. Bioinformatics 29 (14), 1718–1725.
Margulies, M., Egholm, M., Altman, W.E., *et al.*, 2005. Genome sequencing in microfabricated high-density picolitre reactors. Nature 437, 376–380.
Martin, M., 2011. Cutadapt removes adapter sequences from high-throughput sequencing reads. EMBnet J. 17 (1), 10.
McKenna, A., Hanna, M., Banks, E., *et al.*, 2010. The genome analysis toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data. Genome Res. 20 (9), 1297–1303.
Milne, I., Bayer, M., Cardle, L., *et al.*, 2009. Tablet – Next generation sequence assembly visualization. Bioinformatics 26 (3), 401–402.
Morgulis, A., Coulouris, G., Raytselis, Y., *et al.*, 2008. Database indexing for production MegaBLAST searches. Bioinformatics 24 (16), 1757–1764.
Myers, E., 2005. The fragment assembly string graph. Bioinformatics 21 (S2), ii79–ii85.
Myers, E.W., Sutton, G.G., Delcher, A.L., *et al.*, 2000. A whole-genome assembly of Drosophila. Science 287 (5461), 2196–2204.
Needleman, S.B., Wunsch, C.D., 1970. A general method applicable to the search for similarities in the amino acid sequence of two proteins. J. Mol. Biol. 48 (3), 443–453.
Niall, H.D., 1973. Automated Edman degradation: The protein sequenator. Methods Enzymol. 27, 942–1010.
Nurk, S., Bankevich, A., Antipov, D., *et al.*, 2013. Assembling single-cell genomes and mini-metagenomes from chimeric MDA products. J. Comput. Biol. 20 (10), 714–737.
Nurk, S., Meleshko, D., Korobeynikov, A., Pevzner, P.A., 2017. metaSPAdes: A new versatile metagenomic assembler. Genome Res. 27 (5), 824–834.
O'Connell, J., Schulz-Trieglaff, O., Carlson, E., *et al.*, 2015. NxTrim: Optimized trimming of illumina mate pair reads. Bioinformatics 31 (12), 2035–2037.
Okonechnikov, K., Conesa, A., García-Alcalde, F., 2015. Qualimap 2: Advanced multi-sample quality control for high-throughput sequencing data. Bioinformatics 32 (2), 292–294.
Peng, Y., Leung, H.C., Yiu, S.M., Chin, F.Y., 2012. IDBA-UD: A de novo assembler for single-cell and metagenomic sequencing data with highly uneven depth. Bioinformatics 28 (11), 1420–1428.
Perkins, D.N., Pappin, D.J.C., Creasy, D.M., Cottrell, J.S., 1999. Probability-based protein identification by searching sequence databases using mass spectrometry data. Electrophoresis 20, 3551–3567.

Pevzner, P.A., Tang, H., Waterman, M.S., 2001. An Eulerian path approach to DNA fragment assembly. Proc. Natl. Acad. Sci. 98 (17), 9748–9753.
Poskar, C.H., Huege, J., Krach, C., Shachar-Hill, Y., Junker, B.H., 2014. High-throughput data pipelines for metabolic flux analysis in plants. Methods Mol. Biol. 1090, 223–246.
Prjibelski, A.D., Vasilinetc, I., Bankevich, A., *et al*., 2014. ExSPAnder: A universal repeat resolver for DNA fragment assembly. Bioinformatics 30 (12), i293–i301.
Rice, S.A., Steuer, R., Junker, B.H., 2008. Computational models of Metabolism: Stability and regulation in metabolic. Networks. doi:10.1002/9780470475935.ch3.
Robertson, G., Schein, J., Chiu, R., *et al*., 2010. De novo assembly and analysis of RNA-seq data. Nat. Methods 7 (11), 909–912.
Robinson, M.D., McCarthy, D.J., Smyth, G.K., 2010. edgeR: A bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26 (1), 139–140.
Rothberg, J.M., Hinz, W., Rearick, T.M., *et al*., 2011. An integrated semiconductor device enabling non-optical genome sequencing. Nature 475, 348–352.
Ryle, A.P., Sanger, F., Smith, L.F., Kitai, R., 1955. The disulphide bonds of insulin. Biochem. J. 60 (4), 541–556.
Salzberg, S., Delcher, A., Kasif, S., White, O., 1998. Microbial gene identification using interpolated Markov models. Nucleic Acids Res. 26 (2), 544–548.
Salzberg, S.L., Phillippy, A.M., Zimin, A., *et al*., 2012. GAGE: A critical evaluation of genome assemblies and assembly algorithms. Genome Res. 22 (3), 557–567.
Shendure, J., Porreca, G.J., Reppas, N.B., *et al*., 2005. Accurate multiplex polony sequencing of an evolved bacterial genome. Science 309, 1728–1732.
Simão, F.A., Waterhouse, R.M., Ioannidis, P., Kriventseva, E.V., Zdobnov, E.M., 2015. BUSCO: Assessing genome assembly and annotation completeness with single-copy orthologs. Bioinformatics 31 (19), 3210–3212.
Simpson, J., Durbin, R., 2012. Efficient de novo assembly of large genomes using compressed data structures. Genome Res. 22, 549–556.
Simpson, J.T., Wong, K., Jackman, S.D., *et al*., 2009. ABySS: A parallel assembler for short read sequence data. Genome Res. 19 (6), 1117–1123.
Smith, T.F., Waterman, M.S., 1981. Identification of common molecular subsequences. J. Mol. Biol. 147 (1), 195–197.
Smith-Unna, R., Boursnell, C., Patro, R., Hibberd, J.M., Kelly, S., 2016. TransRate: Reference-free quality assessment of de novo transcriptome assemblies. Genome Res. 26 (8), 1134–1144.
Sović, I., Šikić, M., Wilm, A., *et al*., 2016. Fast and sensitive mapping of nanopore sequencing reads with GraphMap. Nat. Commun. 7, 11307.
Tang, S., Lomsadze, A., Borodovsky, M., 2015. Identification of protein coding regions in RNA transcripts. Nucleic Acids Res. 43 (12), e78.
Taylor, J.A., Walsh, K.A., Johnson, R.S., 1996. Sherpa: A macintosh-based expert system for the interpretation of electrospray ionization LC/MS and MS/MS data from protein digests. Rapid Commun. Mass Spectrom. 10, 679–687.
Thorvaldsdóttir, H., Robinson, J.T., Mesirov, J.P., 2013. Integrative Genomics Viewer (IGV): High-performance genomics data visualization and exploration. Brief. Bioinform. 14 (2), 178–192.
Tomar, N., De, R.K., 2014. A comprehensive view on metabolic pathway analysis methodologies. Curr. Bioinform. 9, 295–305.
Tran, N.H., Zhang, X., Xin, L., Shan, B., Li, V., 2017. De novo peptide sequencing by deep learning. Proc. Natl. Acad. Sci. 114 (31), 8247–8252.
Trapnell, C., Pachter, L., Salzberg, S.L., 2009. TopHat: Discovering splice junctions with RNA-Seq. Bioinformatics 25 (9), 1105–1111.
Valouev, A., Ichikawa, J., Tonthat, J., *et al*., 2008. A high-resolution, nucleosome position map of C. elegans reveals a lack of universal sequence-dictated positioning. Genome Res. 18, 1051–1063.
Vasilinetc, I., Prjibelski, A.D., Gurevich, A., Korobeynikov, A., Pevzner, P.A., 2015. Assembling short reads from jumping libraries with large insert sizes. Bioinformatics 31 (20), 3262–3268.
Wang, B.-B., Brendel, V., 2004. The ASRG database: Identification and survey of *Arabidopsis thaliana* genes involved in pre-mRNA splicing. Genome Biol. 5, R102.
Wang, L., Wang, S., Li, W., 2012. RSeQC: Quality control of RNA-seq experiments. Bioinformatics 28 (16), 2184–2185.
Wang, Z., Chen, Y., Li, Y., 2004. A brief review of computational gene prediction methods. Genom. Prot. Bioinform. 4, 216–221.
Wick, R.R., Judd, L.M., Gorrie, C.L., Holt, K.E., 2017. Unicycler: Resolving bacterial genome assemblies from short and long sequencing reads. PLOS Comput. Biol. 13 (6), e1005595.
Woyke, T., Tighe, D., Mavromatis, K., *et al*., 2010. One Bacterial Cell, One Complete Genome. PLoS ONE 5 (4), e10314.
Wu, T.D., Watanabe, C.K., 2005. GMAP: A genomic mapping and alignment program for mRNA and EST sequences. Bioinformatics 21 (9), 1859–1875.
Xie, Y., Wu, G., Tang, J., *et al*., 2014. SOAPdenovo-Trans: De novo transcriptome assembly with short RNA-Seq reads. Bioinformatics 30 (12), 1660–1666.
Xu, D., 2004. Protein Databases on the Internet. Curr. Protoc. Mol. Biol.. 19.4), (CHAPTER: Unit–19.4. *PMC*. Web. 2 Nov. 2017).
Ye, Y., Tang, H., 2015. Utilizing de Bruijn graph of metagenome assembly for metatranscriptome analysis. Bioinformatics 32 (7), 1001–1008.
Zhang, W., Chait, B.T., 2000. ProFound: An expert system for protein identification using mass spectrometric peptide mapping information. Analyt. Chem. 72, 2482–2489.
Zerbino, D.R., Birney, E., 2008. Velvet: Algorithms for de novo short read assembly using de Bruijn graphs. Genome Res. 18 (5), 821–829.

Further Reading

Brudno, M., Malde, S., Poliakov, A., *et al*., 2003. Glocal alignment: Finding rearrangements during alignment. Bioinformatics 19 (Suppl. 1), i54–i62. 90001.
Dohrmann, J., Puchin, J., Singh, R., 2015. Global multiple protein-protein interaction network alignment by combining pairwise network alignments. BMC Bioinform. 16 (Suppl. 13), S11.
Dündar, F., Skrabanek, L., Zumbo, P., 2015. Introduction to Differential Gene Expression Analysis Using RNA-Seq. Applied Bioinformatics Core/Weill Cornell Medical College.
Faisal, F.E., Zhao, H., Milenkovic, T., 2015. Global Network Alignment in the Context of Aging. IEEE/ACM Trans. Comput. Biol. Bioinform. 12 (1), 40–52.
Jones, N.C., Pevzner, P., 2004. An introduction to bioinformatics algorithms. MIT press.
Peris, G., Marzal, A., 2014. Statistical significance of normalized global alignment. J. Comput. Biol. 21 (3), 257–268.
Vyatkina, K., 2017. De novo sequencing of top-down tandem mass spectra: A next step towards retrieving a complete protein sequence. Proteomes 5 (1), 6.

Software

Alignment

a. MUMmer http://mummer.sourceforge.net/
b. Minimap2 https://github.com/lh3/minimap2
c. NCBI BLAST https://blast.ncbi.nlm.nih.gov/
d. Bushnell, B., 2014. BBMap: a fast, accurate, splice-aware aligner. https://sourceforge.net/projects/bbmap/
e. List of sequence alignment software https://en.wikipedia.org/wiki/List_of_sequence_alignment_software

Pairwise alignment

f. CLUSTALW/CLUSTALW2/CLUSTALO https://www.ebi.ac.uk/Tools/msa/clustalo/
g. T-COFFEE http://www.tcoffee.org/
i. MAFFT https://www.ebi.ac.uk/Tools/msa/mafft/
h. MUSCLE https://www.ebi.ac.uk/Tools/msa/muscle/

Reads QC

a. Bushnell, B., 2014. BBTools software package https://sourceforge.net/projects/bbtools/
b. Andrews, S., 2010. FastQC: a quality control tool for high throughput sequence data. https://www.bioinformatics.babraham ac.uk/projects/fastqc/.

Functional annotation systems

a. BioConductor - https://www.bioconductor.org/
b. DAVID - https://david.ncifcrf.gov/summary.jsp
c. Quevillon, E., Silventoinen, V., Pillai,S., Harte, N., Mulder, N., Apweiler, R., Lopez, R. 2005. InterProScan: protein domains identifier. Nucleic Acids Res. 33(Web Server issue):W116–20. https://github.com/ebi-pf-team/interproscan
d. Ensembl: http://uswest.ensembl.org/index.html

Relevant Websites

https://biocyc.org/
BioCyc database.
https://www.ebi.ac.uk
EBI BioModels Database.
https://en.wikipedia.org/wiki/FASTA_format
FASTA format.
https://en.wikipedia.org/wiki/FASTQ_format
FASTQ format.
http://www.geneontology.org/page/introduction-go-resource
Gene Ontology Consortium.
ftp://ftp.geneontology.org
Gene Ontology Consortium.
http://www.genome.jp/kegg/ko.html
KEGG.
www.genome.jp
KEGG database.
http://www.ncbi.nlm.nih.gov
NCBI - NIH.
https://www.nanoporetech.com/
Oxford Nanopore Technologies.
https://omictools.com/pepfrag-tool
PepFrag.
https://en.wikibooks.org/wiki/Proteomics/Protein_Identification_-_Mass_Spectrometry/Databases
Proteomics/Protein Identification.
http://prospector.ucsf.edu/
ProteinProspector.
http://prospector.ucsf.edu/
ProteinProspector.
http://www.rcsb.org/pdb
RCSB PDB.
sbml.org
SBML format.
https://omictools.com/sequest-tool
SEQUEST.
http://www.illumina.com
Solexa technology.
https://www.proteinresearch.net/
The Protein Research Foundation.
http://www.uniprot.org/
UniProt.
http://www.uniprot.org/uniprot/
UniProtKB.
http://www.uniprot.org/
UniProt.
http://www.uniprot.org/uniprot/
UniProt.
https://www.uniprot.org/help/uniref
UniProt.
https://www.uniprot.org/help/uniparc
UniProt.

Biographical Sketch

Andrey Prjibelski (formal transliteration Przhevalsky) received BSc degree in informatics from St. Petersburg State Technical University in 2010 and MSc in applied mathematics and physics from St. Petersburg Academic University in 2012. In 2011 he started an internship at Algorithmic Biology Lab led by Prof. Pavel Pevzner in St. Petersburg Academic University and then continued to work there as a junior research fellow. In 2014 the whole laboratory moved to St. Petersburg University and became Center for Algorithmic Biotechnologies (CAB). Andrey participated in development of one of the main lab projects – SPAdes de novo genome assembler and in 2014 received outstanding student paper award at ISMB 2014, Boston. Later he participated developing SPAdes-based de novo transcriptome assembler and improving SPAdes for large genome assembly. Andrey also teaches "NGS data analysis" for master students St. Petersburg Academic University and participates in various bioinformatics workshops and schools as a teacher and co-organizer.

Anton Korobeynikov received the MSc and PhD degrees in applied mathematics from St. Petersburg State University, Russia, in 2007 and 2010, respectively. He started to work at St. Petersburg State University in 2007, where he currently holds the position of Associate Professor of Statistical Modeling Department, School of Mathematics and Mechanics. The main areas of research interests of Dr. Korobeynikov are computational and applied statistics, including time series analysis, efficient implementation of algorithms of computation statistics and probabilistic methods in bioinformatics.

In 2012 he joined Algorithmic Biology Laboratory at St. Petersburg Academic University. The laboratory led by Prof. P. Pevzner is developing SPAdes – a novel de novo assembler suitable for single cell and multi cell data among other tools. In 2014 the whole laboratory moved to St. Petersburg University and became Center for Algorithmic Biotechnologies. Currently Anton is a SPAdes team leader and oversees the technical questions related to SPAdes development.

Professor Alla Lapidus holds a PhD in Molecular Biology from the Institute for Genetics and Selection of Industrial Microorganisms (IGSIM) and an MS Degree in Physics from the Moscow Engineering Physics Institute. Her primary areas of expertise include high-throughput DNA sequencing, genome structure analysis, reference quality genome assembly, bioinformatics. Currently the deputy-director of the Center for Algorithmic Biotechnology, St. Petersburg State University, Dr. Lapidus was previously an Associate professor at Fox Chase Cancer Center. From 2003 Dr. Lapidus has been applying high-throughput DNA sequencing, genome-wide analysis and bioinformatics to the study of microbial and fungal genomes and metagenomic communities by optimizing data QC, data analysis and de-novo assemblies at Joint Genome Institute (JGI). She has spearheaded multiple projects using NGS to optimize data QC, analysis and de-novo assemblies, created microbial finishing pipeline from the ground up and was the mind behind the development of the next generation de-novo assembly algorithms.

Previously she served as the Director of Sequencing Center, Integrated Genomics, as a Senior Research Scientist at the INRA, France, at The University of Chicago and at the IGSIM, Moscow.Professor Lapidus has authored several on-line courses on Bioinformatics and is a lead of the MS program "Bioinformatics" at SPbU. She also authored and co-authored over 350 papers and patents.