International Journal of Education and Science Research Review
Volume-11, Issue-6 Nov-Dec-2024
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
E-ISSN 2348-6457
P-ISSN 2349-1817
www.ijesrr.org Impact Factor - 7.02 Email- editor@ijesrr.org

STUDY THE USE OF COMPUTATIONAL TECHNIQUES IN ANALYZING BIOLOGICAL DATA"

Anita

Research Scholar, Dr. K. N. Modi University, Newai, Rajasthan

Dr. Amit Kumar Sharma

Assistant Professor, Dr. K. N. Modi University, Newai, Rajasthan

Abstract

This chapter focuses on several biological sequence analysis techniques used in computational biology and bioinformatics. The first section provides an overview of biological sequences (nucleic acids and proteins). Bioinformatics helps us understand complex biological problems by investigating similarities and differences that exist at sequence levels in poly-nucleic acids or proteins. Alignment algorithms such as dynamic programming, basic local alignment search tool and HHblits are discussed. Artificial intelligence and machine learning methods have been used successfully in analyzing sequence data and have played an important role in elucidating many biological functions, such as protein functional classification, active site recognition, protein structural features identification, and disease prediction outcomes. This chapter discusses both supervised and unsupervised learning, neural networks, and hidden Markov models. Sequence analysis is incomplete without discussing next-generation sequencing (NGS) data. Deep sequencing is highly important due to its ability to address an increasingly diverse range of biological problems such as the ones encountered in therapeutics. A complete NGS workflow to generate a consensus sequence and haplotypes is discussed.

Keywords: *techniques, biological data, computational*

Introduction

Biological sequence data are growing exponentially. The rate of growth over the last decade has also been truly astonishing, with the total amount of sequence data produced doubling approximately every seven months . Data growth rate will continue for the foreseeable future, since multiple concurrent genome sequencing projects have begun, with more to come. The availability of big biological data is vital for evolutionary studies. For the first time we can study the governing factors in the evolutional processes of whole genomes. This is therefore an exciting era for evolutional biology. However, as the semi-conductor lithography process approaching its physical limits, the growth of transistors on a single chip is much slower than the growing rate of biological sequence data. The computational load is further compounded by the addition of new data sources (many completed genomes are being reported monthly), increase in the size and number of queries, a growing user base of bioinformatics scientists, new algorithms and methods of analysis. Because of the following factors, it usually takes long runtimes to solve big biological data analysis problems:

- Sequencing technologies to produce biological data are prone to errors. Thus high complexities will be introduced into algorithms in order to handle these errors and uncertainties.
- Big biological data analysis problems have a very high computational requirements even the corresponding algorithms have polynomial time complexities.
- Due to inherent algorithmic complexities, many biological data analysis problems are both data-intensive and compute-intensive. HPC may provide an efficient tool to solve these problems.

OBJECTIVE

1. to study the use of computational techniques in analyzing biological data
2. to study needleman–wunsch matrix

This is a new area of biological sciences where computational methods are essential for the progress of the experimental science, and where algorithms and experimental techniques are being developed side by side.

Traditionally, HPC platforms such as supercomputers were rare and available for only the most critical problems. Since the mid-1990s, however, the availability of supercomputers has changed dramatically. With multi-threading support built into microprocessors and the emergence of multiple processor cores on a single silicon die, supercomputers are becoming ubiquitous. Now, almost all university computer science department has their own HPC platforms. Given the exponential growth in the size of biological sequence data, the computational biology (CB) area has taken dramatic leaps forward with the availability of computational resources. Traditional uses of HPC platforms in scientific computing usually involve problems described in structured grids, with well-defined regular data structures. In contrast, many problems in CB have irregular structures, which appears to be significantly more challenging to parallelize. Thus, the effective use of HPC platforms will become increasingly important in CB. This continues to remain a largely unexplored territory, and is the principal motivation behind our survey work.

In the past few years, the fast increasing power of new generation many-core architectures has opened up a range of new possibilities to achieve HPC for a variety of applications. Graphics Processing Units (GPUs) are one of the most widely used general-purpose many-core architectures. These commodity chips have enhanced their programmability to perform more general computational tasks than the graphics processing they were originally designed for. Examples include scientific computing, image processing,computational biology,electronic design automation (EDA and data science , etc. The computer video game market have driven the evolution of GPUs to yield relatively cheaper price per unit and very rapid iteration of hardware architectures. Intel Xeon Phi is another popular many-core architecture. It is based on the Intel's Many Integrated Core (MIC) architecture which integrates much more simplified hardware cores compared to traditional CPUs. With the easy programmability of x86-based Xeon Phi, these chips are now widely used. Scientists and engineers in a variety of fields have presented their design and implementation of parallel algorithms on Xeon Phi. Examples include scientific computing , database operations and computational biology. Limited by power consumption and advances in lithography, the many-core architectures shows better power-efficiency than the traditional multi-core CPUs. Thus, the many-core based platforms are even more attractive for the HPC community in the near future. However, there are still many challenges to be solved for the CB scientists to facilitate efficient usage of many-core based HPC platforms. In this paper, a survey and taxonomy of HPC big biological data analysis applications on various computing platforms are presented.

PAIRWISE ALIGNMENT AND DYNAMIC PROGRAMMING

Pairwise alignment involves comparing two sequences against each other and finding the best possible alignment between them. The process involves scoring at each position for match, mismatch, and indels. Since matches are preferred over deletions, matches are normally assigned the highest scores, and lowest for insertions. Similarity between two sequences is inversely proportional to the number of mismatches and indels in their alignment. Although the scoring for alignment can be as simple as +1 for match, 0 for mismatch, and −2 for insertion, different scoring models have been developed based on the statistically relevant frequencies of one amino acid changing into another.

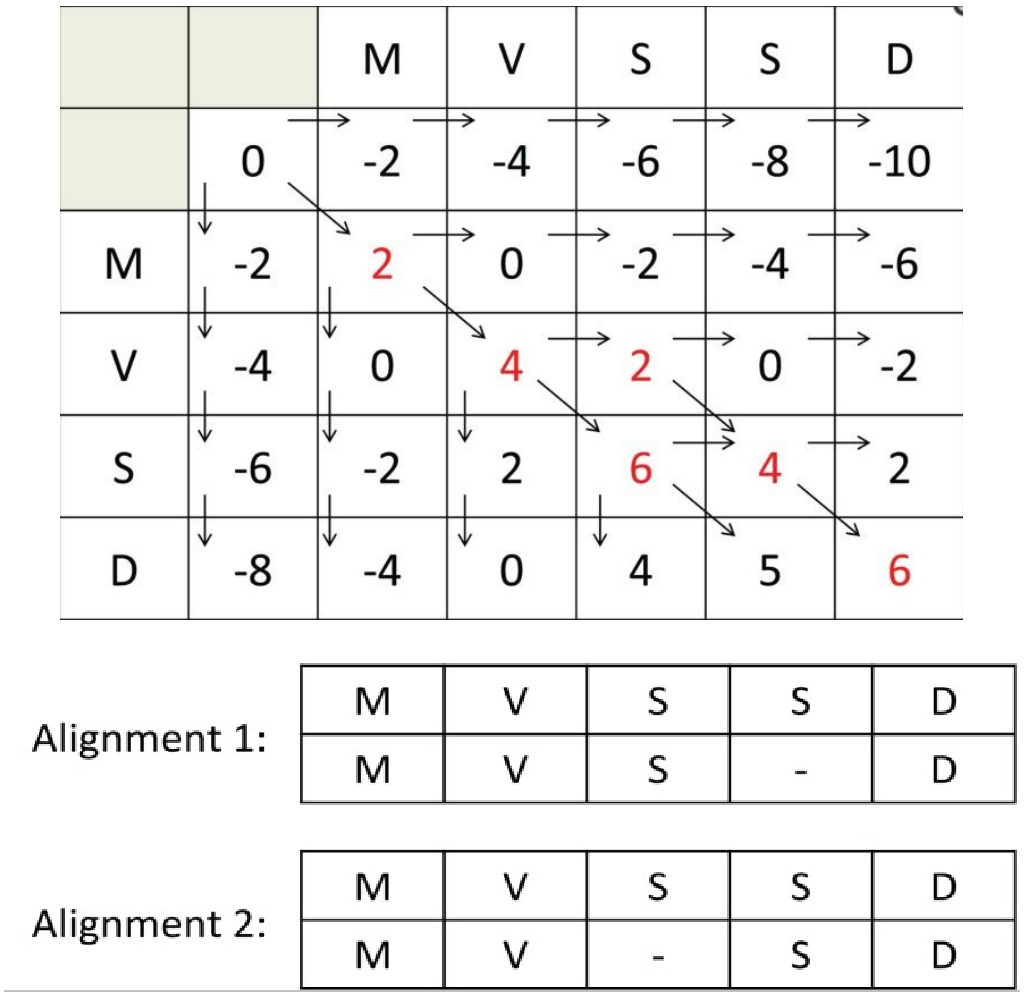
Needleman–Wunsch algorithm

Initially developed by Needleman and Wunsch in 1970, the algorithm is based on dynamic programming and allows for global or end-to-end alignment of two sequences (8). The algorithm involves three main steps, namely initialization, calculation, and trace back. A matrix of dimensions i, j is initialized, where i and j are the length of two sequences under comparison. In the second step, F(i, j) highest score for each comparison at each position is calculated,

$$F(i,j) = max\left\{F(i-1,j-1)+s(x_i,y_i), F(i-1,j)-d, F(i,j-1)-d\right\}$$

where “s(xi, yi)” is the match/mismatch score and “d” is the penalty for deletion.

After the maximum score for each position in the matrix is calculated (Figure 1), trace back starts from the last cell (bottom right) in the matrix. Each step involves moving from the current cell to the one from which the value of the current cell was derived. A match or mismatch is assigned if the maximum score was derived from a diagonal cell. Insertion/deletion is assigned if the score was derived from the top or left cell. After the trace back is completed, we have two sequences aligned end to end with each other with an optimal alignment score (9).



		M	V	S	S	D
	0	-2	-4	-6	-8	-10
M	-2	2	0	-2	-4	-6
V	-4	0	4	2	0	-2
S	-6	-2	2	6	4	2
D	-8	-4	0	4	5	6

Alignment 1:

M	V	S	S	D
M	V	S	-	D

Alignment 2:

M	V	S	S	D
M	V	-	S	D

Figure 1 Needleman–Wunsch matrix. The calculation uses scores for match +2, mismatch −1, and gap −2. The arrows show the matrix cell from where the value is generated. Red-coloured cell values show the trace back that creates alignment.

Smith–Waterman algorithm

Initially proposed by Smith and Waterman in 1981, the algorithm allows for local sequence alignment and is like the Needleman–Wunsch algorithm (10). Local sequence alignment can be used in situations where it is required to align smaller subsequences of two sequences. In the biological context, such a situation may arise while searching for a domain or motif within larger sequences. The algorithm comprises of the same steps as Needleman–Wunsch; however, there are two main differences. Computation of max score also includes an option of 0:

$$F(i,j) = max\,\{0, F(i-1, j-1) + s(x_i, y_i), F(i-1, j) - d, F(i, j-1) - d\}$$

Assignment of “0” as max score corresponds to starting a new alignment. It allows for alignments to end anywhere within the matrix. The trace back therefore starts from the highest value of F(i, j) in the matrix and ends where it encounters 0.

HEURISTIC LOCAL ALIGNMENT

One main challenge in bioinformatics sequence analysis is decoding the vast number and length of sequences. These big data of protein and DNA sequence databases (over 100 million sequences) come from species across the tree of life. Although the local alignment methods based on dynamic programming are quite accurate and guarantee to find an optimally scored alignment, they are slow and not practical for sequence alignments against databases with millions of sequences. The time complexity of dynamic programming algorithms is O(mn), that is, the product of sequence lengths. In the initial attempts to improve the speed for sequence comparisons, heuristic algorithms like BLAST (11), BLAT (12), and FASTA (13, 14) were created. Further advancements in the efficiency of similarity search algorithms came with algorithms like LSCluster (15), Usearch (16), Vsearch (17), Diamond (18) and Ghostx (19). In general, these algorithms search for exact matches and extend the alignment from those matches trying to estimate the optimal scoring alignment.

Basic Local Alignment Search Tool, initially developed by Altschul and colleagues (11), is based on the idea that the best scoring sequence alignment would contain the highest number of identical matches or highly scoring sub-alignments. The algorithm carries out the following steps: (i) reduce the query sequence into small subsequences called seeds, (ii) search these seeds across the database for exact matches, and (iii) extend the exact matches into an un-gapped alignment until a maximal scoring extension is reached. The use of seeds to first search for exact matches greatly increases the whole searching process and the un-gapped alignment misses only a small set of significant matches. The accuracy and sensitivity of BLAST made it amongst the most widely used search algorithm in the biological world. A variant of BLAST named Position-Specific-Iterative BLAST (PSI-BLAST) extends the basic BLAST algorithm (20). PSI-BLAST performs multiple iterations of BLAST and uses the hits found in one iteration as a query for the next iteration. Although slower due to sheer amount of calculations required, PSI-BLAST is considered a reliable tool to find distant homology relationships.

Although BLAST and PSI-BLAST are extensively used, recently developed methods offer results with higher accuracy and sensitivity. Hidden Markov models (HMM) have been used efficiently for numerous applications to understand and explore biological data. One such example is HMM–HMM-based lightning fast sequence search (HHblits) introduced in 2012 (21). The tool can be used as an alternative for BLAST and PSI-BLAST and is 50 to 100 times more sensitive. The high sensitivity of the tool can be attributed to the algorithm which relies on comparing the HMM representations of the sequences. Although profile–profile or HMM–HMM alignments are very slow to compute, the prefilter in HHblits reduces the required alignments from millions to thousands, thus giving it a considerable speed advantage. HHblits represents each sequence in the database as a profile HMM. Prefiltering reduces the number of HMM comparisons for similarity search by selecting only those target sequences where the largest un-gapped alignment exists, and a Smith–Waterman based alignment reveals a significant E-value.

MACHINE LEARNING AND SEQUENCE ANALYSIS

Biological data provide amongst the perfect use cases of machine learning and artificial intelligence algorithms. This is the reason that researchers in the field of bioinformatics and computational biology have used statistical analysis and inference since the very beginning. Techniques like maximum likelihood (22) and neighbor joining (23) have been used for comparative genomics. Naïve Bayes and Markov chains have been extensively used for sequence analysis. Logistic regressions, support vector machines, and random forests have been used in numerous applications ranging from prediction of protein sequence or structural elements to classification of proteins into different structural and functional classes. With the development of deep neural networks, we observe an increase in the use of the algorithms like long short-term memory (LSTM) (24) and convolutional neural networks (CNN or ConvNet) (25) to predict the different features and behavior of proteins, for example, protein contact prediction and prediction of post-translational modifications.

Machine learning methods are broadly divided into two types, supervised and un-supervised learning. Based on the inherent features of the data, if it is not labeled and cannot be assigned to any type, then classification is done using unsupervised learning. For instance, the classification of proteins into different groups is done based on their sequence similarity to each other. K-means clustering algorithm (26) and Markov clustering (27) can be used in unsupervised classification. On the other hand, if the data are labeled into different sets, this information can be used to train the computer by showing it positive and negative examples. Once the training is complete, the accuracy of training can be tested using similar data not used in the training dataset. Any classification technique following training and testing procedures using labeled data is termed supervised machine learning. Examples for this type of learning include SVM, HMM, random forest, and CNN.

Hidden Markov Models

HMM is a statistical method that can be used to predict the probability of occurrence for a future event. HMMs provide the foundations for a range of complex models that can be used for multiple sequence alignment, profile searches or detection of sequence elements. In order to understand the HMMs and their use in biological data, consider the example of binding site recognition on a DNA sequence. There is an observable sequence of nucleotides which in the right order hides underneath a binding site. We can observe the nucleotide sequence, but the presence or absence of a binding site remains hidden to us. HMMs are particularly suited for such problems because they use observed frequencies to calculate emission and transition probabilities to decipher the hidden states. An HMM involves two types of probabilities, transition and emission probabilities. The probability of moving from one state to another is called the transition probability. The probability to observe a variable within a state is called emission or output probability.

Figure 2 shows a schematic HMM with basic architecture and elements. HMMs have been used not only to create sequence profiles but also to create probabilistic model representation of protein clusters. Pfam is an example database that clusters proteins based on their functional elements and represents them with HMM. The downside to HMMs is that they assume a future event depends only on the event that happened immediately before and not in the distant past. This creates a limitation to use standard HMMs in complex cases where sequence elements influence each other that may be close in the three-dimensional space but sequentially lie far from each other. Outside of the biological world, one such example is autocomplete or word suggestions. The words appearing in suggestion are directly dependent on the word that appeared immediately before the present suggestion.

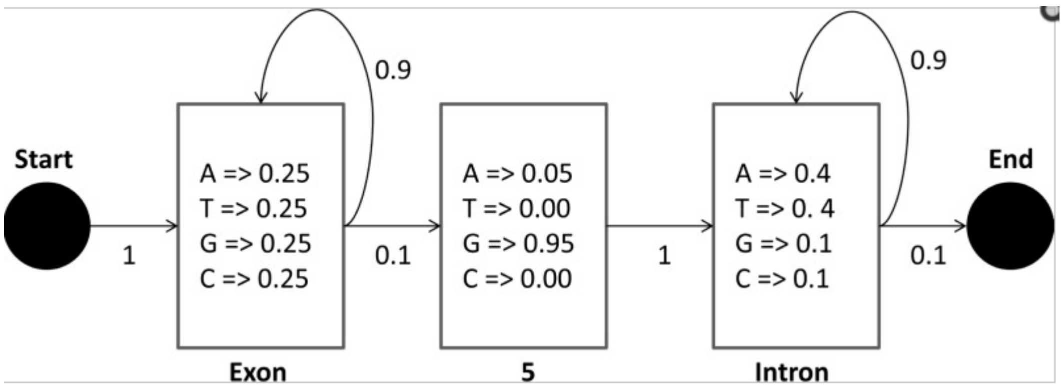


Figure 2

Hidden Markov model. The HMM is designed to predict the G rich splice site. The value inside the boxes show emission probabilities, that is, the probability for each nucleotide to appear while the values outside show transition probabilities to move from one state to the next. HMM representation adapted from (9).

Neural networks

Artificial neural networks is another classification technique with numerous applications in computational biology. Neuron is the basic unit of an artificial neural network. Each neuron can have multiple input connections with weights assigned to each of them. The output value from the neurons is calculated according to its activation function. A neural network may consist of multiple layers, with each layer containing multiple neurons. Figure 3 shows a multi-layered neural network with 32 neurons and 192 edges. Neural networks are used in supervised learning and classification. This approach uses labeled data and follows the main steps listed below:

1. Dataset: Divide the data into training sets and testing set (mostly 70–30% split or 60–40% split, respectively).
2. Training: Use the training data to traverse over the neuron and estimate the output.
3. Iterate: Based on the difference between the actual and estimated output, calculate the error and adjust the weights accordingly. Repeat step 2.
4. Testing: After multiple iterations between step 2 and 3, the model is trained and can be tested. Use the test set (unseen data for model) to compute the output. As the actual label is known, the accuracy and sensitivity can be calculated based on the correct (true positives or true negatives) and incorrect classifications (false positives or false negative).
5. Validation: The training- and test-set splits are randomized and new sets are created from the existing dataset. This new test-train split is then used again iterating over steps 2–4. The idea is to create a model independent for generalized datasets. Depending on situations, there can be multiple iterations for this step and hence referred to k-fold cross validation.

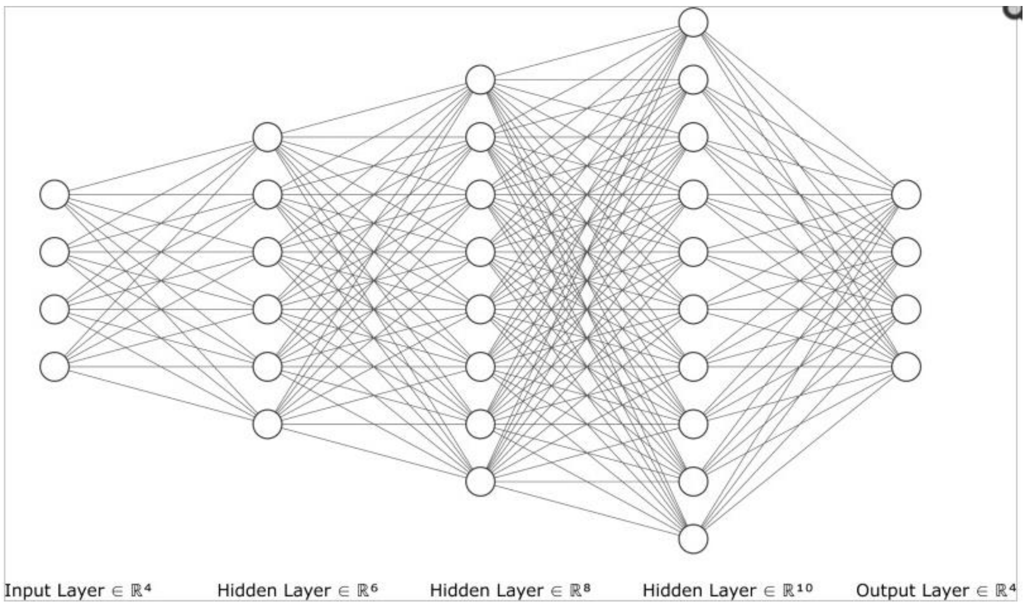


Figure 3

Neural network representation. Each node represents a neuron, and the edges depict weights that connect the neurons between layers. After each iteration, the weights are adjusted to correct for error.

In order to assess the performance of the model, outputs are calculated from different models based on different activation functions or even different neural network architectures. Sensitivity (recall) and accuracy are calculated for each of the models, and the best performing model should have a high recall rate.

The performance of machine learning in general and neural networks in particular depends highly on the quality of the data. A high-quality data would have low noise/junk while having a high homogeneity. Noise in biological data can refer to ambiguous sequence elements or incorrect labels. A high homogeneity results in an equal distribution of diversity in data across different data splits. Assuring the good quality of data before model training is a very important and time-consuming step for data scientists. If the training dataset is not a homogenous representative of the population, it can lead to a biased classification in the models. A bias model can show promising results for the testing dataset but fails in the actual world. This happens because the model is trained to classify only those types of cases that it observed during the training, and a bias sample resulted in a skewed perception of the real-world scenario. The quality of classification from neural networks also depends highly on the training iterations and size of datasets. While the ability for high-powered computation has greatly increased in the last decade, coupled with biological big data, neural networks can be used to train accurate classifiers. Neural networks have now evolved into their more complex form called “Dense Networks” or “Deep learning.” These networks (e.g., LSTM) comprise numerous neurons and high number of hidden layers between the input and output layers (hence deep network). Although the depth of a network results in a better-quality model, they are difficult to train due to the requirement of high computing power.

NEXT-GENERATION SEQUENCING

The last three decades have seen a continuous evolution of sequencing technologies. Starting from traditional Sanger sequencing to whole genome shot gun sequencing by Craig Venter and later next-generation sequencing (NGS) (4). The latest amongst these is the "Nanopore," highly compact and efficient sequencing that connects to a computer via USB; it is easily transportable and fits on a small desktop. The technology that initially required thousands of dollars per nucleotide is much cheaper now. An NGS pipeline comprises of two main sections: a wet lab section involves sample preparation, amplification, and sequencing; and the second section involves a bioinformatics workflow that uses the data generated by the wet lab to derive a sequence and other information. It is important to note that the bioinformatics section involves sequence analysis algorithms that are based on statistical and heuristic techniques to analyze and generate sequences. This section focuses on the bioinformatics aspect of NGS since it has evolved an ecosystem of computational algorithms and pipelines around it for accurate and efficient sequencing. NGS is a massively parallel sequencing technology, also referred as high-throughput sequencing, that allows analysis of large fragments of DNA and RNA genomes with high sensitivity, much more quickly and cheaply than the previously used Sanger sequencing methodology. In NGS, different platform technologies follow the same eight major steps (Figure 4):

Library preparation: The first step in NGS workflow involves preparation of high-quality and high-yield sequence library. The isolated genomic DNA or RNA is sheared into smaller fragments ranging from 150–5000 base pairs (bp) depending on the sequencing platform. The desired library can be created using either of the two fragmentation approaches, mechanical shearing or enzyme-based fragmentation (28, 29). Mechanical shearing methods include acoustic shearing, needle-shear, sonication, and nebulization, whereas enzyme-based methods involve transposons and restriction enzymes (endonucleases) (30). The small fragments known as reads have short overhangs (sticky ends) of 5'-phosphate and 3'-hydroxl groups. These ends are repaired by adenylation at 3' ends resulting in adapter ligation that is important for amplification. During library preparation, unique barcodes can be added to the fragments facilitating multiple sequencing of various samples in the same run (31).

Amplification: The goal of this step is to create thousands of copies for each read. The library is loaded onto the flow-cell, and the fragments are amplified using clonal amplification methods such as emulsion PCR or bridge amplification. In emulsion PCR, the library is amplified within a tiny water droplet floating in an oil solution (32, 33). In bridge amplification, the single-stranded DNA from the library is hybridized to the flow-cell's surface-bound forward and reverse oligos that are complementary to the library adapter sequences. Hybridized at one end, the singe-stranded DNA then folds over to form a bridge and binds to adapter-complementary oligos at the other end. DNA polymerase adds nucleotides to amplify DNA, and a clonal cluster is generated as the original strand is washed away leaving complementary strands of amplified DNA attached to the flow cell. (34).

Sequencing: The amplified individual sequences are sequenced using different platforms and sequencing technologies that include Illumina (Solexa) sequencing, Roche 454 sequencing, and Ion Torrent (Proton/PGM sequencing). Illumina (Solexa) sequencing works by simultaneously identifying DNA bases (A, T, C or G), and each base emits a unique fluorescent signal as it is added to the nucleic acid chain. Illumina sequencing involves 100–150 bp read length. Illumina has some variations that mainly differ in the amount of DNA sequenced in one run (Table 1). Roche 454 sequencing is based on pyrosequencing; a technique that detects pyrophosphate release, again using fluorescence, after nucleotides are incorporated by polymerase to a new strand of DNA. Roche 454 sequencing produces sequence reads of up to 1000 bp in length. Like Illumina, it does this by sequencing multiple reads at once by reading optical signals as bases are added. Ion Torrent (Proton / PGM sequencing) measures the direct release of H+ (protons) from the incorporation of individual bases by DNA polymerase and therefore differs from the previous two methods as it does not measure light. As in other kinds of NGS, the input DNA or RNA is fragmented, this time ~200 bp. These sequencing technologies result in raw sequencing reads (20 to 1000 bp) stored in the FASTQ format which contains both the nucleotide sequence and its corresponding quality scores. These reads can be either “single-ended” or “paired-ended.” Paired-end reads are produced when the fragment size used in the sequencing process is much longer (typically 250–500 bp long).

Quality control and read filtration: After sequencing is complete, the read data are in electronic form and can be processed to generate a whole genome or a specific gene sequence using a bioinformatics NGS pipeline. Although quality control and filtration is the fourth step in generating a full analyzable sequence, it is the first step in a bioinformatics NGS pipeline. Read filtration involves removing low confidence and erroneous reads from the dataset. The amplified raw reads pass through quality control check using FastQC (35) that can produce a detailed report on the number, quality, and coverage of reads. These methods mostly work on sequence analysis techniques like clustering short reads to calculate their frequency and quality scores. It is followed by read filtration, clipping of adapters and low-quality base pairs from 3’ and 5’ ends using software such as CutAdapt (36), trimmomatic (37) and others.

Alignment: Once the read quality is acceptable, millions of raw sequence reads (single-end or paired-end) are mapped and aligned using either a reference based assembly (in which reference sequence is available) or de novo assembly (in the absence of a reference sequence). The sequence reads of variable lengths are aligned using different bioinformatics alignment tools such as BWA (38), Bowtie (39), and TopHat (40). These heuristic-based aligners allow fast sequence alignment and generate a consensus sequence from the alignment by searching the overlapping portions of the reads and merging them into longer reads in order to construct a region of interest, that is, genes or a whole genome. The main aim of this step is to generate a consensus sequence from the millions of reads. A consensus sequence shows the genetic makeup at the time of the sample collection. This step marks the completion of sequence generation for a partial or a whole genome. The following steps are important for an in-depth analysis beyond generation of only a single sequence.

APPLICATIONS OF NGS IN CLINICAL PRACTICE

The NGS technologies have several applications in research to solve a diverse range of biological problems. Comprehensive analysis of NGS data includes whole-genome sequencing, gene expression determination, transcriptome profiling, and epigenetics. NGS has enabled the researches to sequence large segments of the genome (i.e., whole-genome sequencing) and provides insights into identifying and understanding the genetic variants such as SNPs, insertions, and deletions of DNA, and rearrangements such as translocation and inversions associated with diseases for further targeted studies (45). Researchers use RNA sequencing (RNASeq) to uncover genome-wide transcriptome characterization and profiling (46). Analysis involving genome-wide gene expression (i.e., gene transcription, post-transitional modifications, and translation) and the molecular pathway analysis provide a deeper understanding of gene regulation in neurological, immunological, and other complex diseases. Other applications include studying heritable changes in gene regulation that occur without a change in the DNA sequence. Epigenetics play a significant role in growth, development, and disease progression. The studies on epigenetic changes in cancer provide insight into important tumorigenic pathways.

CONCLUSION

Sequence analysis is a broad area of research with sub-domains. Alignment of sequences can reveal important information concerning the structural and functional sites within sequences. It is used to explore the evolutionary path of sequences by identifying the sequence orthologs and homologs. Sequence analysis also involves the use of machine learning techniques for classification and prediction of sequence elements. Statistical methods are used to create sequence profiles and identify other distantly related sequences with a higher precision. Advancement of sequencing technologies has resulted in a next-generation era that opened the doors to personalized medicine and haplotype/quasi-species detection. With correctly organized NGS pipelines, it is possible to analyze the effects of drugs directly at the sequence level.

References

1. Chial, H. DNA sequencing technologies key to the Human Genome Project. Nature Education 1(1):219, (2008).

2. Hood, L., Rowen, L. The Human Genome Project: big science transforms biology and medicine. Genome Med 5, 79 (2013). https://doi.org/10.1186/gm483.

3. W.J.S. Diniz, F. Canduri, 2017. Bioinformatics: an overview and its applications. Genet. Mol. Res. 16 (1): gmr16019645, http://dx.doi.org/10.4238/gmr16019645.

4. Dash, S., Shakyawar, S.K., Sharma, M. et al. Big data in healthcare: management, analysis and future prospects. J Big Data 6, 54 (2019). https://doi.org/10.1186/s40537-019-0217-0

5. Ivan Merelli, Horacio Perez-Sanchez, Sandra Gesing, Daniele D'Agostino ―Managing analysis and integrating Big data in medical bioinformatics: Open problems and Future Perspectives‖, BioMed Research International, vol. 2014, Pages 13, (2014). https://doi.org/10.1155/2014/134023.

6. Ison, J., Ienasescu, H., Chmura, P. et al. The bio.tools registry of software tools and data resources for the life sciences. Genome Biol 20, 164 (2019). https://doi.org/10.1186/s13059-019-1772-6

7. Kuhlman, B., Bradley, P. Advances in protein structure prediction and design. Nat Rev Mol Cell Biol 20, 681–697 (2019). https://doi.org/10.1038/s41580-019-0163-x.

8. Jianyi YangIvan Anishchenko, Hahnbeom Park, Zhenling Peng, Sergey Ovchinnikov and David Baker ―Improved protein structure prediction using predicted interresidue orientations‖ PNAS, 117 (3): 1496-1503, January 21, 2020, https://doi.org/10.1073/pnas.1914677117.

9. Vallat B, Madrid-Aliste C, Fiser A (2015) Modularity of Protein Folds as a Tool for Template-Free Modeling of Structures. PLoS Comput Biol 11(8): e1004419. https://doi.org/10.1371/journal.pcbi.1004419.

10. Suryakant Chaubal ‗Operating System Concepts‖ Lulu publisher, Pages 146, ISBN-13:978-1411688599, April 2006