TECHNION
Israel Institute
of Technology

Project in Bioinformatics (236524)

Sanger Aligner

Name: Eden Nagar

ID: 312589815

E-mail: edenagar@campus.technion.ac.il

Supervisors: Yahav Festinger, prof. Ayelet Lamm

Biology faculty, Technion

November 2022

Abstract

Sanger sequencing is the most widely used sequencing method for small-scale DNA strand sequencing projects, which was developed by Fredrick Sanger about 45 years ago. Once a researcher receives the Sanger sequence from his sample, he wants to compare it to a reference sequence and find mismatches. These mismatches can be essential to study mutations and changes in the organism's DNA. Performing the right alignment between the reference's and sample's sequence might be quite challenging and time consuming. Thus, an easy-to-use application is needed to perform this analysis.

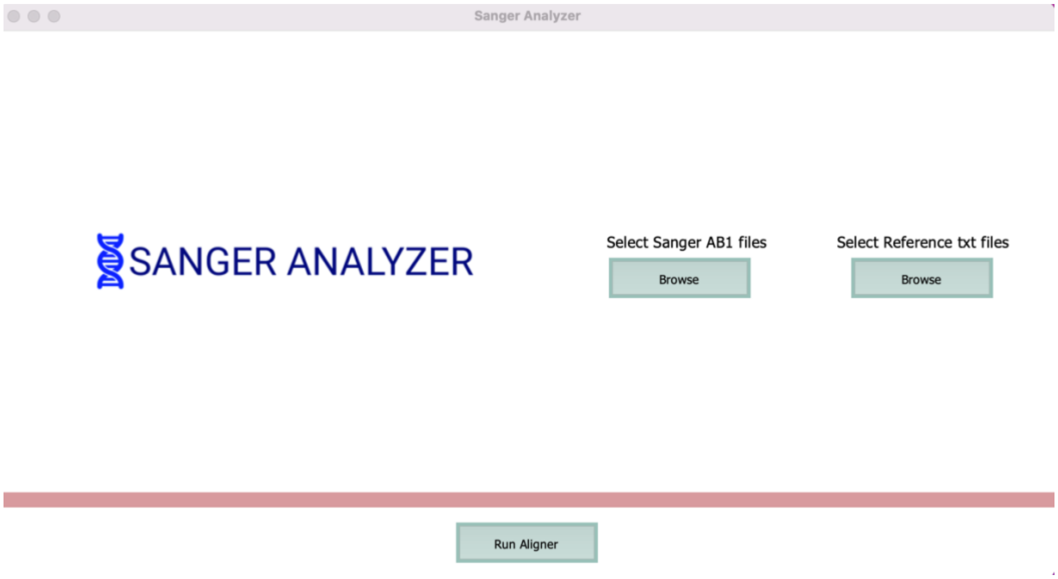
This project aims to develop a native application to help the laboratory staff quickly process the Sanger sequence. The developed application finds the most appropriate location of the user's sequence, which aligns best with a reference sequence.

The Sanger Analyzer is composed of four main components:

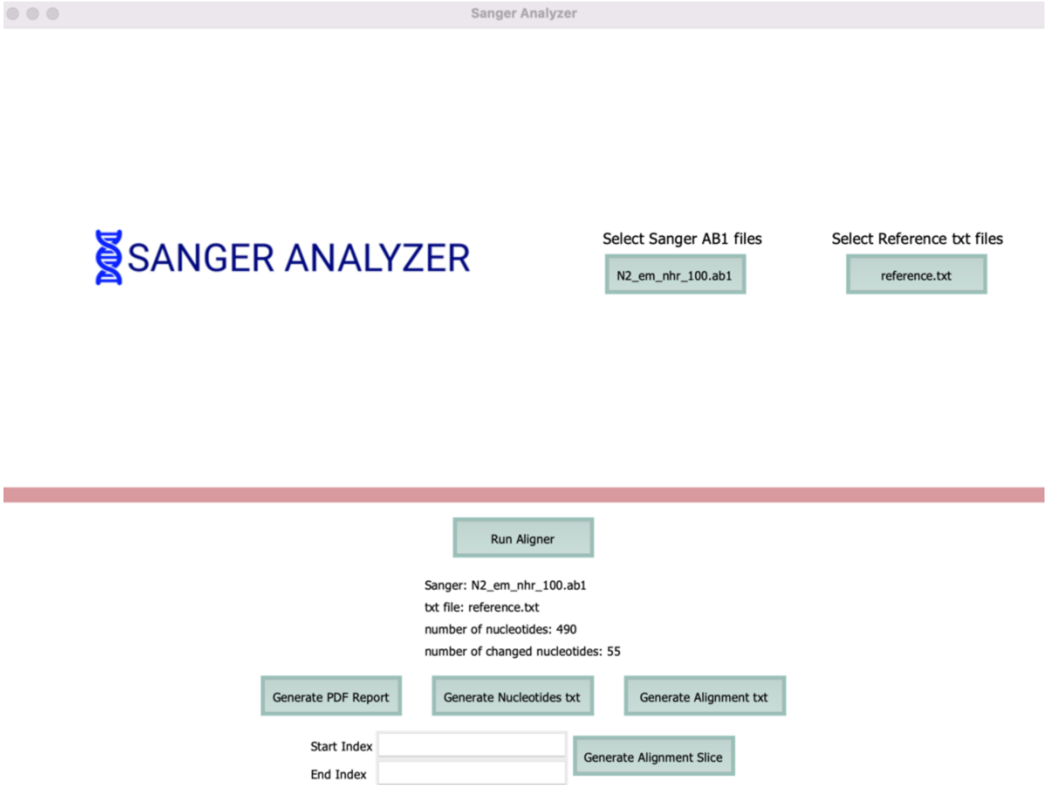
- The Sanger Analyzer can process AB1 and user text files and generate the relevant data, such as the chromatogram value of all four nucleotides, samples, and plots.
- The aligner algorithm mimics a well-established Needleman-Whench sequence aligner.
- An Interpreter algorithm, which interpreters AB1 file to the nucleotide sequence.
- The Aligner framework library was built as the communicator between the UI and the algorithm and contains several methods for further use by the lab.

The User Interface

A Simple user interface (UI) was developed. The user needs to select an AB1 file and a text reference file, and Run the Aligner. This can be done by clicking “Browse”, and then “Run Aligner”.



Then, the bottom section will appear, reviling all the data features:



N2_em_nhr_100.ab1

reference.txt

```
Sanger: N2_em_nhr_100.ab1
txt file: reference.txt
number of nucleotides: 490
number of changed nucleotides: 55
```

Now, instead of “Browse” appears the name of the chosen file.

At this window, the user can choose the following options:

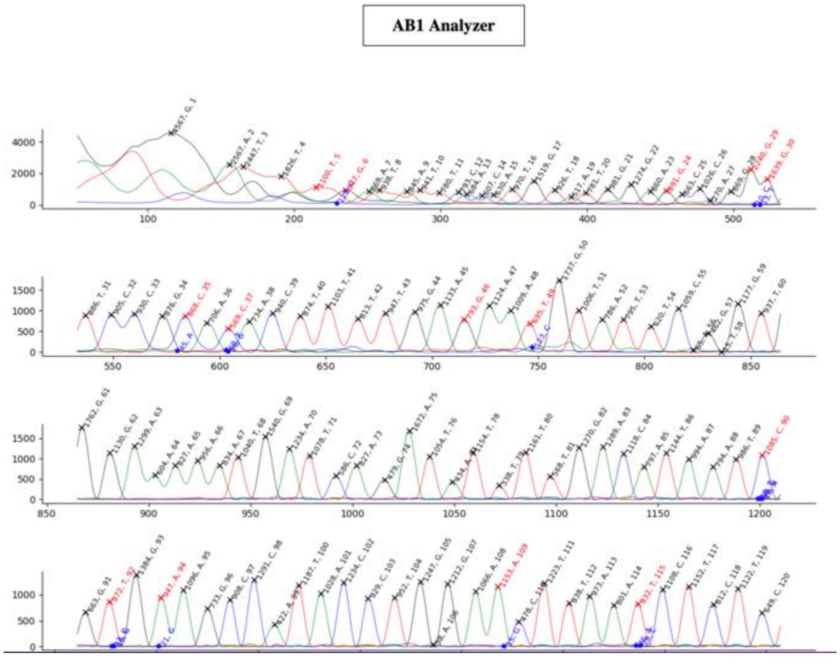
1. Generate PDF Report:

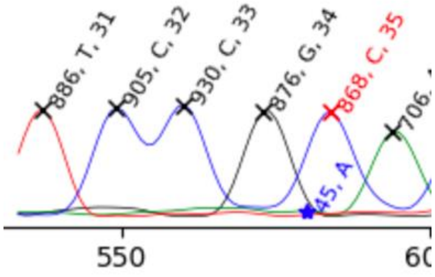
Generates a full report of the chromatogram, a relevant sanger sequence, a nucleotide type, the location of every nucleotide with its sequence Index (for instance: the sequence “ATAGCC” the index of nucleotide G is 4), and the value of the chromatogram. Another feature is the color of the nucleotides, that hints information about the alignment:

Black- match.

Red- mismatch.

Blue- a peak that is located under a mismatch.


AB1 Analyzer



2. Generate Nucleotides txt:

By clicking it, it generates a text file that contains the following information for all nucleotides originated from the AB1 file:

- Location: location of the nucleotide relative to the sample time by the Sanger machine.
- Height: chromatogram mass.

- left_ips: start approximation of the amplitude.
- right_ips: end approximation of the amplitude.
- index: nucleotide index.
- is_changed: if this nucleotide was change after alignment.
- under_peaks: all the peaks that are beneath the current amplitude and is not part of the sequence.
- amplitude_ratio: the ratio between the peak's height and the highest peak in the sequence.

For example:

location=746, height=695.0, left_ips=741.8306451612904, right_ips=749.4961832061068, nucleotide='T', index=49), is_changed=True, under_peaks=(746, [Peak(location=747, height=123.0, left_ips=743.1458333333334, right_ips=751.0263157894736, nucleotide='C', index=None)]), amplitude_ratio=0.13568918391253418

This date can be used for parsing in order to extract a specific column of data.

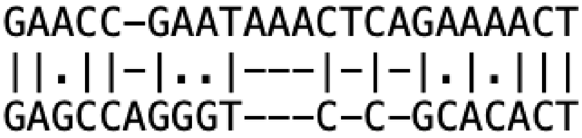
3. Generate Alignment text:

This part generates a text file that show the chosen optimum alignment. Every row contains 100 nucleoides of the alignment.

The first line shows the reference sequence.

The last line shows the AB1 sequence.

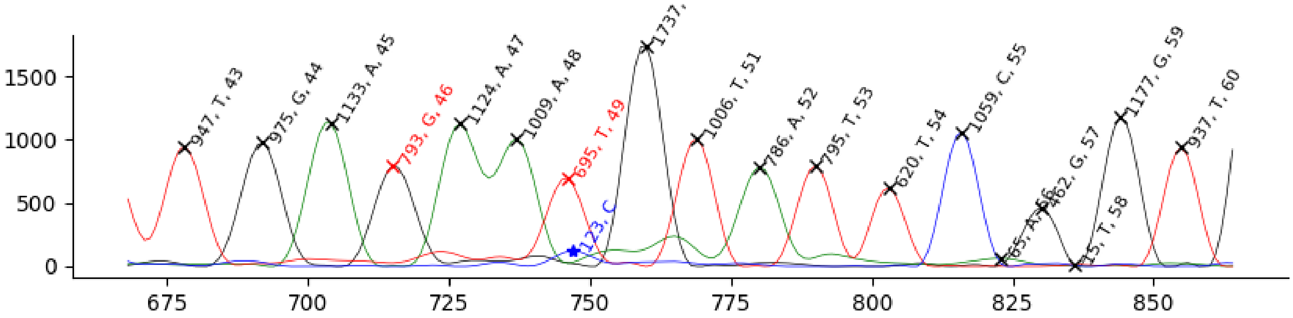
The middle line represents the symbols line when "-" stands for a gap, "|" for a match, and "." presents a mismatch.



```
GAACC-GAATAAACTCAGAAAACT
||.||-|..|---|-|-|.|.|||
GAGCCAGGGT---C-C-GCACACT
```

4. Generate Alignment Slice:

The user needs to fill a specific nucleotide's start and end indices. Given these indices, the application generates a cropped image of the desired part of the sequence.



Needleman-Wunsch Algorithm

The Needleman-Wunsch algorithm is a well-known nucleotide sequence algorithm, which finds the best scoring alignment of two sequences. For coherency, from now on we shall call the sequences “Alpha” and “Beta”.

The algorithm performs with the following complexity:

$$worst - case\ performance\ O(mn)$$

$$worst - case\ space\ complexity\ O(mn)$$

while n and m are the Sanger Sequence (Alpha) and the reference sequence (Beta) lengths. The input of the algorithm is:

1. Alpha sequence.
2. Beta sequence.
3. Match value.
4. Mismatch value.
5. Gap value.

The working steps of the algorithm:

1. The algorithm generates an empty $n + 1 \times m + 1$ grid.
2. It fills the first row and the first column: $Gap\ value * index\ of\ cell$
3. For every cell of indexes $[i, j]$, if cells $[i - 1, j], [i, j - 1]$ are filled then this cell has three possible candidate sums:
    a. The diagonal top-left neighbor has score x. If Alpha[i]==beta[j] it is a match, so the score is: $A = x + match\ value$, else it’s a mismatch then: $A = x + mismatch\ value$
    b. The top neighbor has score y and moving from there represents an indel, so it adds the score for gap value: $B = y + gap\ value$
    c. The left neighbor also has score z, represents an indel and also produces: $C = z + gap\ value$.

    The highest among A, B, C will be entered to the cell.
4. From the cell $[n, m]$ the sequence is constructed by these rules:

a. A diagonal arrow represents a match or a mismatch. If the current score is equal to the sum of the diagonal arrow and the mismatch value or match value, thus the column’s and row’s letters of the original cell will align.

b. A horizontal or vertical arrow represents an indel. If the current score is equal to the sum of the horizontal or vertical score and the gap value, it will align a gap ("-") to the letter of the row (the "side" sequence), horizontal arrows will align a gap to the letter of the column (the "top" sequence).

c. If there are multiple arrows to choose from, they represent a branching of the alignments. If two or more branches all belong to paths from the bottom right to the top left cell, they are equally viable alignments. In this case, note the paths as a separate alignment candidates.

AB1 Interpreter

The AB1 file is translated by the Aligner framework from a binary file to a four-integer type array. Every one of them represents the chromatogram level relatively to the sample time. Then, the AB1 interpreter calculates its local peaks by checking the numeric derivative and chooses all those with derivative zero as a peak, and leaves only the ones that have the highest value of all the other nucleotides. Additionally, the left_ips and the right_ips are calculated in the same manner as the numeric derivative zero. Each one of them are a local minimum.

“Aligner” Framework Library

In this project, a framework library named “Aligner” was developed. The Aligner framework is the backbone of the application; it contains all the methods that are needed for the user interface and the algorithms. The framework works like any other standard class, which needs initializing with a path to an AB1 file. The user is grunted with the following methods:

- Align_with – gets a string of the sequence and performs the alignment.
- Number_of_nucleotides – return the number of nucleotides that the AB1 sequence contains.
- Number_of_changed_nucleotides -return the number of the mismatched nucleotides.
- Print_best_alignment – prints the best alignment to the console.
- best_alignment – return the best alignment in the formatted version.
- Plot_matter_sequence_by_indexes – gets left_index, right_index, file_path, name and returns a PNG file, at the file_path with the name that was given from left to right indecis.
- Generate_pdf – in case of given file_path, it generates a PDF file with a plot of the whole sequence.
- Generate_nucleotides_info_file – gets a file_path and a string name, and generates a text file at the file path. It contains all the AB1 nucleotides information.
- Generate_alignment_file – gets file_path and a string name, and generates text file with the alignment.

In addition, the library includes several functions that are already implemented for future use:

- Plot_heat_map – gets start and finish sequence indices, and shows a heat map of the close-range changes.
- Is_noisy – gets amplitude_ratio_threshold and noise_percent_threshold. it returns a Boolean- if the ratio between the noisy elements and the size of the sequence is greater than noise_percent_threshold. Noisy elements are nucleotides that the amplitude_ratio is grater then amplitude_ratio_threshold.

In order to activate the library, the user can run a virtual environment by running: venv/bin/activate in the root folder.

The framework needs python 3.6 and above and includes all requirements in the requirement.txt file. Moreover, the framework comprises an example file, which the user can execute the major features by uncommenting.

Future Work

Although the Sanger Aligner application is enough to work with the Sanger machine and align to a given reference, these are several suggestions for future work in this field:

1. The user interface can be served on a web-based user interface, using technologies such as Django or Flask (no python environment and run). In this way, it can be accessed by any computer and include history data.
2. To add more options of file types such as generating plots of JPEG or SVG.
3. Expand the plotting configuration possibilities, to change the width of the removed text and to change colors.
4. When an AB1 file is given, the application can check on Blast's close relative sequence in the European database and add it as an extra information for the user.
5. The algorithm can be best suited for the task by distributing the weight value, of the match, mismatch and gap values by a normal distribution or a Bayesian distribution.
6. The algorithm can be improved by classifying noisy amplitudes using advance tools such as Machine learning.