# Sanger Aligner

Project in Bioinformatics (236524), Department of Computer Science, Technion
DevelopedforProf.AyeletLamm'sLab,DepartmentofBiology, Technion Spring semester, 2022

Eden Nagar, ID: 312589815, edenagar@campus.technion.ac.il

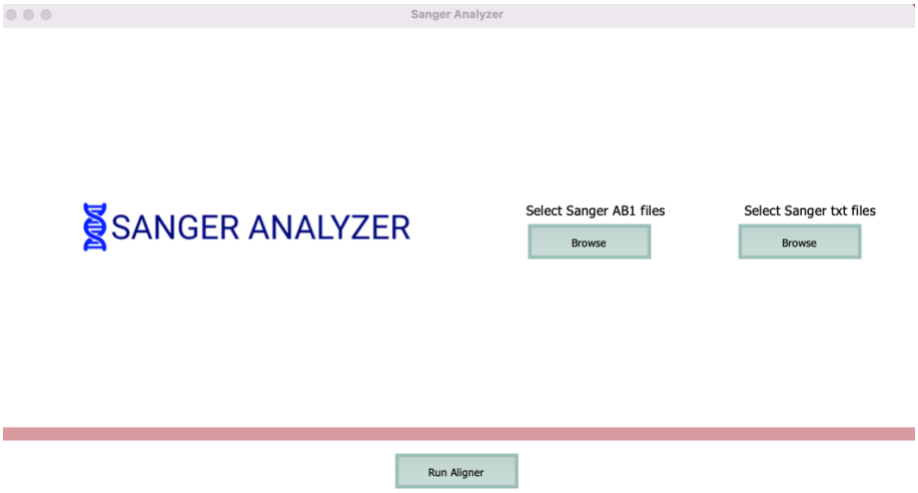
## Abstract

Sanger Analyzer project was made by the guidance of prof. Ayelet Lamm and Yahav Fester threw out the whole development.
Our goal was to develop an native application to help the lab process Sanger Sequencing machine binary file while adding their own reference sequence so that the app will find the most fitting alignment of the two so the user can extract all the relevant data an plots.
out of the need to take the output binary file from the Sanger Sequencing machine at the lab and find the sequence approximate location relative to the reference sequence.
The alignment was delt by a human until now by hand
After the alignment the lab would check miss match nucleotides to further understand if there are similar patterns.

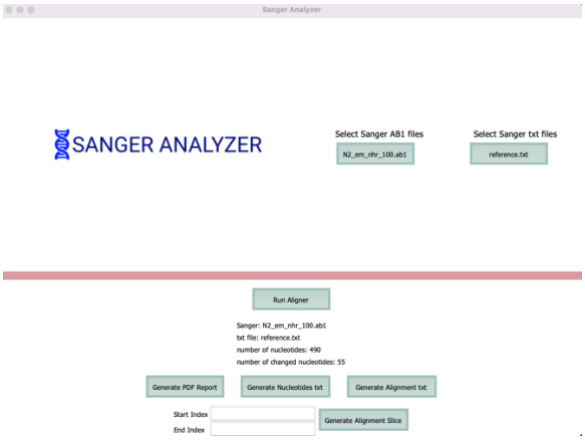
The Sanger Analyzer is composed from four main components:

- The Sanger Analyzer user interface can process AB1 files and text files from the user and generate relevant data such as the chromogram value of all four nucleotides sample and plots.
- The aligner algorithm that mimics a well establish Needleman-Whench sequence aligner.
- The AB1 to nucleotide sequence interpreter algorithm.
- The Aligner framework that was mainly built to be use as the communicator between the UI and the algorithm and hold much more methods for further use by the lab.

# The User Interface

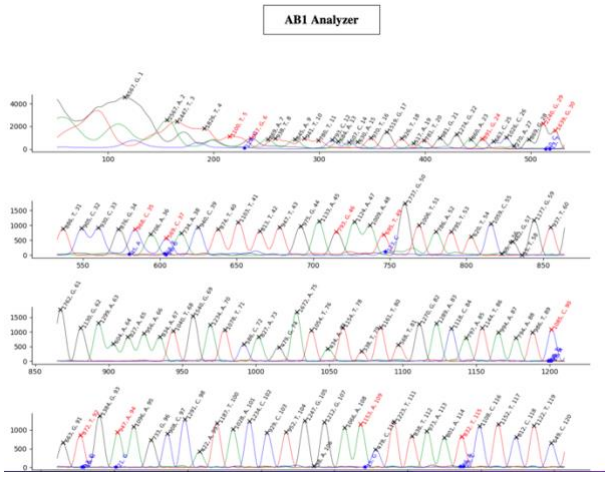



Simple user experience, the use needs to select an AB1 file and a text reference file and Run the Aligner, and then the bottom section will appear to the user reviling all the data features




The user is informed by the current Alignment and the option of proceeding, all call-to-action button here a dialog window to understand the dedicated folder of the file.

1. Generate PDF Report:
2. Generates a full report of the chromatogram of the relevant sanger sequence the nucleotide Type, the location of every nucleotide with its sequence Index (for example the sequence “ATAGCC” the nucleotide G index is 4), and the value of the chromatogram.




another feature is the color of the nucleotide that hints at its part in the alignment:
Black: it’s a match
Red: it’s a mismatch
Blue: a peak that’s under a mismatch

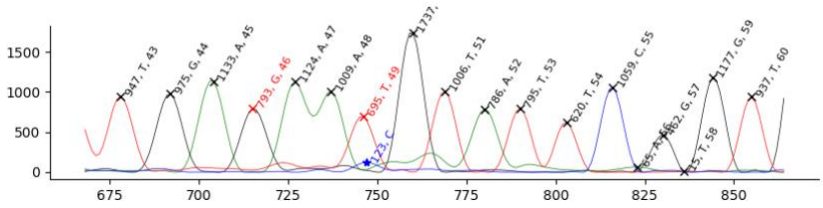
3. Generate Nucleotides txt:
   Generate a text file that holds for all nucleotide the following information for every nucleotide from the AB1 file.

   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: after alignment is this nucleotide was change.
   under_peaks: all the peaks that are beneath the current amplitude and is not part of the sequence.
   amplitude_ratio: this peak height / the highest peak in the sequence.
4. Generate Alignment text:
   Generates a text file that show the chosen optimum alignment that every row holds 100 nucleotides,

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

The top sequence is the reference sequence, The bottom is the AB1 sequence and between is the symbol row that symbolizes three symbols:
"-" is a gap, "|" is a match, and "." is a mismatch.

5. Generate Alignment Slice:
   Given the indexes to it left the application generates a cropped image with index start and index end as its border.



# Needleman Wunsch Algorithm

A Well-known nucleotide sequence algorithm that finds the best scoring alignment of two sequences, for coherency we shall call them from now on Alpha and Beta.
the algorithm perform with the following complexity:

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

while $n$ and $m$ are the Sanger Sequence and the reference lengths.

the algorithm input is:

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

The steps:

1. the algorithm generates an empty $n+1 \times m+1$ grid
2. fill first row and 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's a match, so add the score for match: $A = x + match\ value$,
    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 add 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 cell $[n,m]$ the sequence is constructed by these rules:
    a. A diagonal arrow represents a match or mismatch, so if the current score is equal to the diagonal arrow plus mismatch value or match value, thus the letter of the column and the letter of the row of the original cell will align
    b. A horizontal or vertical arrow represents an indel, so if the current score is equal to the horizontal or vertical score plus 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 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 is representing the chromatogram level relative to the time of the sample.
The AB1 interpreter then calculates its local peaks by checking the numeric derivative and chooses all those with derivative 0 as a peak and leaves only the ones that have the highest value of all the other nucleotides.
the chosen peaks are translated relative to the time of sample to the nucleotide sequence.

## Aligner Framework

The Aligner framework is the backbone of the application, it holds all the methods needed by the user interface and the algorithms themselves, but it holds more than that for future use of the lab.

the frameworks work like any other standard class, you need to initialize it with a path to an AB1 file, and you are 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 the AB1 sequence holds.
- Number_of_changed_nucleotides -return the number of changed nucleotides.
- Print_best_alignment – prints the best alignment to the console.
- best_alignment – return the best alignment in the formatted version.
- Is_noisy – gets amplitude_ratio_threshold and noise_percent_threshold returns a Boolean if id the amount of noisy element divided by the size of the sequence is greater than not from the amplitude_ratio_threshold.
- Plot_matter_sequence_by_indexes – gets left_index, right_index, file_path, name and return a PNG file in the file_path with the name given from index left to right.
- Generate_pdf – given file_path it generates a pdf doc that plot the whole sequence.
- Generate_nucleotides_info_file – gets file_path, name and generates a text file in the file path holding all the AB1 nucleotides with information.
- Generate_alignment_file – gets file_path, name, and generates text file with the alignment.
- Plot_heat_map – gets start and finish and shows a heat map of the close-range changes.

To activate it you can run a virtual environment by running:
venv/bin/activate

the framework needs python 3.6 and above and holds all the requirements in the requirement.txt file.

The framework also holds an example file that you can execute the major feature by documenting.

## Further development

The sanger Aligner itself is enough to work with the Sanger machine and align with a given reference,

1. the user interface can serve better on a web-based user interface using technology such as Django or Flask so no python environment and run is needed and can be accessed by any computer and hold history data.
2. Add more options of file types to generate from the needed plot.
3. Plotting configuration, change the width of the removed text and change color.
4. When an AB1 file is given, the application can check on Blast's close relative sequence in the European database and give it as extra information to the user.
5. The algorithm can be best suited to the task by the distribution of the weight value of the match mismatch and gap value by a normal distribution or a Bayesian distribution.