ShuffleAnalyzer Readme

Legal disclaimer

Background

ShuffleAnalyzer is a python-based software that has been developed as a user-friendly tool for the analysis of chimeric DNA or protein sequences generated via DNA shuffling. It addresses some missing features of the java-app Salanto, a similar software that has been published by Herrmann et al., 2018. In contrast to Salanto, ShuffleAnalyzer offers a variety of direct graphical output formats of the shuffling analysis. The development of the software took place over 2.5 years in an iterative way, such that new ideas were implemented consecutively in each development step. These enabled a lot of new features, but came along with several redesigns, especially concerning the graphical user interface (GUI). This procedure is, to a certain extent, reflected in the overall structure of the software.

The software does not require any prior programming skills, but the user should be familiar with the biological background of DNA and protein sequences, FASTA files, peptides, mutations, and DNA barcodes.

The software was developed under Mac OS Catalina 10.15.7 with

* Spyder version: 5.4.3 (conda)
* Python version: 3.8.10
* Qt version: 5.15.2
* PyQt5 version: 5.15.10
* pillow: 10.2.0
* matplotlib: 3.7.2
* numpy: 1.22.3
* pandas: 2.0.3
* seaborn: 0.12.2
* tkinter: 8.6

and tested under Win 10 / Win 11

Installation

After downloading the executable file suitable for the according operating system (available for Windows, Linux, Mac), we recommend copying it to your home directory or to the directory where your FASTA files are stored.

The software does not require a pre-existing python installation and can be started simply via double-click on the executable. This will load ShuffleAnalyzer into the main memory, which will then load relevant python modules and it may take a few moments depending on the capacity of your computer system. Note that the first launch may activate protective software from the operating system such as Microsoft Defender or MacOS Gatekeeper that needs to be overcome. For the Windows version, download the exe-file and double click on it. To overcome Microsoft Defender, click on “More info” and then “run anyway” (Figure 1a). This alert should only occur during the first launch. To start the Mac version, right-click on the icon and select “Ok” in the pop-up window (Figure 1b, left image). Next, right-click again on the icon and click on “Open” (Figure 1b, right image). Note that launching the software for the first time may take a bit. However, afterwards the software can be launched simply by double-clicking on the icon. To start the Linux version, download the file and update the permissions. Therefore, open the terminal and navigate to the directory of the ShuffleAnalyzer file. Then, type in the command “chmod +x shuffleAnalyzer”. Afterwards, the program can be started by double-clicking on the ShuffleAnalyzer icon.

Alternatively, the Python scripts can be downloaded separately from a zip-file and executed with Python (v3.8) via the “shuffleMain.py” file.

ShuffleAnalyzer workflow

The general ShuffleAnalyzer workflow is described below, a detailed overview of the graphical user interface is shown in Figure 2 and a usage flow chart in Figure 3.

Loading the input FASTA-file

To start the analysis, first the input FASTA file is loaded by a simple browse dialogue, whereas the path name of the selected FASTA file is displayed within the GUI. The input FASTA file should be an alignment of the chimeras and all parental sequences, which can be generated for example via the Clustal Omega Multiple Sequence alignment tool (https://www.ebi.ac.uk/Tools/msa/clustalo/). Here, parental as well as the chimeric sequences (obtained for example by Sanger or SMRT sequencing) can be uploaded either individually or within a combined FASTA-file. Clustal Omega then generates a new FASTA-file of aligned sequences when the output format is set to "Pearson/FASTA". Please note that the quality of the alignment will impact on the assignment, and it may require manual readjustments to correct improperly aligned positions for optimal results. For reasons of comparability, all sequences are cut to the maximal common length. As correctly aligned FASTA-files do have equal sequence lengths, this is usually not necessary.

Grouping and color assignment

In the next step, sequences will be grouped in parental and chimeric sequences. For this, the user must specify a pattern in a case sensitive manner that allows discrimination of parental and chimeric sequences, i.e., a pattern that exclusively occurs at the beginning of the FASTA header of parental sequences. For example, if parentals are named "AAV-parentals" and chimeras "AAV-chimera", the pattern must include the first 5 letters, "AAV-p", as "AAV" alone would be ambiguous. This is an essential step and if the headers do not allow discrimination, the user needs to rename them. The next step is assigning colors to parentals for subsequent graphical visualization. This can be done automatically and, if desired, colors can be reordered (by drag-and-drop) or edited. New colors can also be manually implemented by typing in the according color with a preceding " ::" into the "all sequence identifier" field next to the parental name. A list of available colors is provided here:
https://matplotlib.org/stable/gallery/color/named_colors.html

Optional settings for barcodes and peptides

If chimeras are part of a barcoded library, these barcodes can be displayed later in the horizontal bar plots. For this, a FASTA-file containing the barcodes can be loaded, which is again done by a simple browse dialogue. The software will then search for each barcode sequence and, if found, will add the FASTA header to the bar plot. If specific regions of the chimera (such as peptides) should also be analyzed, the user can specify these regions via two ways: the simple way is with position numbers, the more sophisticated way is with basic patterns (up-/downstream consensi). The rules for

feasible patterns are the following: Uppercase letters specify sequences, whereas uppercase letters in brackets allow multiple letters at a specific position (e.g. TT[AC]TT means either TTATT or TTCTT). Note that using numeric values, both the start and end position will be included. However, upon using a consensus pattern, only the sequence within both consensus patterns will be displayed. Also note that if the consensus pattern is found twice, only the first occurrence will be displayed.

Method for the analysis

Parental assignment can be performed by three different algorithms. The first one is a bidirectional approach (BD), where for each letter, sequences are analyzed in both directions (up- and downstream). The parental having the longest match with the chimera in both directions will be assigned to this specific sequence position and the procedure will be repeated for the next letter until the end of the sequence is reached. An example is shown in Figure 4a. Here, the letter in the chimera sequence Chim1 at position "i" is the same as in both parental sequences AAV1 and AAV2. Now, we go from position "i" to the left as well as to the right, and we see, that the subsequence in chimera Chim1 matching AAV2 is longer than the one matching AAV1. Hence, the chimera letter in position "i" is assigned to parental AAV2 (red). If the length of both is equal, the assignment would be indetermined (indet).

The other algorithms scan the parental sequences only in one direction, from left to right (L2R) or vice versa (R2L) as illustrated in Figure 4b and c. Similarly, the parental with the longest matching fragment will be assigned to the entire matching region of the chimera. If two or more parentals have the same matching length, the algorithm will define this region again as indetermined (indet), which can be later resolved by defining a parental set and assigning a new color to it.

Starting the assignment

Initiation of the parental assignment generates a new folder as a subdirectory within the folder of the input FASTA-file. This subdirectory is named according to the input FASTA-file name, with the extension "_Files". For example, if your FASTA-file is called "AAV27.fasta", the directory will be named as AAV27_Files. Make sure that you have writing permissions in this directory and be aware that, if the directory already exists, files within may be overwritten without an additional warning. The subdirectory stores all results including a csv file containing a detailed summary for individual chimeras, as well as the figures generated in the next step. Depending on the length of the sequences, the number of parentals and chimeras, as well as characteristics of the underlying computer system, the process of analyzing can last for a few moments. Note that the bidirectional is a bit slower than the L2R/R2L algorithms. Upon completion, a dialog with "data are analyzed" pops up and the resulting files (results.txt, diagnostic.txt, summary.csv) are saved in the directory with path name, which is shown in "path to output files".

Visualization

Diagrams are saved by default (can be turned off by unchecking the box) and comprise the following:

- Chimera peptide reference: a table showing peptide occurrences within chimeras.
- sequence or partition-based heat- and cluster maps: a figure showing the similarity/distance between chimeras. Cluster maps are generated by the default setting of the seaborn cluster map package, which uses Euclidean metric to perform hierarchical clustering.
- horizontal bar plots: a figure showing the composition of individual chimeras.
- frequency per position: a figure showing the frequency of parentals for each position (note that a binning can be set that groups positions together and accelerates processing time).
- 2D/3D length distribution diagrams: diagrams showing the length distribution of parental fragments.
- Detailed view: comprehensive table showing the exact parental assignments as well as the sequence of all chimeras and parentals. This table should be used with some special attention: it must be closed before any other operation is started! This is due to some so far not solved python problems, the global interpreter lock (GIL).
- Logo: Graphic showing the frequency of nucleotides/amino acids per position within the above defined peptide position. This may be restricted to show only peptides with a certain length. Note that sometimes grey grids are displayed. They can be removed by closing the image, then selecting first the horizontal bar plot and then again the logo.

For most graphical representations, general settings can be adjusted (font size, dpi and format (png, pdf, svg)). Furthermore, horizontal bar graphic may be viewed in color (default) or black & white (hatched). For heat-/cluster maps, values may be displayed either showing the identity between chimeras (percentage) or the number of mismatches (as absolute values). It is also possible to reorder the chimeras within the horizontal bar plot according to their similarity calculated from the cluster maps (cluster map must be generated first; default order is the order of the input FASTA-file). If peptides have been defined, they can be shown within the horizontal bar plots in addition to their position (optional). Barcodes are shown whenever defined.

For horizontal bar plots, indetermined regions may be resolved (default is “no resolve”) by 7 rules (R1-R7). For this, up to three parentals (named A, B, C) can be chosen, whereas correct notation is required (best way is via copy & paste from list of parental identifiers). A special resolution within a split horizontal bar plot can be performed for one chimera, which needs to be selected (clicked) first in the list “chimera identifier” on top of the GUI. All graphics are stored by default in the directory (“path to output files”) with a (hopefully) self-explanatory filename. This default saving mode can be reversed by unchecking the box “save graphics”. Because the pathname to this directory is generated from the pathname of the input FASTA-file, existing files will be overwritten without warning - so if

running the program twice with the same input file (or an input file with a name identical with a previous used file), rename files to prevent overwriting!

Beside the already mentioned output files, other so called json-files are generated and stored. These files can be easily read into python data structures of custom programs to perform other computations.

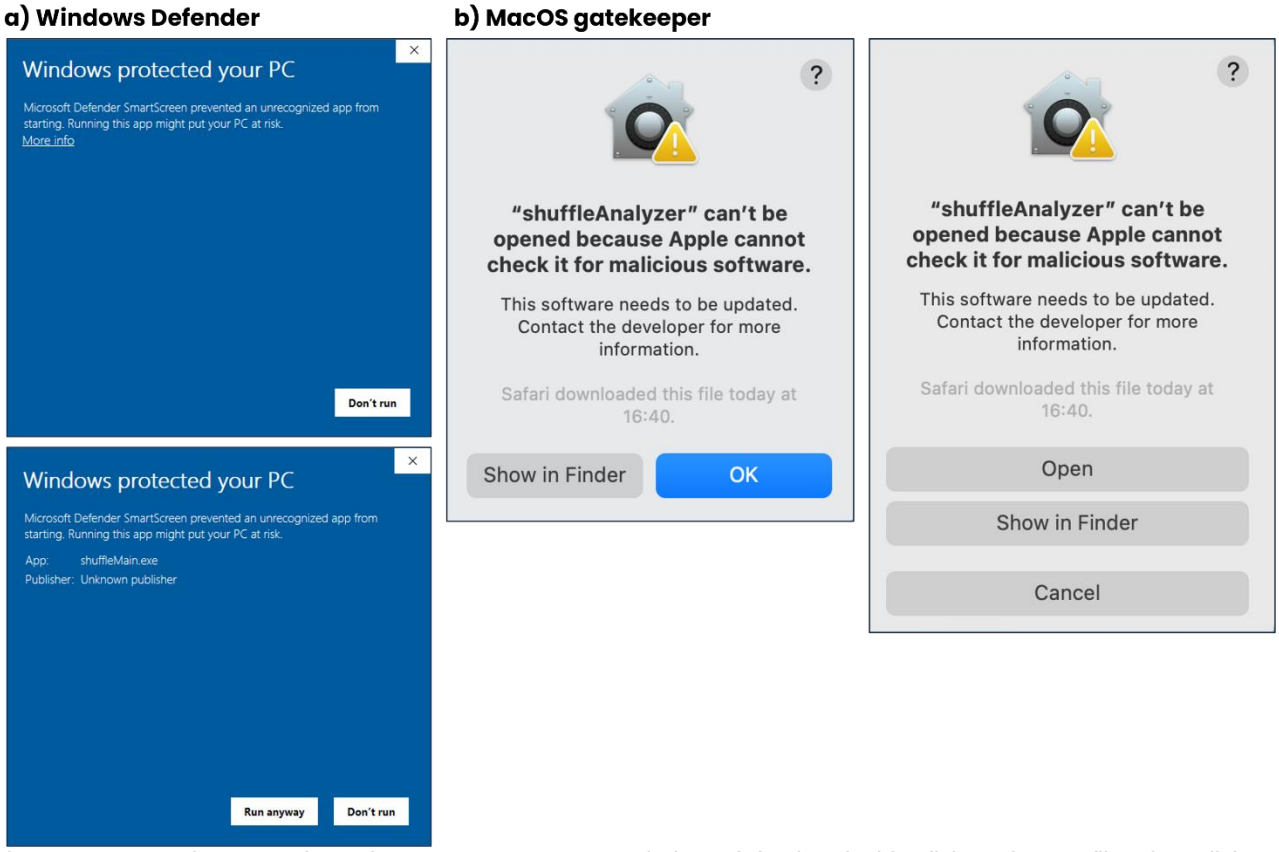

Figure 1. Overcoming protective software. A: To overcome Windows defender, double-click on the exe-file. Then click on "More info" (top image) and on "Run anyway" (bottom image). B: To overcome Mac Gatekeeper, right-click on the icon and select "Ok" in pop-up window (left image). Next, right-click again on the icon and click on "Open" (right image). Note that launching the software for the first time may take a while. However, afterwards the software can be launched simply by double-clicking on the icon.

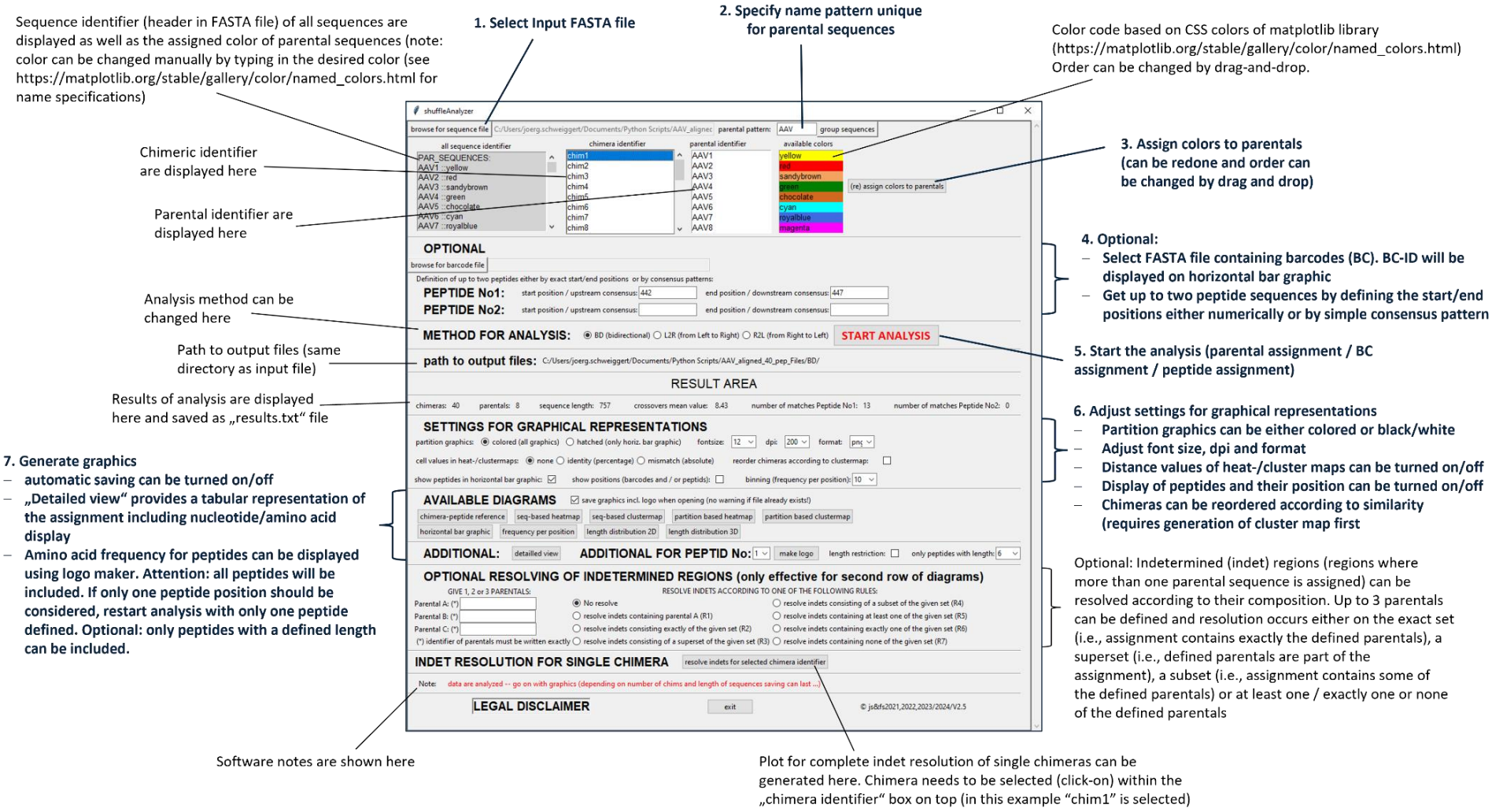

Figure 2. ShuffleAnalyzer GUI with descriptions

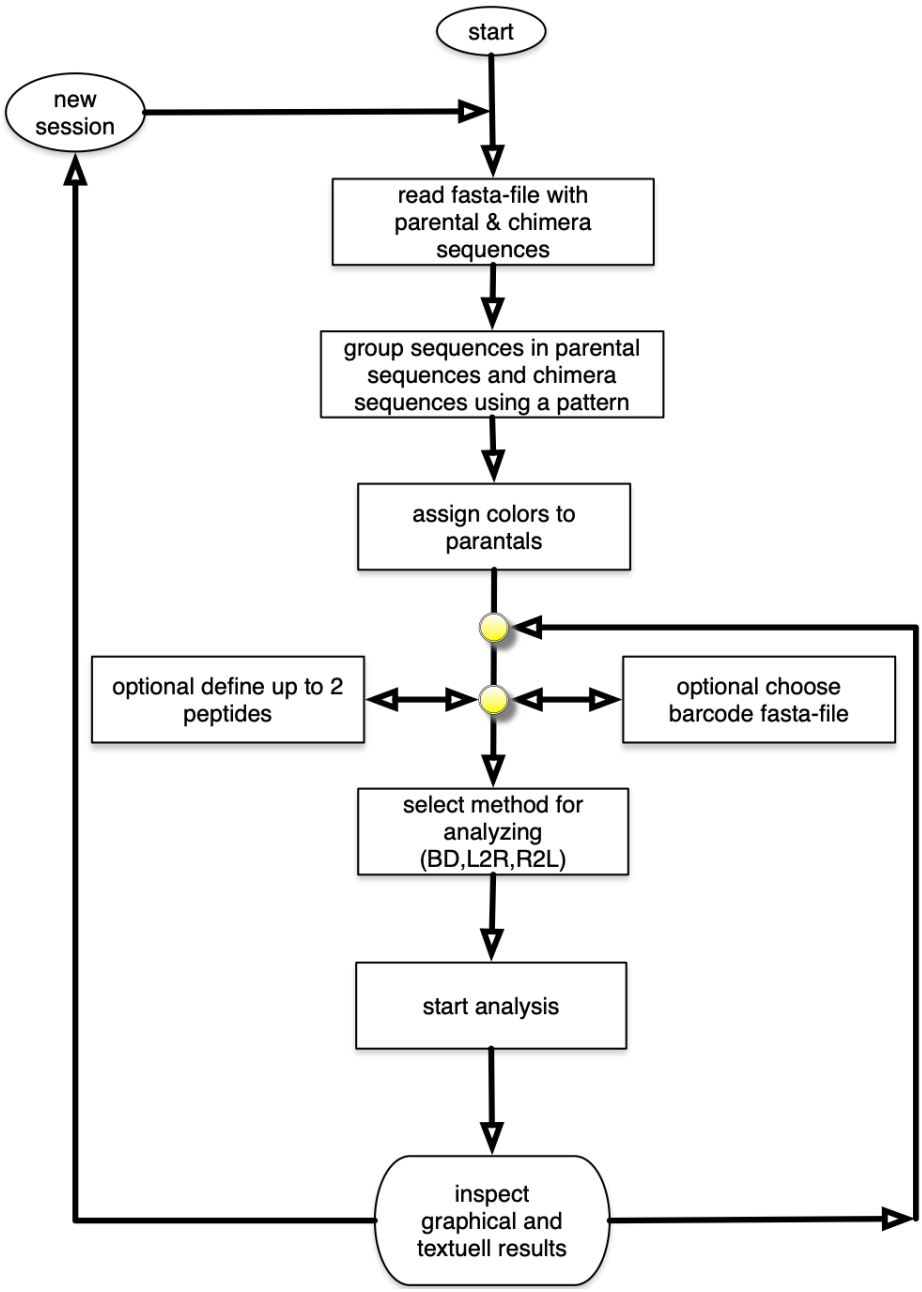

Figure 3. Flow chart of using shuffleAnalyzer

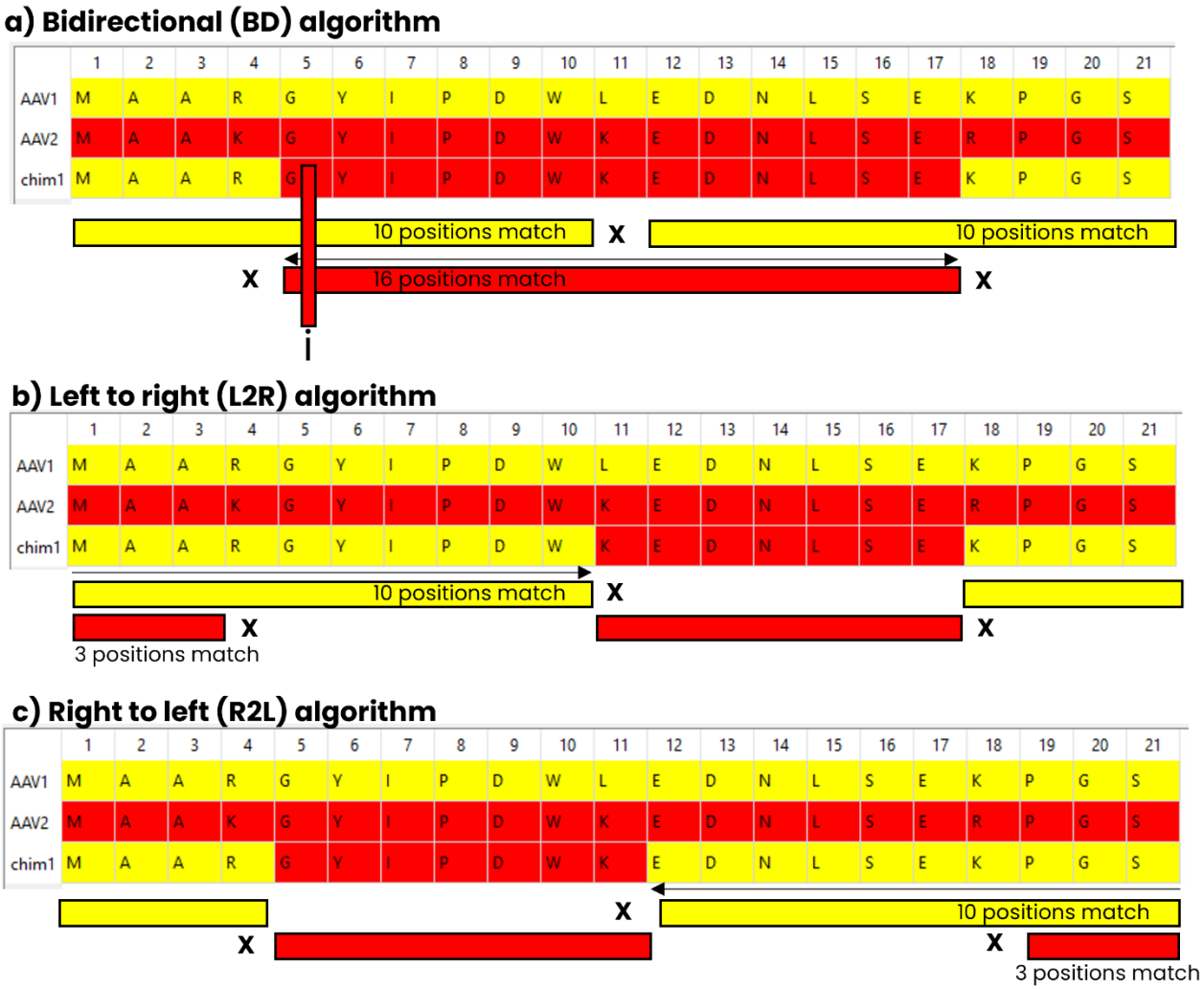

Figure 4. Exemplified analysis of the three different algorithms