SSR-viz - a toolbox to detect and visualize protein subfamily specific residues

Paul Zierep

August 1, 2018

Abstract

Protein families can often be further divide into functional diverse subfamilies. Each subfamily posses very specific functions which distinguishes them from each other. Examples are substrate specificity, protein-protein interaction or different reaction types. In many cases these functions are based on a limited set of residues. Therefore, to understand the protein diversity, the identification of those residues is crucial. In order to support researchers in this task we developed a toolbox which allows to detect and visualize those residues, based on a multiple sequence alignment of proteins with experimental validated functionality.

Contents

1 Install

add paths to the install

SSR-viz in implemented as a standalone GUI framework. It is entirely written in Python 3 and therefore the easiest way to obtain it is trough PIP - the official python repository. For Debian bases systems and Ubuntu:

```
pip3 install ssrviz
```

Should be enough ... in general. Some packages, especially **wxPython** which are normally automatically installed by PIP can make problems, as they are using system dependent, which can lead to errors when the dependencys are missing.

Good advice to install wxPython can be found at: `https://wxpython.org/pages/downloads/index.html`. They also supply custom builds which work on Ubuntu 16.04 and various other Linux systems.

Nevertheless wxPython needs some system dependencys on Linux: They can be easily installed for Ubuntu 16.04 the packages would be:

```
sudo apt-get install libwebkitgtk-dev libgtk2.0-dev libsdl1.2-dev \
freeglut3 freeglut3-dev libnotify-dev libgstreamerd-3-dev \
libgl1-mesa-glx libglu1-mesa libgl1-mesa-dev libglu1-mesa-dev \
libgconf2-dev libsdl1.2-dev zlib1g-dev libjpeg62-dev
```

In some cases the **matplotlib** might also need some Help with **tkinter**.

```
sudo apt-get install python3-tk
```

Additionally we implemented standalone executables for Windows (tested on windows 10) and Linux (tested on Ubunt 16.04). These a much bigger then the pure python module, but ship everything needed out of the Box.

The only external tool needed is mafft, an excellent alignment tool which is required to map protein structure indices to the alignment. SSR-viz runs without mafft, but for the **Add_pdb** tool the mafft executable needs to be assigned (see section 5).

Dnt't worry mafft is easy to install:
`https://mafft.cbrc.jp/alignment/software/`

2 Getting started

The SSR-viz algorithm is based on a multiple sequence alignment (MSA) file in FASTA format, which can be generated with various tools, such as Clustalo and Mafft or with a Webserver such as (). cite and add server

The topic of sequence alignment is beyond the scope of this manual. Nevertheless one should keep in mind that the quality of the alignment is crucial for the detection algorithm. (Is is difficult to interpret the importance of a position, which has more gaps then amino acids.)

The first step is the classification of the sequences into subfamilies. This is undoubtedly the most difficult part, as it often requires to identify the specific functionality based on scientific literature or even undertake experimental validation. Dedicated databases such as can help to identify detailed protein add some DB functionality.

Even though there are various tools available that can cluster protein sequences, these clustering methods always apply some kind of similarity scoring, which leads in most cases to a clustering based on evolutionary relationship rather then functional similarity. This is demonstrated on an example in section 6.

Ones you collected the class information of your sequences you can add them to your alignment. The **CSV_Builder** tool allows to creates a comma separated value (CSV) file which can be used to add the class label to the sequences (see section 3).

An alignment and the CSV file is everything thats needed to detect subfamily specific residues in the sequences. The **SSR_plot** tool handles the actual execution of the detection algorithm, the output can be a mathplotlib style plot (see section 4) as pdf, a Javlview annotation file (which can show the results together with the alignment) as well as a 'stats.csv' file which summarizes the SSRs.

cite

In many cases it is desired to observe the SSRs inside a protein structure (if available). Therefore, we also developed a tool **Add_pdb** which allows to map the indices of a protein structure file (*.pdb) to the indices of the alignment in the "stats.csv' file (see section 5)

An overview chart which explains the setup of the three tools is shown in figure

figure

3 CSV builder

The **CSV_Builder** handles the input and takes care, that the alignment and CSV class label file have the right formating.

The mapping scheme is shown in Fig. .

fig

3.1 Arguments

`Input sequence alignment file`

The alignment file with the sequences of the family. The deisred format is in FASTA fromat (clustalo), see section 6 for an example.

`Inplace FASTA conversion / Temporary alignment file name`

The **CSV_Builder** routine will remove duplicates from the alignment, as multiple identical sequences will overestimate the importance of this subfamily. The alignment can be converted inplace, meaning the original alignment is overwritten or a new alignment can be created.

`Regex extraction of the class label`

The normal **CSV_Builder** routine will create a CSV file, with an empty column for the class labels. Which must be manually filled. In some cases the class label is part of the sequence names, this labels can be extracted using regular expressions (regex) patterns. The entire scope of regex is to big for this manual, but the set of examples in the appendix 7 should help to get started. There are various tools available which can be used to test regex before usage, most text editors support regex as a search option. You can for example load the alignment file into sublime or notepad, then search with regex and if the pattern is correct, it should only highlight the desired class label. An example is shown in appendix 7.

`Output`

The name of the CSV file, by default it will be created in the same folder as the sequence file.

`Delete`

Allows to overwrite existing CSV files.

4 SSR plot

The

4.1 Algorithm

4.2 Arguments

4.3 Output

5 Add pdb

5.1 Arguments

6 Example

7 Appendix

7.1 Regex examples