KnotPull workshop June 24, 2021

Bartosz Ambroży Greń

Short introduction

KnotPull github repository: https://github.com/dzarmola/knot_pull. According to this page: "KnotPull reduces a user provided 3D structure, to simplify it, while preserving the topology of the chain. It has been successfully used for knot detection in proteins and chromatin chains."

For chain reduction KnotPull uses KMT algorithm https://aip.scitation.org/doi/10.1063/1.460889.

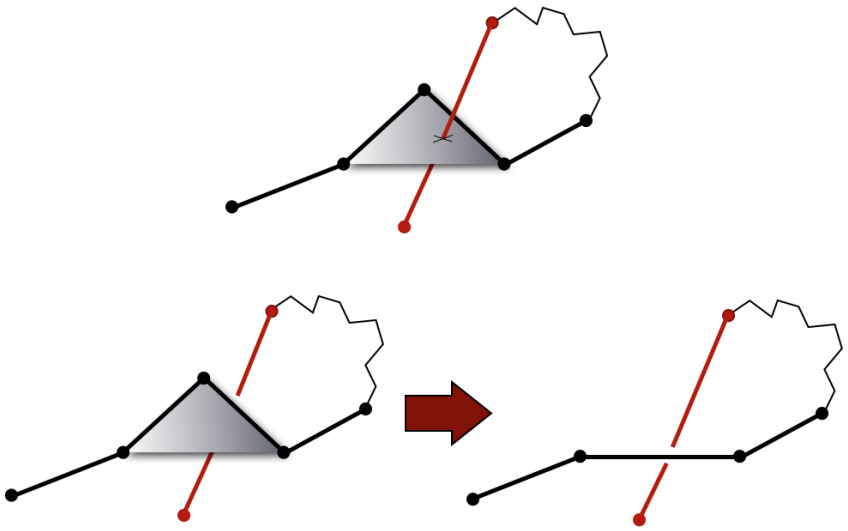

Figure 1. If the triangle defined by any three sequential vertices is not crossed by any bond, then the middle vertex is removed and these two left vertices are directly connected.

Installation

As KnotPull is distributed as Python (version ≥2.7) package it can be found on Python Package Index https://pypi.org/project/knot-pull/ and installed using `pip`:

```
pip install knot_pull
```

For full functionality installation of PyMOL is advised.

Simplest usage

- Lets start with the simplest, default usage:

  `knot_plot_check 2efv.pdb`

- compare the output with output of:

  `knot_plot_check -k 2efv.pdb`

Option `-k` calculates also knot type. Notice that final chain is shorter than without `-k` flag. It is due extra reduction with Reidemeister moves.

- Lets check what happens when we input a multi-chain structure:

  `knot_pull_check 3i40.pdb`

- To limit your calculations to a single chain use `-c` argument and specify the chain. If you want check more chains add them after a comma.

  `knot_pull_check -c A 3i40.pdb`

Visualization

- For visualisation of reduction process we need to output our structure to a file using argument `-o` with a name of output file. Besides fully reduced structure the file will also have intermediate steps.

  `knot_pull_check -o output.pdb 3i40.pdb`

- Then to visualize with PyMOL type:

  `knot_pull_show output.pdb`

  You can look at the frames one by one or with a looped animation.

- As you can see, there are two chains being reduced. If you want only one chain, then combine arguments `-c` and `-o`

  ```
  knot_pull_check -c A -o output.pdb 3i40.pdb
  knot_pull_show output.pdb
  ```

- You van try with other files: `2efv.pdb`, `2m37.pdb`, `3bjx.pdb`, `4mcb.pdb`, `4m4s.pdb`, `chromosome_f.xyz`, `chromosome_h.xyz` – xyz files are accepted too!

Proposed routine

- ```
  knot_pull_check -k -o -c A file.pdb
  knot_pull_show file_out.pdb
  ```

  First function will smooth and analyze topology of chain A and generate `file_out.pdb` file, which can be an input for visualisation in PyMOL. Note that default name of output file is same as input file extended by `_out` before the extension.
  ```

Extras

- You can also load .xyz file: try with `chromosome_f.xyz` and `chromosome_h.xyz` structures.
- For minimal output on screen use `-q` flag.
- You can ignore parts of the and of the chain with `-b` and `-e` arguments. E.g.:

  ```
  knot_pull_check -b 21 -e 60 2efv.pdb
  ```

  will ignore first 20 residues and residues after 60th residue. It you want to specify by residue ID, then add `i` before number:

  ```
  knot_pull_check -b i21 -e i60 2efv.pdb
  ```
- `knot_pull_check -h` prints quick help with all possible arguments.