**Slipknots:**

Slipknots are a different class of knots which rarely occur in the protein models. They are different when compared to the traditional knots and detecting them is a tedious task. It is also different from detecting the commonly found knots.

Slipknots are knots in which the knotted protein chain doubles back on itself and mathematically gets untied. If we assume to hold both N and C terminals and pull from both the ends[[1]](#endnote-1), the knotted chain would get unknotted and would simplify into a straight line. But, a knot is still present in the chain and is affecting the energy of the protein model, even these should be processed and removed to have a protein with most minimum energy or a protein with the correct fold. Slipknots are not mathematically knots and hence the normal knot detection algorithms cannot report them, even Knotfind cannot report them when the entire backbone is passed to it at once. The Knotfind algorithm can simplify all the atoms in the chain other than N and C termini and would report no knots when the entire chain is passed. However, proteins with slipknots have a knot in them which when processed in subsections can be reported using knotfind.

We leverage this property of slipknots to find them and report them. We extend the already existing knotfind algorithm to detect slipknots. We divide the entire protein chain into small subsections and iteratively run the knotfind algorithm on these subsections and increase the size of the chain after every pass until all the residues are checked. While processing, slipknots are mathematically knots until it finds that one atom (which we mark as the k1 atom) which unties the whole chain and makes the chain unknotted.

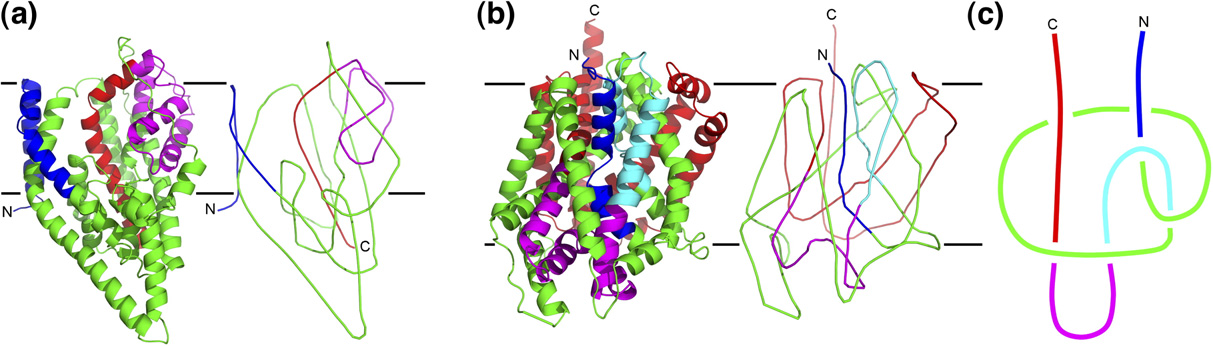
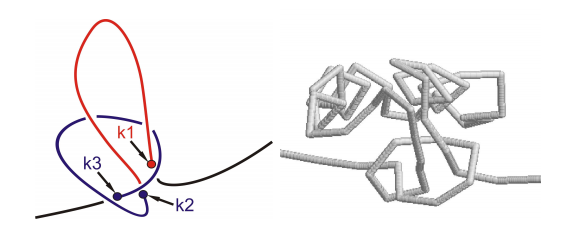


Figure .1: Simplifying the protein chain to highlight slipknot2.

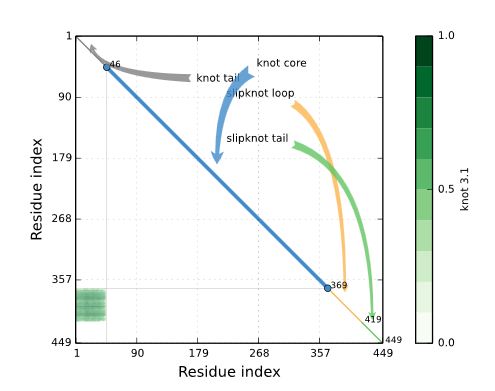
The transition from image A to C in the above figure[[2]](#endnote-2) represents the simplification of the protein chain to have just the alpha carbon atoms after smoothening the backbone of the initial structure.In the above image, in section C, we can notice that pulling both N and C terminals would untie the chain and make the whole protein unknotted. These types of knots go undetected when we run traditional knot find algorithms on them.



**Figure .2:** The atoms which define the slipknotted region3.

The above image[[3]](#endnote-3) defines the points which form a slipknot. Atom k3 to k2 form the knot and continues to be knotted until k1 is found which unties the whole chain and makes the protein chain unknotted.

**Slipknot Classification:**



**Figure .2:** KnotProt for 1ALK chain A4,5.

A slipknotted protein chain is classified into four different regions and each of them represents a part of the knot. Following are its details:

* Knot Core[[4]](#endnote-4),[[5]](#endnote-5): The shortest sub chain with a knot.
* Knot Tail: A segment between one terminal and the Knot core.
* Slipknot Tail: The longest segment starting at one terminal, for which no change in topology is detected.
* Slipknot Loop: The segment between knot core and the slipknot tail.

**Extending Knotfind to detect Slipknots:**

We extend the knotfind algorithm to find slipknots, we run knotfind algorithm on subsections of the protein chains. We start the processing with only three atoms in the chain, Knotfind would return false with a chain size of three as no line segment could possibly intersect a line drawn to connect these three atoms. After every attempt of knotfind we increase the size of the chain by one, as in we append the next available cα atom in the pool to the sub chain which we are considering to process slipknots, and run the knotfind algorithm on it.

For any knotted or slipknotted sample, we first focus on getting the k3 and k2 atoms which define the knot core for any knotted sample. If no knot core is found during the entire process it is evident that the sample is not knotted or slipknotted. If the knotfind algorithm returns a knot for any given subsection of the protein we report the k3 and k2 atoms and continue increasing the size of the chain till ‘n’.

Once a knot is found, the algorithm checks until what length the algorithm would return true. If at any point the algorithm returns false, then we mark that atom which removes the knot as k1 atom and report it along with k3 and k2. And we mark the entire chain as slipknotted. The regions from k3 to k2 would form the knot core and k2 to k1 would be the slipknot loop. Atoms from one to k3-1 would be marked as knot tail and atoms from k1+1 to ‘n’ would be marked as slipknot tail.

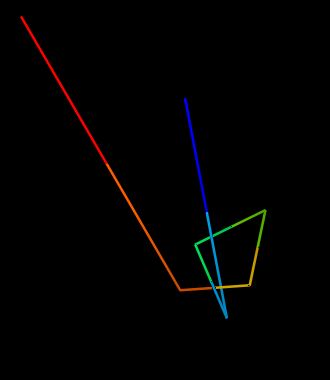
The algorithm logs all the residues which are unsimplified cα atoms from k3 to k1 onto a PDB file along with all the atoms which constitute the knot tail and slipknot tail. With the knot tail and slipknot tail we can visualize the slipknots better.

At the end of the analysis, based on what type of knot the protein has, we create PDB files with the corresponding atom data in it. The new PDB files will have only the cα data in it for knotted structures and knot tail and slipknot tail along with the unsimplified cα atoms for the slipknotted protein structures.

**Visualizing Knots Using Pymol**

The new PDB’s are loaded into PyMol for visualization. PyMol has an API of its own written in Python that we have used to highlight knots and slipknots. We wrote a python script to connect all the atoms and visualize them serially.

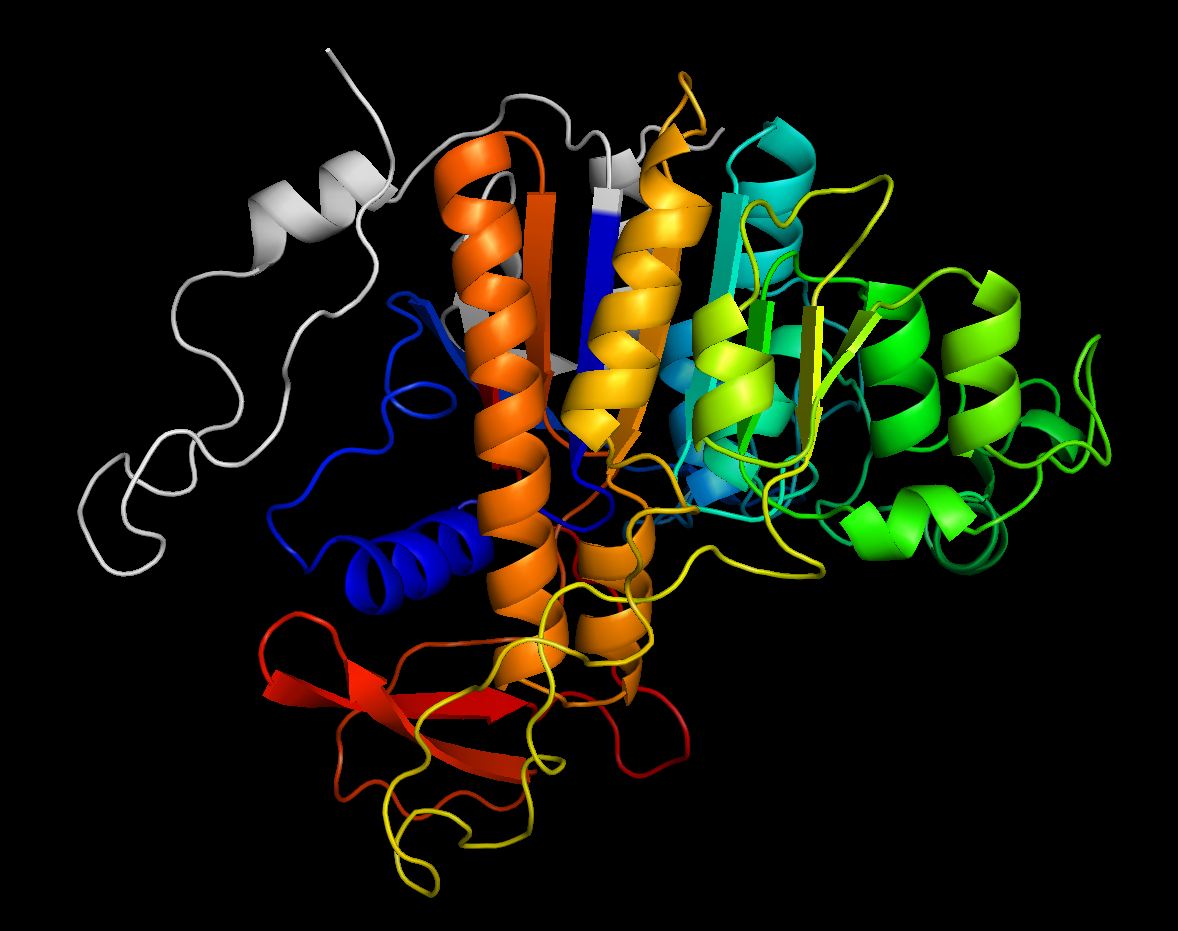
Here is an example of how a knotted protein would look like before and after applying the script:



**Figure 3:** Connecting atoms to highlight knotted region.

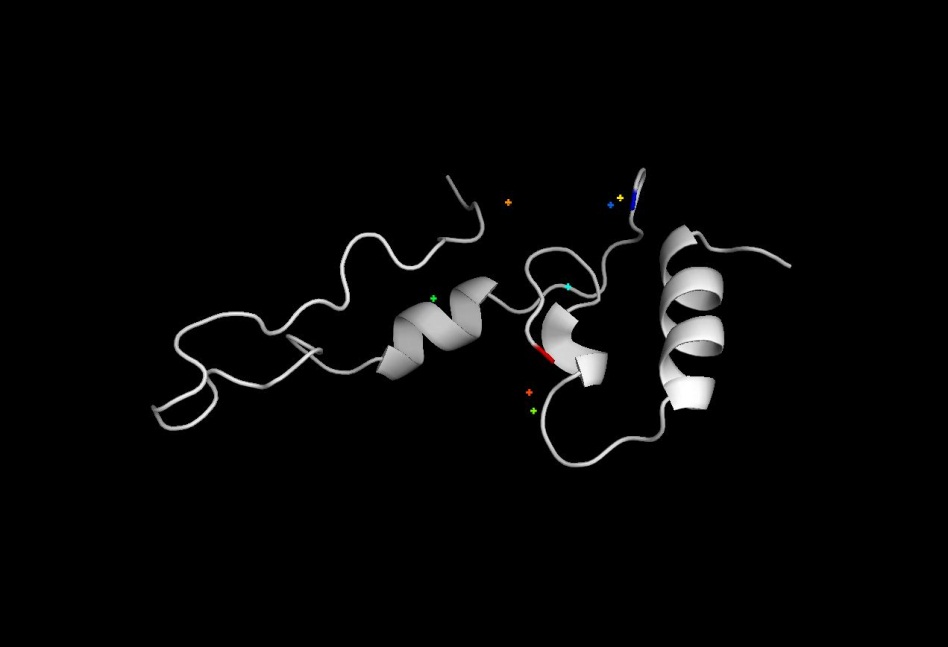
The above images are the unsimplified atoms of the protein 1js1. The knot can be visualized clearly in the second image after connecting the atoms and color them with spectrum coloring. Once the PDB is loaded into PyMol, the script has to be loaded along with it to connect cα atoms.

**Visualizing Slipknots using PyMol**



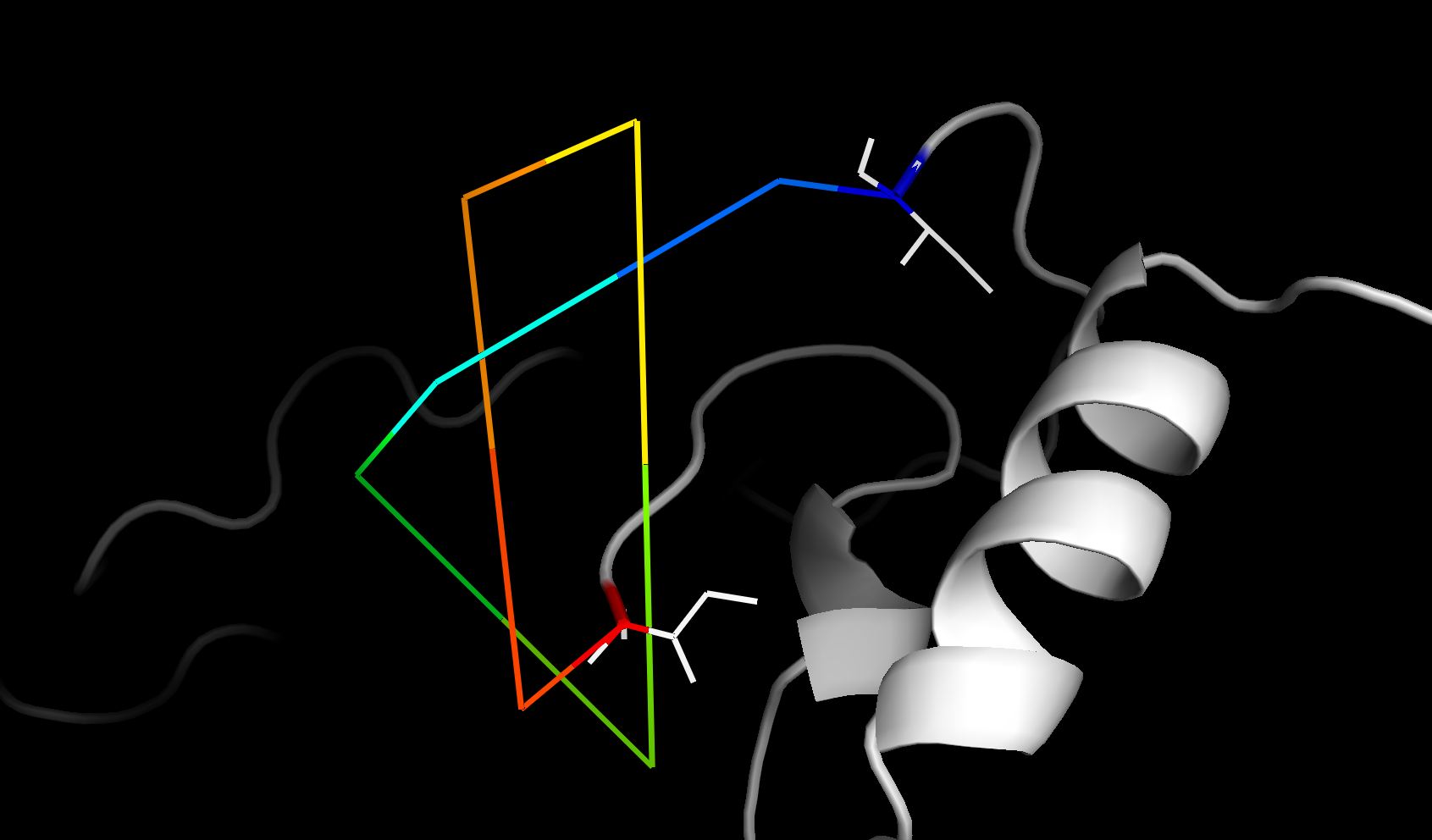
**Figure 4:** Knot Core of 1ALK.

The above image is an example of how a slipknotted region would look like. The colored region forms the knot core and the slipknot loop and the tails are marked in gray. It is very hard to figure out how the colored region would form a slipknot and is a tedious task. After we run the slipknotfind algorithm on the above protein, 1ALK, we get the following result:



**Figure 4:** Simplified knot Core of 1ALK.

The above image has a few atoms which are not connected. These are the atoms which define the slipknot here is what we get by connecting these cα atoms.



**Figure 5:** Atoms connected to highlight slipknot.

By connecting cα atoms, the slipknot in the above PDB becomes evident. And atoms from blue to through red define the knot core and the slipknot loop.

**Visualizing Knots and Slipknots using JSmol molecular visualizer**

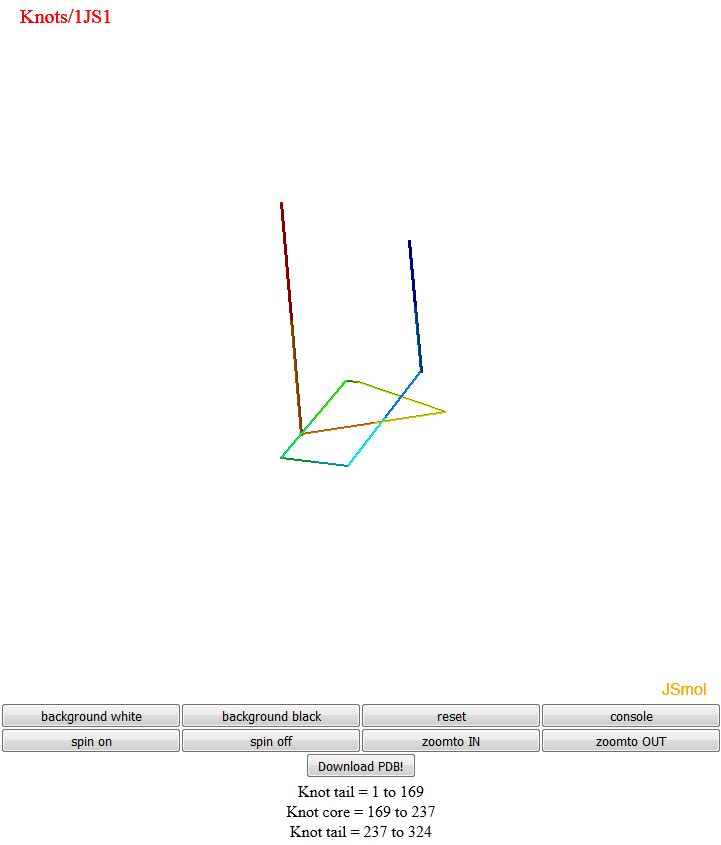
Knotfind and Slipknotfind were initially implemented using Java, using PyMol we could enable its reach only to PC’s, but we wanted biochemists and structural biologists to use Knotfind and Slipknotfind to process their protein structures to detect knots from anywhere across the globe so we came up with an idea to extend the abilities of Knotfind and Slipknotfind to a website that can be accessed from anywhere in the world with a stable internet connection.

The java class that was used to find the knots has been extended using a JSP page to create an interface to upload a PDB file. The uploaded PDB file is then passed to the Knotfind and Slipknotfind class which runs on the apache tomcat server along with the chain ID that has to be processed. Once the analysis is complete, the server generates a simplified PDB and passes it to the browser along with details that are needed to highlight the knot.

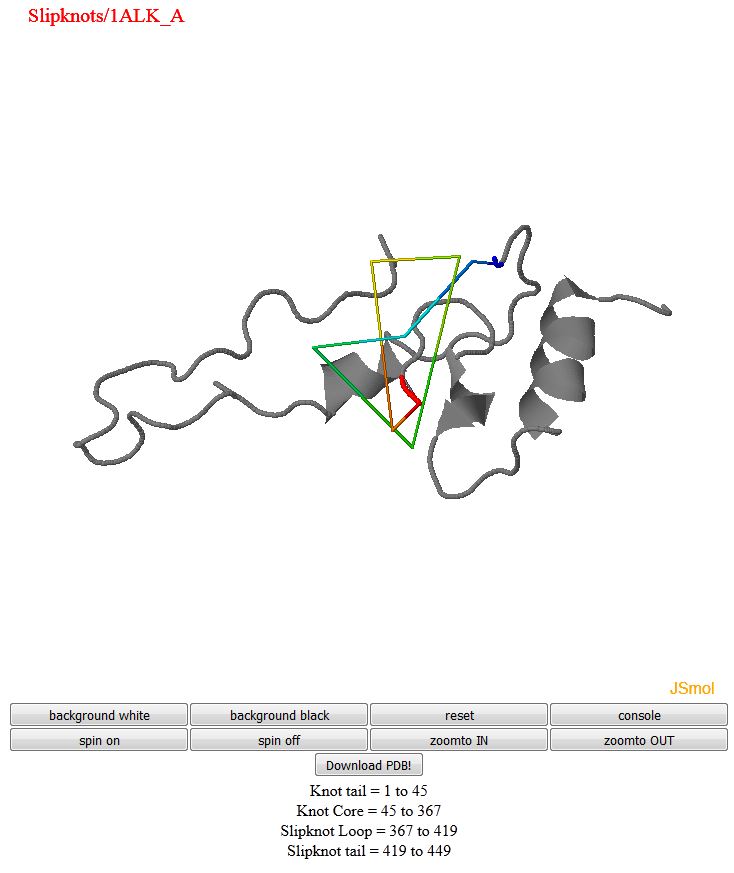
The connection between the server and the client is maintained using sessions and browser cookies. The pdb ID, server path, atoms which constitute the knot or slipknot are sent to the browser using the browser cookies.

JSmol a Javascript molecular visualization tool equipped with a library of its own is used to highlight the knot on the browser. JSmol is an extension of Jmol a Java based molecular visualizer, JSmol supports all the commands that Jmol console accepts.

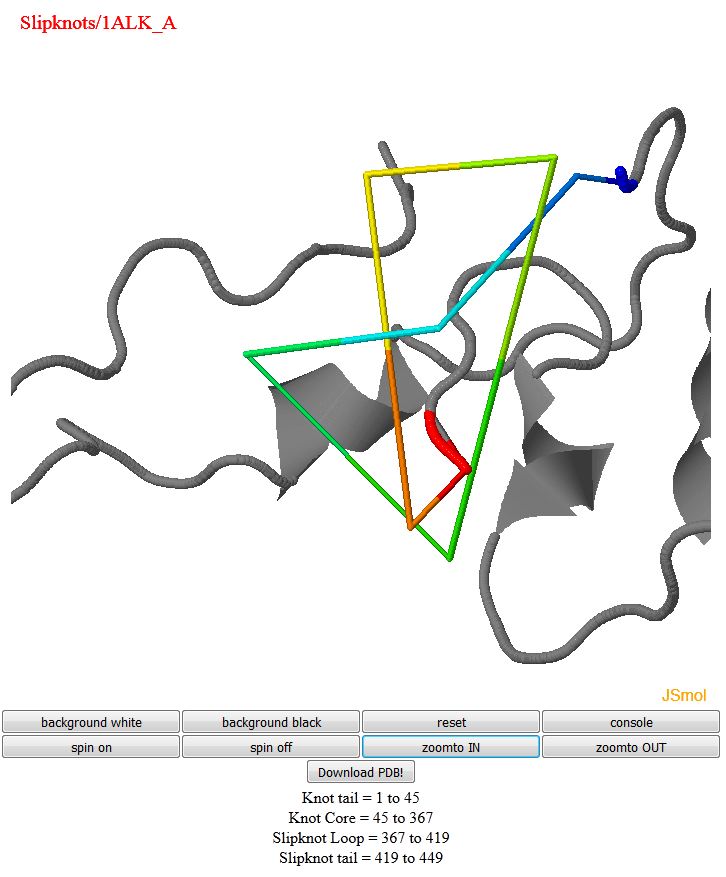
A JSmol script using JSmol API is loaded along with the PDB. The script initializes JSmol and connects the appropriate atoms based on the type of the knot in the given PDB and highlights the knot. The browser window also allows the users to call some basic JSmol functions to rotate or zoom into the PDB to have a closer look. The window also gives the knotting fingerprint of each protein after computing the knot.



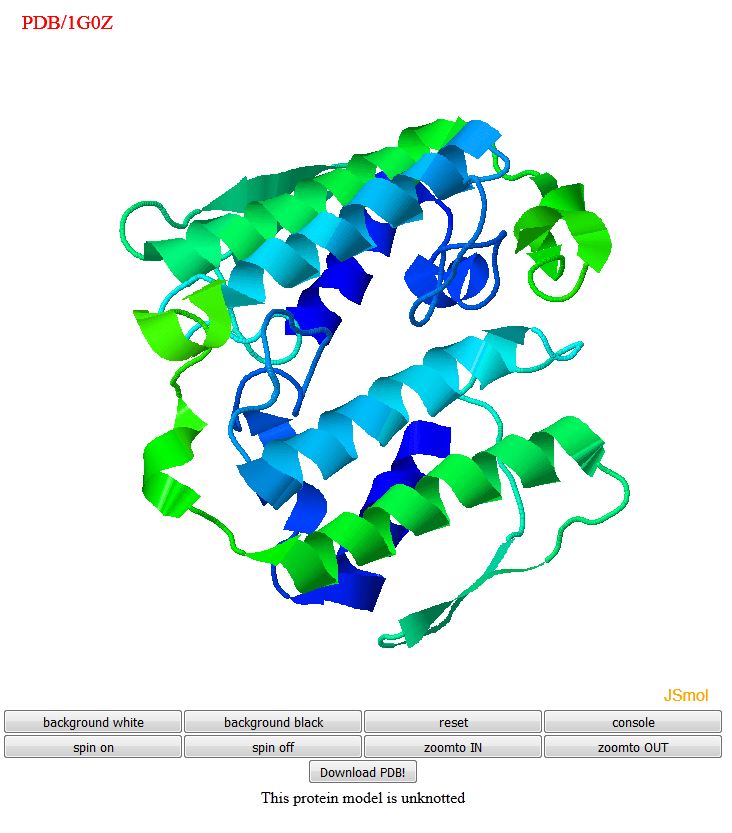
**Figure 6:** Highlighting the knot in 1JS1.



**Figure 6:** Slipknot in 1ALK.



**Figure 7:** Closer look of slipknot in 1ALK.



**Figure 7:** Closer look of slipknot in 1ALK.

If the uploaded PDB is not knotted, then the entire PDB is displayed in a three dimensional view with a message stating that the structure is not knotted.

1. Nureki,O. et al. (2002) An enzyme with a deep trefoil knot for the active-site architecture. Acta Crystallogr D Biol Crystallogr. [↑](#endnote-ref-1)
2. Neil P. King1, E. O. (2007). Identification of Rare Slipknots in Proteins and Their Implications for Stability and Folding. [↑](#endnote-ref-2)
3. Joanna I. Sułkowska, Piotr Sułkowski, and José N. Onuchic (2009) Jamming Proteins with Slipknots and Their Free Energy Landscape. Phys. Rev. Lett. 103, 268103. [↑](#endnote-ref-3)
4. Sulkowska JI, Rawdon EJ, Millett KC, Onuchic JN and Stasiak A (2012) Conservation of complex knotting and slipknotting patterns in proteins, Proc. Natl. Acad. [↑](#endnote-ref-4)
5. Jamroz M, Niemyska W, Rawdon EJ, Stasiak A, Millett KC, Sułkowski P, Sulkowska JI (2014) KnotProt: a database of proteins with knots and slipknots, Nucleic Acids Research. [↑](#endnote-ref-5)