

CS612 Homework Assignment 4

Due Mon. April 24, on Gradescope

1. **Representing a sequence as a matrix:** One thing you will have to do for your term project is encode a protein sequence as a set of binary vectors of size 21. This is called "one-hot encoding". You can use other representations, of course. Given the following sequence: "DDHHGFDYNGVMVV", express it as a binary matrix of size 14×21 , where 14 is the sequence size and 21 is the "alphabet" of amino acids one letter codes + space. Every position is a binary vector where The one-letter codes for all amino acids is: "ACDEFGHIKLMNPQRSTVWY-" (notice the dash in the end). You should attach your code and the end matrix. You can look here for help: <https://machinelearningmastery.com/how-to-one-hot-encode-sequence-data-in-python/>. There are also many R functions.

Here is a code for example, and the result:

```
aa_letters = ['A', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'K',
              'L', 'M', 'N', 'P', 'Q', 'R', 'S', 'T', 'V', 'W', 'Y', '-']

import numpy as np
import math

def seq_to_one_hot(sequence, alphabet=aa_letters):
    arr = np.zeros((len(sequence), len(alphabet)))
    aa_key = {l: i for i, l in enumerate(alphabet)}
    for j, c in enumerate(sequence):
        arr[j, aa_key[c]] = 1
    return arr

arr = seq_to_one_hot("DDHHGFDYNGVMVV")
print(arr)
```

```
[[0.  0.  1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [0.  0.  1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [0.  0.  0.  0.  0.  0.  1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [0.  0.  0.  0.  0.  0.  1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [0.  0.  0.  0.  1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [0.  0.  1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  1.  0.]
 [0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  1.  0.  0.  0.  0.  0.  0.  0.  0.]
 [0.  0.  0.  0.  0.  1.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  1.  0.  0.  0.]
 [0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  1.  0.  0.  0.  0.  0.  0.  0.  0.  0.]
 [0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  1.  0.  0.]
 [0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  0.  1.  0.  0.]]
```

2. AlphaFold: You will need the AlphaFold 2 colab notebook:

<https://colab.research.google.com/github/sokrypton/ColabFold/blob/main/AlphaFold2.ipynb>

Calmodulin is a relatively small (less than 150aa) multifunctional calcium-binding protein found in all eukaryotic cells. It acts as a calcium sensor, regulating a wide range of cellular processes by mediating the effects of calcium signaling. Several different conformations of this protein exist, as this protein undergoes large conformational transitions due to calcium binding. Conformational transitions are not easy for AlphaFold to model.

- (a) Look at the AlphaFold model for this protein: <https://alphafold.ebi.ac.uk/entry/P0DP23>. Which regions of the protein model have low or very low confidence? (you can get the info by hovering the mouse over the region).

These are residues 1–5, 74–82 and 148–149. These are the termini and the middle of the long helix connecting the two sides of the protein.

- (b) Two calmodulin conformations, 1CLL and 1CTR, represent the open and closed conformations, respectively, using the pdb alignment module:

<https://www.rcsb.org/alignment> like in a previous homework assignment, using the AlphaFoldDB entry P0DP23 for one entry, 1CLL and 1CTR for the other two. Always make sure that the AlphaFold model is the first, as it is the reference. Use chain A in all cases and hit "compare". What is the RMSD between the AlphaFold model and each of the two calmodulin structures? Attach the screenshot of the alignment results.

The RMSD of the AlphaFold (orange) with respect to 1CLL (blue) is 3.25Å, and with respect to 1CTR (green) is 2.34Å. However, it is important to notice that the AlphaFold model is much more similar to 1CLL – the open conformation. Notice that the RMSD only applies to the amino acids that were aligned, not the overall protein. In this case, more reliable measures would be the number of aligned residues – 95 and 71 for 1CLL and 1CTR, respectively, and the TM scores, which are 0.58 and 0.47 for 1CLL and 1CTR, respectively.

Pairwise Structure Alignment

This tool allows the selection of protein 3D structures for alignment. Use an existing PDB or [Computed Structure Model](#) entry ID, upload a local file with atomic coordinates, or enter a URL of a file on the web

[Alignment API](#)

Compare Protein Structures

Entry	Chain	RMSD	TM-score	Identity	Aligned Residues	Sequence Length	Modeled Residues
AF-P0DP23-F1	A	-	-	-	-	149	149
1CLL	A	3.25	0.58	80%	95	148	144
1CTR	A	2.34	0.47	95%	71	148	141

Sequence Alignment in 3D

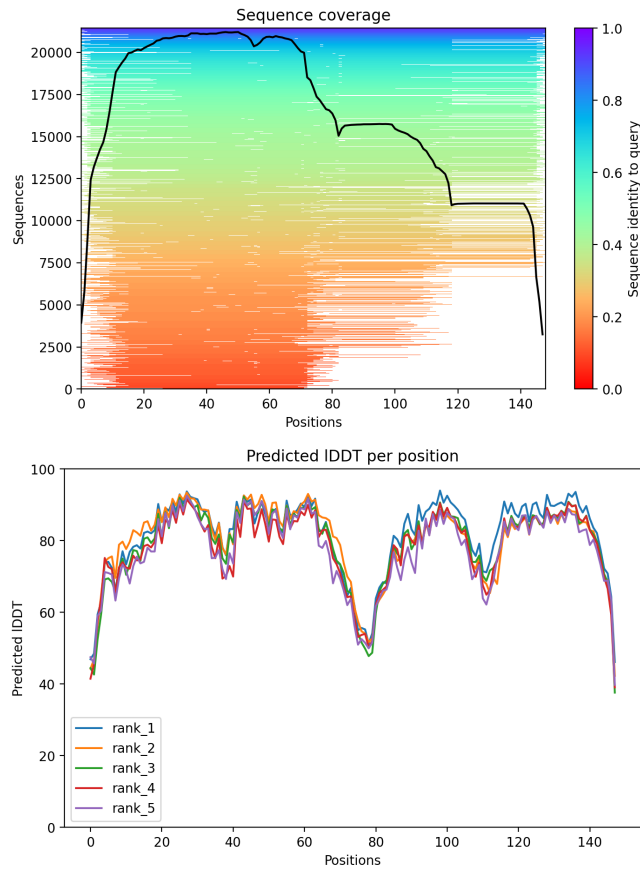
[Export](#)

[Copy Link](#)



- (c) Paste the 1CTR (closed conformation) sequence at the AlphaFold sequence window: <https://www.rcsb.org/fasta/entry/1CTR/display> (only the second line).

Modify the num_recycles to 1 and the max_msa to 16:32. Hit "Run all" at the runtime tab above. Wait patiently, it takes a few minutes to finish. The results will be downloaded to your computer as a zip file. Unzip and attach the figure ending with coverage.png and the one ending with plddt.png to your submission. These images indicate the sequence coverage and the confidence in the prediction per position. What is the area with the lowest confidence? Does it coincide with your answer to (a)? The lowest confidence areas are similar to (a) – the termini and the region around 80, where the middle of the helix is.



- (d) Upload the first/top model – it is a pdb file that has *model_1* somewhere in the name. Repeat the alignment process from part (b). Attach the screenshot and report the RMSD of the first model with respect to 1CLL and 1CTR. Which of the two PDB structures are more similar to the AlphaFold model? The open (1CLL) or closed (1CTR)? Don't let the RMSD numbers confuse you – look at the TM score and the entire structures.

As before, the model is similar to the closed conformation, 1CTR. The RMSD with respect to 1CLL (blue) is 3.81Å, and with respect to 1CTR (green) is 2.37Å, but as before, the number of aligned residues is 95 and 78, respectively and the TM scores are 0.62 and 0.52, respectively. Not great, but the resemblance to 1CLL is slightly better. Notice the results are not very good because of the small number of sequence information fed to AlphaFold, among others.

Pairwise Structure Alignment

This tool allows the selection of protein 3D structures for alignment. Use an existing PDB or Computed Structure Model entry ID, upload a local file with atomic coordinates, or enter a URL of a file on the web

Alignment API

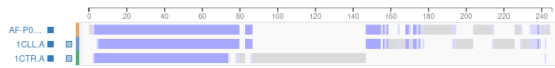
Compare Protein Structures

Entry	Chain	RMSD	TM-score	Identity	Aligned Residues	Sequence Length	Modeled Residues
AF-P0DP23-F1	A	-	-	-	-	149	149
1CLL	A	3.25	0.58	80%	95	148	144
1CTR	A	2.34	0.47	95%	71	148	141

Sequence Alignment in 3D

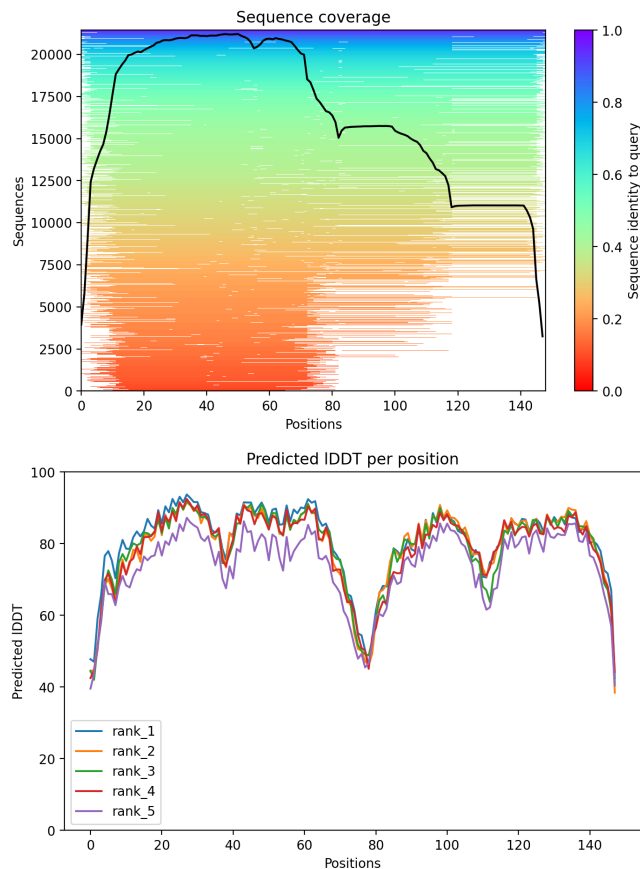
Export

Copy Link



- (e) Repeat part (c) but change the msa_max parameter to 64:128. This means you will use many more sequences, hoping to get a better modeling. It will take you a bit longer to run.

The lowest confidence areas are again similar to (a) – the termini and the region around 80, where the middle of the helix is.



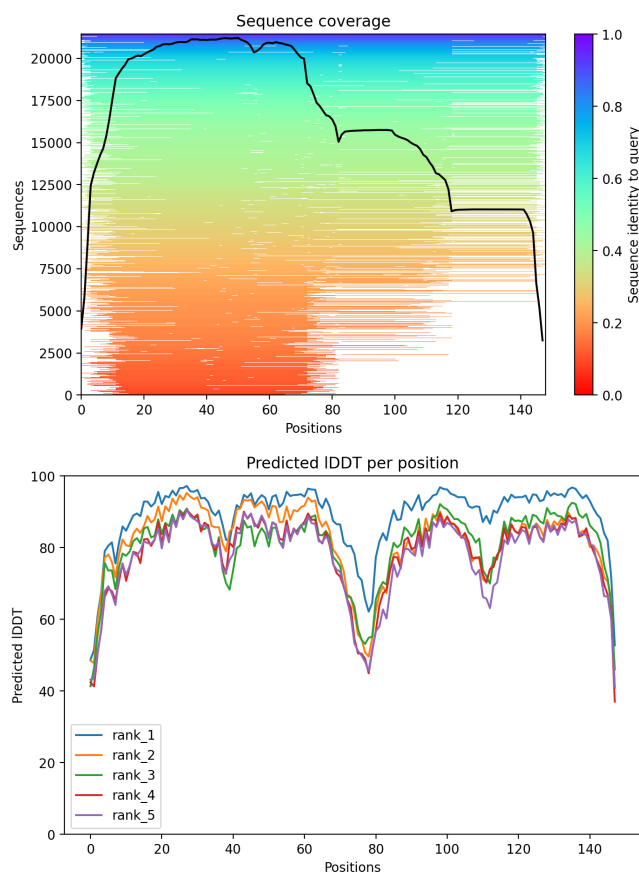
- (f) Repeat part (d) – what are the RMSD and TM scores now, with respect to the open conformation

1CLL and the closed on, 1CTR? Do you see a big difference?

The results are actually the same. This means that in this case the number of sequences probably don't make a lot of difference. Since there are many different structures for similar sequences, one probably needs a template.

- (g) Finally, you may use a structural template to help the modeling. We won't use 1CTR (too obvious). Rather, let us use PDB 1iq5, which is the closed conformation of calmodulin from a frog. Download the file from the PDB website at <https://www.rcsb.org/structure/1IQ5>. On the AlphaFold window, leave all other parameters as-is, but change the template_mode parameter to custom. Hit run all. It will soon prompt you to upload a template. Upload the 1iq5.pdb (or 1iq5.cif) file and wait for the run to finish. Repeat part (c).

The coverage and confidence score are quite similar to previous experiments.



- (h) Repeat part (d) above – compare the top model to 1CLL and 1CTR. What are the RMSD and TM score now? Does the model look more like the closed or open conformation?

Now, finally, the results are more similar to 1CTR, the closed conformation. As in previous questions, do not only look at the RMSD. The RMSD with respect to 1CLL and 1CTR is 1.98Å and 3.7Å, respectively, but the number of aligned residues is 77 and 113, resp. (even better than before!) and the TM scores are 0.51 and 0.65, respectively, which is even better than the AlphaFold model in part (a).

3. Transformations on Molecules:

- a. Read the PDB file 2EZM from the PDB website. Leave only the C,N and CA atoms (you can either edit the file or use the biopython code). These are only the main backbone atoms (except

the carboxyl O but it's not important for the dihedral calculations below). Calculate the backbone dihedral angles (ϕ and ψ) for amino acid 10. **Answer:** $\psi = 35.70$ and $\phi = 56.20$. See solution code.

- b. Calculate the energy for the structure above with only the backbone atoms. Use a simplified energy function that counts only steric clashes between atoms. The energy is the sum over all the pairs of atoms that are at least 4 atoms apart on the peptide chain (that is – do not share a covalent bond, a planar angle or a dihedral angle). The energy for each pair of atoms is:

$$E_{ij} = \begin{cases} 100/r_{ij}^2 & \text{if } r_{ij} < 3.4 \\ 0 & \text{Otherwise} \end{cases}$$

Where r_{ij} is the distance between two atoms. In this (simplified) function we model each atom as a hard sphere with a radius of 1.7\AA , hence the 3.4 – two atoms collide only if their distance is smaller than the sum of their radii. Output the total energy and also report the pairs of severely clashing atoms, whose contribution to the energy is at least 20, and their distances.

Answer: $E=255.96$ and there were no severely clashing atoms this time. See code.

- c. Rotate the dihedral bond between the N and CA of the last amino acid by 30° . You can use the `internal_coords` module if you want. Does the structure change much? What atoms move in space as the result of this rotation? What is the RMSD between the original and the modified structure? You can measure the RMSD the code you wrote for HW3.

To view the structures you can convert the transformed coordinates back to a PDB file, and view them using Biopython.

Answer: The structure changes very little. Only one atom moves (the last one). The energy is the same and the RMSD is very small – 0.043\AA or so. See two structures superimposed. Notice the only difference is the last atom. To obtain the figure I saved the results of the new matrix as a `crd` file and uploaded it with the `pdb` file as a second frame. It makes the RMSD calculation easier. Do NOT use the `calc_dihedral` function in `biopython` – you may use it for validation only. You may use the Matlab, R or the Python `numpy` library.

- d. Now rotate the dihedral bond between CA and C of amino acid 40 by 20° . What is the RMSD between the original and the modified structure now? Attach an image of the superimposed structures as part of your homework.

Answer: This time the change is much bigger, because we rotate more than half the atoms. There are several severe clashes (see below). The energy is 1213.77 or so. The RMSD is 1.355.

- e. Calculate the energy for the two structures you created in c and d above.

Answer: See above.

General Setup

To calculate the backbone dihedral angles of residue 10 you will need the coordinates of 5 atoms: C^9 , N , CA , C , N^{11} . You can use this file as input to R or Matlab (see setup from previous homework assignment for help).

Common troubleshooting:

- Notice that $v_1 \times v_2 = -(v_2 \times v_1)$, so make sure the order of the vectors is correct.
- Make sure your vectors are normalized when calculating the angle between them.

Setup for Matlab

Matlab offers a lot of matrix operations that you would otherwise have to implement yourself. We provide some hints/directions for those of you who choose to implement this assignment in Matlab. The very first step you need to do is to read from an ASCII file that contains the cartesian coordinates of the chain you will manipulate and save these coordinates in a data structure that will represent your chain. The simplest data structure at this point is an array, where positions $3 * (i - 1) + 1$, $3 * (i - 1) + 2$, and $3 * (i - 1) + 3$ contain the coordinates of atom i . Note that Matlab starts counting from 1. You can read the coordinates into Matlab using the command:

```
cartesian_coordinates = textread(input_file, '%f');
```

Where the input file contains the coordinates. You can write a simple program that leaves in only the coordinates or save the PDB file in crd format on VMD, and delete the first dummy line, or use the awk command as above. You can check how many coordinates you have read with the command `length(cartesian_coordinates)`. The `textread` command stores all coordinates read from the backbone.crd file in the cartesian coordinates array. You could manipulate this array with the dihedral rotations you will define. However, it might be more convenient for matrix operations and for clarity to store the cartesian coordinates in a matrix with N rows and 3 columns for each row. In this way, row i contains all the three cartesian coordinates for atom i . Define the number of atoms in the backbone with the command:

```
N = length(cartesian_coordinates)/3;
```

Now you need to declare a matrix with the right dimensions. Since you actually need to work with homogeneous coordinates, it might be convenient to declare a matrix with N rows and 4 columns, where the last column contains 1. You can do so with the command:

```
backbone_chain = zeros(N, 4);
```

which creates a matrix with N rows and 4 columns initialized to all zeros. You can set the fourth column to 1 with the command:

```
backbone_chain(:, 4) = 1;
```

Now you need to place the cartesian coordinates from the array to this matrix. You can do so with the for loop below:

```
for i = 1 : N
  for j = 1 : 3
    backbone_chain(i, j) = cartesian_coordinates((i - 1) * 3 + j);
  end;
end;
```

The cartesian coordinates you have just read will serve as a basis for performing rotations, according to what method you choose to represent and manipulate the protein. Say the position of atom i needs to be transformed by your transformation matrix. You can do so by multiplying `trans mat` with `backbone chain(i, :)` as in `trans mat * backbone chain(i, :)`. Note that the colon is very convenient as it gives an entire row or an entire column and that `backbone chain(i, :)` gives you the transpose that is necessary for the multiplication to be carried out. In any case, your transformation matrices need to have dimension 4×4 to work with homogeneous coordinates. You can initialize an empty 4×4 matrix by writing

```
transmat = zeros(4, 4);
```

Then you can set the elements of this matrix to what they should be for the particular method you use. You can evaluate all the cosines and sines that you need in Matlab. Matlab offers you built-in operations such as dot product or vector norm. Make sure that you understand these operations before you apply them.

* I thank Prof. Lydia Kavraci for providing this section.

Setup for R

You can see basic vector and matrix operations in the previous homework setup. I couldn't find an easy way to do cross product in R, so I attach an R file with some functions implemented by me, as well as some basic quaternion stuff. Feel free to use it as a basis for your implementation. Don't forget the "apply" and "sweep" options in R that can save you lots of time instead of using loops which are less efficient.