**Lab 10: Designing a Protein**

# The Protein Folding Problem and Alpha Fold 2

A longstanding grand challenge in Biology has been to predict the 3-dimensional structure of a protein from its amino acid sequence. In principle, this could be done using the laws of physics and chemistry. However, progress using such approaches based on “first principles” has been limited. Instead, strategies based on “learning” empirical rules from existing data have been much more successful. Recent advances in machine learning have driven a major step forward in these approaches during the last few years.

In particular, an artificial intelligence program called Alpha Fold 2 has made a large impact by generating highly accurate predictions for many proteins. This program won a competition for computational protein structure prediction by a wide margin, and it is enabling new strategies in research.

So what can you do with this tool?

We will explore this question by using it to guide the design of a protein that does something interesting.

# An interface for predicting protein structures with Alpha Fold 2

While the existence of a program that can predict protein structures is great, it would be much better if it was available in a way that makes it easy for any researcher to use. Fortunately, this is the case!

The following link will take you a website that provides an interface to run Alpha Fold 2 to predict a structure for whatever amino acid sequence you want to try:

<https://colab.research.google.com/github/sokrypton/ColabFold/blob/main/AlphaFold2.ipynb#scrollTo=kOblAo-xetgx>

Open the link and take a look at the page. This page runs Python code on a server to walk through steps of setting up and using Alpha Fold 2. You will notice that just like the Matlab scripts we have worked with during the quarter, this page is organized into “blocks” which can be executed sequentially.

Based on the heading for the first block (“Input protein sequence …”), can you figure out the simplest way to use this page? Where will you find the “Run all” command? (Look in the menu at the top of the page)

According to instructions, we input the protein sequence of interest and then hit “run all”. “Run all” lives in the “Runtime” menu.

You can scroll up and down on the page to look at the blocks. Some of them will be very important for you. Others will need to run, but you can largely ignore them. As a quick overview:

Block 1: Input protein sequence(s), then hit Runtime -> Run all

This is where you will enter the sequence for the protein you want to model. You can also enter a “job name” which can be useful to keep track of results if you run predictions for multiple sequences.

Block 2: Install dependencies

This needs to run, but you can otherwise ignore it. (There are advanced options and settings that could be set here, but the defaults will work just fine.)

Block 3: Run predictions

This will run, and you will see updates with some pictures of predictions in progress before the program finishes. You don’t need to do anything with this block, except look at the pictures. You will see one heatmap figure, like the one I showed in the lecture. This one shows how your sequence compares to sequences in the program’s database. Colors indicate similar sequences, white indicates a lack of similarity. You will also see the low resolution versions of structures.

Block 4: Display 3D Structure

This is where you will see the final result, in a nice form, where you can rotate the structure around by clicking and dragging with your mouse. You can also see the next best predictions by using the dropdown menu “rank\_num” (1 = best prediction, 2 = second best, …). You can also change the coloring scheme if you want. The default will color the structure by prediction confidence.

Block 5: Package and download results

This block will automatically download a zip file of the results for you. The name of the zip file will be determined by the “job name” you specified above. You could then use the files inside the zip to view the structure using another structure viewing software tool at later times, without having to run the prediction again.

Instructions

There are more detailed instructions under this heading at the bottom if you need them.

# Viewing and exploring protein structures

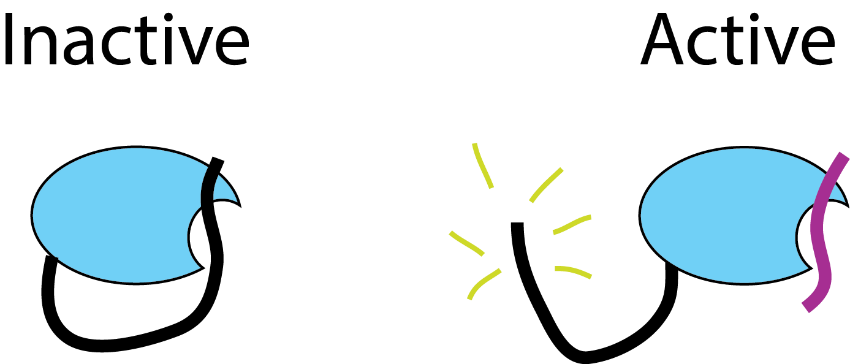
Once we have generated model structures, we can explore them directly on the modeling web page. However, it can also be useful to download structures and view them separately at another time. The Alpha Fold 2 webpage we are using automatically downloads the resulting structures for you. You can then view them later using another handy website:

<https://www.rcsb.org/3d-view>

I have provided a structure for green fluorescent protein (GFP). You can explore this structure using the rcsb website in a little bit, while you are waiting for results from Alpha Fold 2.

# A protein “switch”

As we discussed in class, many proteins are regulated, and they can have different shapes in their “active” and “inactive” forms. Furthermore, an “inactive” protein could be “caged” or “auto-inhibited” by interactions between different parts of the protein.

We will try to build a model protein switch based on this idea. Our protein should be auto-inhibited when it is on its own, but it could then be “opened” or activated by interaction with another protein. In cartoon drawings, its two states should look like this:

You will build this protein by assembling a full amino acid sequence and modeling its structure using Alpha Fold 2. You should use the following building blocks to design your protein:

**SspB**: This is a small protein that binds specifically to a peptide called SsrA. It also binds specifically to a second peptide that we will call Ssj.

EFSSPKRPKLLREYYDWLVDNSFTPYLVVDATYLGVNVPVEYVKDGQIVLNLSASATGNLQLTNDFIQFNARFKGVSRELYIPMGAALAIYARENGDGVMFEPEEIYDELNIG

**SsrA**: This is a peptide that binds specifically to SspB

ANDENYALAA

**Ssj**: This is a second peptide that binds specifically to the same part of SspB

TAFQIAEAANDENYF

**Linker**: To add flexibility when joining two protein pieces into one larger sequence, we often add a “flexible linker”. These often have sequences that are built from repeats of GGS (glycine-glycine-serine).

GGS… ()

# For the following sections:

You are encouraged to work together in small groups. Different members of the group could then model different sequence possibilities in parallel, and you can compare your results to find the best solution.

# Modeling the “inactive” state

First, you should build a sequence for the inactive form. You should use SspB, a linker, and one of the peptides. You can build your protein sequence by pasting these elements together to make one long protein sequence.

You may need to try a few different sequences to get one that “looks good.” In particular, you made need to try a few different linker lengths. A good structure should have the peptide docked against the side of the more compact SspB piece, and the color coding for the peptide part should be yellowish. (A red-colored peptide part would mean that the program is not confident about the position and structure of that part of the protein.)

Once you have a good structure, paste in the sequence of your protein here:

EFSSPKRPKLLREYYDWLVDNSFTPYLVVDATYLGVNVPVEYVKDGQIVLNLSASATGNLQLTNDFIQFNARFKGVSRELYIPMGAALAIYARENGDGVMFEPEEIYDELNIGGGSGGSGGSGGSGGSGGSGGSGGSGGSGGSANDENYALAA (10 GGS linkers)

# Modeling “activation”

You will now model activation of your designed protein. Our protein will be activated by a second peptide which binds to SspB and frees the tail of our protein. You can imagine that this free tail may have another interaction or activity that transmits a signal in the cell.

To model this step, we will use two amino acid sequences as the input for Alpha Fold 2.

As you can see in the brief instructions on the web page, we can provide two amino acid sequences by **putting a colon between them** (such as Protein1Sequence:Protein2Sequence).

You should try using the protein you designed above, but now also use one of the peptides (on its own) as the second protein. Your goal is that the presence of the second protein will “open up” your switch protein.

To get this to work well, you may need to try switching the peptides (you could use the same peptide in your protein and as the activator, or you could use one of the two peptides for each. In the latter case, you might try switching SsrA and Ssj if your initial attempt fails).

Paste in the sequence (including the peptide activator):

EFSSPKRPKLLREYYDWLVDNSFTPYLVVDATYLGVNVPVEYVKDGQIVLNLSASATGNLQLTNDFIQFNARFKGVSRELYIPMGAALAIYARENGDGVMFEPEEIYDELNIGGGSGGSGGSGGSGGSGGSGGSGGSGGSGGSANDENYALAA:ANDENYALAA

and a screenshot of your best “activated” protein:

