MODELLER - I - Introduction

Jean-Yves Sgro

October 26, 2017

Contents

1 Introduction . . . 2
2 Acknowledgments . . . 2
3 Set-up . . . 3
3.1 Terminal . . . 3
3.2 Text editing . . . 3
4 Using MODELLER . . . 4
5 Simple example . . . 4
5.1 INPUT: Target sequence . . . 5
5.2 INPUT: download PDB structure . . . 5
5.3 INPUT: Align sequences . . . 6
5.4 Model building . . . 8
5.5 Run model building script . . . 9
6 Compare model and template graphically . . . 9
6.1 PyMOL . . . 9
6.2 Chimera . . . 10
7 Comparing the model(s) with solved strcutures. . . . 11
8 MODELLER tutorials online . . . 13
8.1 Official web site . . . 13
8.2 Other courses . . . 13
REFERENCES . . . 14

```
## Warning: package 'knitr' was built under R version 3.5.2
```

1 Introduction

From the MODELLER web site[1] :

[1] https://salilab.org/modeller/

`MODELLER` is used for homology or comparative modeling of protein three-dimensional structures (Webb and Sali 2016, Marti-Renom et al. (2000))

The user provides an alignment of a sequence to be modeled with known related structures and **`MODELLER` automatically calculates a model containing all non-hydrogen atoms**.

`MODELLER` implements comparative protein structure modeling by satisfaction of **spatial restraints** (Sali and Blundell 1993, Fiser, Do, and Sali (2000)), and can perform many additional tasks, including *de novo* modeling of loops in protein structures, optimization of various models of protein structure [. . .]

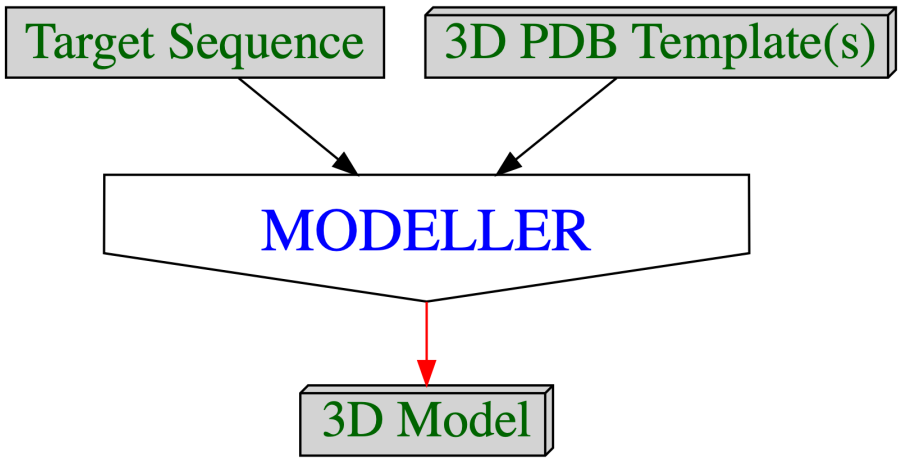

Figure 1: MODELLER process flow

Figure 1.

Modeller is `9.18` is intalled on all the iMacs. However, each user should register with the web site to obtain the install keyword at https://salilab.org/modeller/registration.html

2 Acknowledgments

Part of this tutorial is from "Comparative Protein Structure Prediction MODELLER tutorial" by Marc A. Marti-Renom (PDF)[2]

[2] http://sgt.cnag.cat/www/presentations/files/slides/20081104_MODELLER_Tutorial.pdf

3 Set-up

We will use `MODELLER` on a Macintosh system but it would work exactly the same on other platforms.

`MODELLER` is made of a collection of python scripts, that the user just has to modify to reflect the name of the target sequence(s) and the template structure(s).

It is always good practise to create a directory for a specific project. Let's create a directory on the desktop called `MOD1` where we will save the necessary files.

TASK

Create a folder/directory on your desktop called `MOD1` or any name you wish.

3.1 Terminal

Then `MODELLER` is invoked on the line command with the name of the current version. The current release is `9.18` and is invoked on the line command as **`mod9.18`** followed by the name of the script to run.

TASK

Open a text `Terminal`.

It is necessary to open a text `Terminal` to run `MODELLER`. On Mac `Terminal` is found as `/Applications/Utilities/Terminal` but can easily be launched by typing `Terminal` within the "Spotlight Search" on the top-right corner of the Mac screen (magnifying glass icon.)

(On a Windows computer you would need to open a command line by searching for the `cmd` program with Cortna or the Start button.)

Next it is necessary to change where the `Terminal` is "looking" with the "change directory" `cd` command:

```
cd Desktop
cd MOD1
```

You can check which directory `Terminal` is looking into with the command:

```
pwd
```

In the next section we will add files and scripts to this folder.

3.2 Text editing

Script and/or plain text files can be edited on a Macintosh with the built-in text editor **`TextEdit`**. However, it is necessary to verify that the format is plain text by engaging the menu **`Format`** > **`Make Plain Text`** if the program opens in `Rich Text` format as it is often the default behavior.

Within `Terminal` the full screen word processor **`nano`** could also be used (and is also available on Linux systems.)

Windows users can use `Notepad` or `Wordpad` to easily create plain text files.

To create the necessary text files simply Copy/Paste the information from this page into a text document on your computer using one of the text editors mentioned above.

4 Using MODELLER

To run `MODELLER` we need input data: sequence(s) and 3D template(s) in the proper format as well as python scripts. The later are found on the `MODELLER` web site as example files to be modified.

The output will consist of 1 or more (if requested in the script) 3D PDB format models, an alignment of sequence(s), a log file and other ancilary output.

INPUT:

- sequence(s) target(s): FASTA/PIR format
- structure(s) template(s): PDB format
- Python command file(s): plain text format

OUTPUT:

- Target-Template Alignment
- Model(s) in PDB format
- Other data

5 Simple example

This simple example assumes that some prior study work has been done on the sequence to be modeled to find a suitable 3D template (*e.g.* with `BLAST`.)

The purpose of the exercise is to **create a 3D model from the sequence** of the "brain lipid-binding protein" (blbp) of a mouse sequence based on one existing 3D structure with a different sequence that has been solved and published on the Protein Data Bank (PDB) (Berman et al. 2000).

The sequence in FASTA format looks like this, and has accession code **`NP_067247.1`**.

```
>NP_067247.1 fatty acid-binding protein, brain [Mus musculus]
MVDAFCATWKLTDSQNFDEYMKALGVGFATRQVGNVTKPTVIISQEGGKVVIRTQCTFKNTEINFQLGEE
FEETSIDDRNCKSVVRLDGDKLIHVQKWDGKETNCTREIKDGKMVVTLTFGDIVAVRCYEKA
```

Prior analysis (*e.g.* `BLAST`) reveals that the sequence of the "brain lipid-binding protein" is closely related of that of "human muscle fatty acid binding protein" that has been solved by X-ray crystallography with accession code **`1HMS`** 1hms.pdb (Young et al. 1994).

The sequence of that protein in FASTA format looks like this:

```
>1HMS:A|PDBID|CHAIN|SEQUENCE
VDAFLGTWKLVDSKNFDDYMKSLGVGFATRQVASMTKPTTIIEKNGDILTLKTHSTFKNTEISFKLGVEFDETTADDRKV
KSIVTLDGGKLVHLQKWDGQETTLVRELIDGKLILTLTHGTAVCTRTYEKEA
```

A simple two-sequence BLAST alignment reveals that the protein sequences are 62% identical and 78% similar with no sequence gaps (see below.)

Therefore these are a perfect subject for homology modeling.

Score	Expect	Method	Identities	Positives	Gaps
177 bits(450)	8e-64	Compositional matrix adjust.	81/130(62%)	102/130(78%)	0/130(0%)

```
Query  1    VDAFLGTWKLVDSKNFDDYMKSLGVGFATRQVASMTKPTTIIEKNGDILTLKTHSTFKNT  60
            VDAF  TWKL DS+NFD+YMK+LGVGFATRQV ++TKPT II + G  + ++T  TFKNT
Sbjct  1    VDAFCATWKLTDSQNFDEYMKALGVGFATRQVGNVTKPTVIISQEGGKVVIRTQCTFKNT  60

Query  61   EISFKLGVEFDETTADDRKVKSIVTLDGGKLVHLQKWDGQETTLVRELIDGKLILTLTHG  120
            EI+F+LG EF+ET+ DDR  KS+V LDG KL+H+QKWDG+ET   RE+ DGK+++TLT G
Sbjct  61   EINFQLGEEFEETSIDDRNCKSVVRLDGDKLIHVQKWDGKETNCTREIKDGKMVVTLTFG  120

Query  121  TAVCTRTYEK  130
               V  R YEK
Sbjct  121  DIVAVRCYEK  130
```

5.1 INPUT: Target sequence

TASK

Create a text file called `blbp.seq` containing the sequence sequence in the `MOD1` directory.

You can copy/paste the sequence below. The format starts with `>P1` which is an original annotation form from the early `PIR` protein database .

The `:` colon separators are part of the `MODELLER` format and will make more sense later when you see the PDB sequence transformed in this format automatically below. For now simply copy/paste te following sequence into a plain text file

Example Target: Brain lipid-binding protein (BLBP).

BLBP sequence in PIR (`MODELLER`) format:

```
>P1;blbp
sequence:blbp:::::::
VDAFCATWKLTDSQNFDEYMKALGVGFATRQVGNVTKPTVIISQEGGKVVIRTQCTFKNTEINFQLGEEFEETSIDDRNCKSVV
RLDGDKLIHVQKWDGKETNCTREIKDGKMVVTLTFGDIVAVRCYEKA*
```

5.2 INPUT: download PDB structure

The input structure has accession code **`1HMS`**.

The downloaded file will appear in your `Downloads` directory as 1HMS.pdb.

TASK

Download and then move downloaded file `1HMS.pdb` to the `MOD1` directory.

5.3 INPUT: Align sequences

The target sequence and 3D structure sequence need to be aligned and saved in a file with the proper format.

To accomplish this we need to edit a python script listing the name of the files containing the sequences. The sequence will be extracted from the PDB file itself by `MODELLER` from the script instructions.

TASK

Create a text file called `align.py` with the following content and save it in folder `MOD1`:

```
# Example for: alignment.align()
# This will read two sequences, align them, and write the alignment
# to a file:

log.verbose()
env = environ()
aln = alignment(env)
mdl = model(env, file='1hms')
aln.append_model(mdl, align_codes='1hms')
aln.append(file='blbp.seq', align_codes=('blbp'))
# The as1.sim.mat similarity matrix is used by default:
aln.align(gap_penalties_1d=(-600, -400))
aln.write(file='blbp-1hms.ali', alignment_format='PIR')
aln.write(file='blbp-1hms.pap', alignment_format='PAP')
```

> *Note*: Since these are python functions, they need parentheses `()` even if there is nothing inside them. The meaning of the commands can be found under `MODELLER` online manual https://salilab.org/modeller/manual/ and described succintly below.

Explanations for the commands contained within this script:

- `log.verbose()` : display all log output
- `env = environ()` : create a short name for `environ()`
- `environ()` : contains most information about the MODELLER environment, such as the energy function and parameter and topology libraries [...].
- `aln = alignment(env)` : This creates a new `alignment` object; by default, this contains no sequences. `aln` is the short name for this object.
- `mdl = model(env, file='1hms')` : create a new 3D model. Here we pass on the information about the PDB file and atom information will be read. `mdl` is the short name for this object.
- `aln.append_model(mdl, align_codes='1hms')` : append the sequence of `1hms` to the alignment. In more complex analyzes there could be multiple PDB codes passed on.
- `aln.append(file='blbp.seq', align_codes=('blbp'))` : append the target sequence to the alignment.
- `# The as1.sim.mat similarity matrix is used by default:` This is a comment line
- `aln.align(gap_penalties_1d=(-600, -400))` the command `aln.align` create the alignment based on the indicated gap penalties.
- `aln.write(file='blbp-1hms.ali', alignment_format='PIR')` the alignment is written in `PIR` format.

- `aln.write(file='blbp-1hms.pap', alignment_format='PAP')` the alignment is written in `PAP` format.

It is worth noting the following point:

- the PDB codes are within single quotes, for example `'1hms'`
- If there are multiple arguments passed to a function, there is a space after the comma `,` for example before the word `alignment_format=` in the lines above.

5.3.1 Run script to create alignment files

TASK

Run alignment script `align.py` within `MOD1`.

Verify that you are within the `MOD1` directory:

```
pwd
```

The answer should be something like:

```
/Users/yourname/Desktop/MOD1
```

Now run the alignment script by typing:

```
mod9.18 align.py
```

This will create the files: `blbp-1hms.ali`, `blbp-1hms.pap`, and `align.log`.

To see the content of the alignment files we can use the simple `cat` command on the `Terminal` (or use the graphical interface with TextEdit for example.)

Note the use of the `:` colon separator in the PDB sequence file.

```
cat blbp-1hms.ali
```

```
>P1;1hms
structureX:1hms:   1 :A:+131 :A:MOL_ID  1; MOLECULE  MUSCLE FATTY ACID BINDING PROTEIN; CHAIN  A; ENGINEERED
VDAFLGTWKLVDSKNFDDYMKSLGVGFATRQVASMTKPTTIIEKNGDILTLKTHSTFKNTEISFKLGVEFDETTA
DDRKVKSIVTLDGGKLVHLQKWDGQETTLVRELIDGKLILTLTHGTAVCTRTYEKE*

>P1;blbp
sequence:blbp:     : :     : :::-1.00:-1.00
VDAFCATWKLTDSQNFDEYMKALGVGFATRQVGNVTKPTVIISQEGGKVVIRTQCTFKNTEINFQLGEEFEETSI
DDRNCKSVVRLDGDKLIHVQKWDGKETNCTREIKDGKMVVTLTFGDIVAVRCYEKA*
```

This alignment extracted sequence information from the PDB file for `1HMS` including header information about the content that is placed within the header of `structureX:1hms`.

The `.ali` formatted alignment file is used later by `MODELLER` to create the 3D model(s).

The `.pap` formatted alignment is easier for human eyes to evaluate the alignment with the marked conserved (identity) regions.

```
cat blbp-1hms.pap
```

```
 _aln.pos         10        20        30        40        50        60
1hms      VDAFLGTWKLVDSKNFDDYMKSLGVGFATRQVASMTKPTTIIEKNGDILTLKTHSTFKNTEISFKLGV
blbp      VDAFCATWKLTDSQNFDEYMKALGVGFATRQVGNVTKPTVIISQEGGKVVIRTQCTFKNTEINFQLGE
 _consrvd ****  **** ** *** *** **********   **** **   *      *  ******* * **

 _aln.p   70        80        90       100       110       120       130
1hms      EFDETTADDRKVKSIVTLDGGKLVHLQKWDGQETTLVRELIDGKLILTLTHGTAVCTRTYEKE
blbp      EFEETSIDDRNCKSVVRLDGDKLIHVQKWDGKETNCTREIKDGKMVVTLTFGDIVAVRCYEKA
 _consrvd ** **  ***  ** * *** ** * ***** **   **  ***   *** *  *  * ***
```

5.4 Model building

We now have the necessary "ingredients" to create the 3D model:

- aligned sequences
- 3D original template

We now need to create/edit the `MODELLER` python script that will list these ingredients and call the `MODELLER` functions to build the model.

TASK

Create a text file called `model.py` with the following content and save it in folder `MOD1`. Note that the comments noted with `#` do not need to be re-typed if not creating the file with a copy/paste method. The blank lines are only for text clarity and can also be omitted if desired.

To create the file you can use `TextEdit` or `nano` for example.

```
# Homology modelling by the automodel class
from modeller.automodel import *  # Load the automodel class
log.verbose()                     # request verbose output
env = environ()                   # create a new MODELLER environment

a = automodel(env,
   alnfile = 'blbp-1hms.ali',     # alignment filename
   knowns = '1hms',               # codes of the templates
   sequence = 'blbp')             # code of the target

a.starting_model= 1               # index of the first model
a.ending_model = 1                # index of the last model
                                  # (determines how many models to calculate)

a.make()                          # do the actual homology modelling
```

Remarks: The `automodel` function is renamed `a` and the "dot notation" is used to call on sub function appended to `a` as it is the usual writing mode in python.

In this simple file we create only one model, but to obtain *e.g.* 5 models the `a.ending_model` argument would be set to `5`.

5.5 Run model building script

TASK

Run `model.py` within `MOD1` in the same manner as we ran the `align.py` script:

```
mod9.18 model.py
```

This will create the following files:

```
blbp.B99990001.pdb
blbp.D00000001
blbp.V99990001
model.log
blbp.ini
blbp.rsr
blbp.sch
```

The final 3D model is called `blbp.B99990001.pdb` and that is the "end product" that was desired.

In real life, multiple models would be calculated (*e.g.* `5`) and various evaluation methods could be applied to decide which are "best."

You can explore the content of the remaining file (all text files) with the `less -S` command that will display the file content to the screen without wrapping long lines.

6 Compare model and template graphically

Now that we have a model we can compare the structure onbtained with the original template.

For this you can use `Chimera` or `PyMOL` or any other molecular graphics software that can read `PDB` files.

6.1 PyMOL

To open and compare files in PyMOL open the PyMOL program first.

- At the line command type: `fetch 1hms` to load the original template file.
- Using the menu cascade **`File`** > **`Open...`** navigate to the `MOD1` directory to open file `blbp.B99990001.pdb`.
- Use left mouse button to rotate structure.

Note: the 2 structures will not be superimposed at first and it will be necessary to align them in 3D.

- Align the structures: on the Names panel at right, click on **`A`** (action) button next to the line that reads **blbp.B99990001.pdb 1** for the model. Following further down on this pull-down menu follow the menu cascade: **`A`** > **`align`** > **`to molecule (*/CA)`** > **`1hms`**
- To hide or show either structure simply click **once** on the name of the structure on the list at the right hand side Names panel.

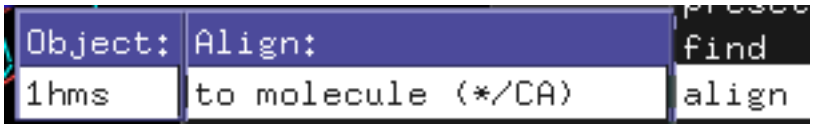

Figure 2: *"Align structures menu."*

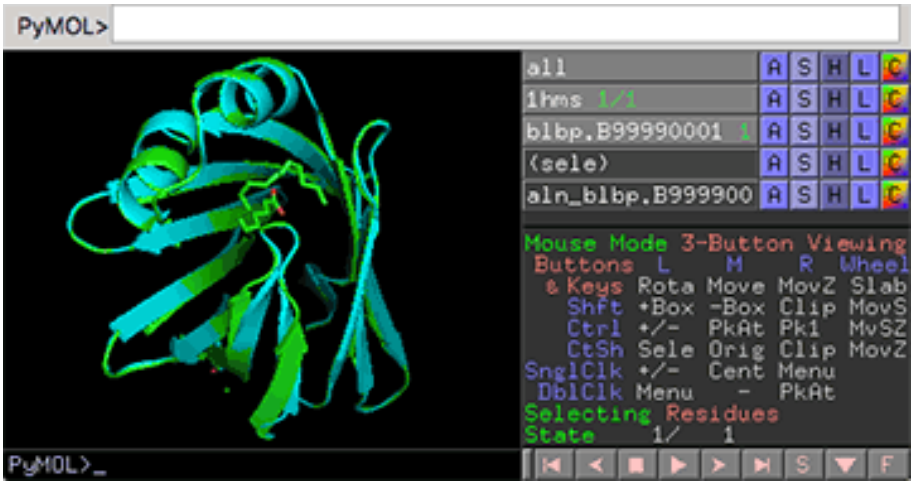

Figure 3: *"Open and align structures in PyMOL."*

- In order to highlight the bound lipid use the following menu casade next to the **all** line on the right hand side: **`all`** > **`S`** > **`organic`** > **`sticks`**
- To hide the red dot water molecules: **`all`** > **`H`** > **`waters`**

  Note: only the protein is modeled, the ligands are not modeled by `MODELLER`. These are typically written as `HETATM` within the PDB file.

6.2 Chimera

If you prefer using Chimera:

- Open Chimera
- Open template structure: **`File`** > **`Fetch by ID...`** and enter `1HMS` in the `Fetch Structure by ID` in the text space next to the `PDB` button. This will open the structure in "first view" mode as a cartoon ribbon diagram.
- Open the model: **`File` > `Open...` and navigate to the `MOD1` directory to open file `blbp.B99990001.pdb`. The default view will also be as a cartoon ribbon.

Note: the 2 structures will not be superimposed at first and it will be necessary to align them in 3D.

- **`Tools`** > **`Sequence Comparison`** > **`MatchMaker`** will open the `MatchMaker` window. Keep everything the the current default and click `1HMS (#0)` for the "Reference structure" and `blbp.B99990001.pdb (#1)` for the "Structure(s) to match"
- Click `Apply` and the 2 structures will be aligned.
- Use left mouse button to rotate structure.

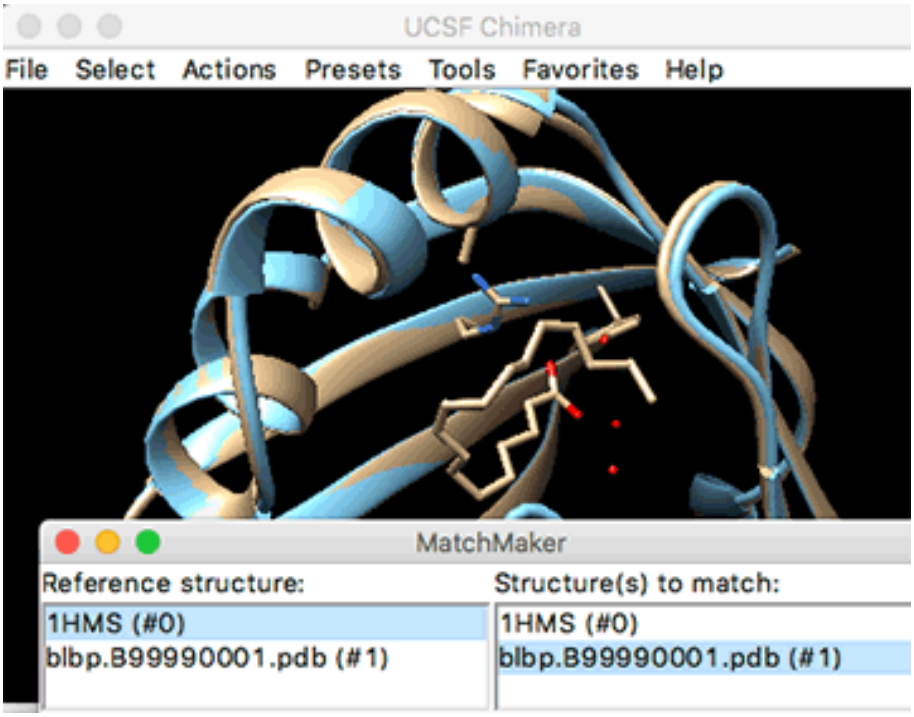

Figure 4: *"Open and align structures in Chimera."*

7 Comparing the model(s) with solved strcutures.

It happens that since this exercise was written many actual structures were solved.

A `BLAST` restricted to the Protein Data Bank will give some PDB codes of solved structures. For example:

Description	Max score	Total score	Query cover	E value	Ident	Accession
Chain A, Enantiomer-specific Binding Of The Potent Antinoci-ceptive Agent Sbfi-26 To Anan-damide Trans-porters Fabp7	240	240	100%	5e-84	87%	`5URA`_A

Description	Max score	Total score	Query cover	E value	Ident	Accession
Chain A, Crystal Structure Of Human Brain Fatty Acid Binding Protein	238	238	99%	4e-83	87%	1FDQ_A
Chain A, Human Fabp3 In Complex With 6-chloro-2-methyl-4-phenyl-quinoline-3-Carboxylic Acid	180	180	99%	3e-60	63%	5HZ9_A
Chain A, Serial Femtosecond X-ray Structure Of Human Fatty Acid-binding Protein Type-3 (fabp3) In Complex With Stearic Acid (c18:0) Determined Using X-ray Free-electron Laser At Sacla	180	180	99%	3e-60	63%	3WXQ_A

The table is much longer!

Here is the alignment for the first in the table: `5URA` chain `A`.

```
Range 1: 4 to 135
Alignment statistics for match #1
Score   Expect  Method  Identities  Positives   Gaps
240 bits(613)   5e-84   Compositional matrix adjust.    115/132(87%)    124/132(93%)    0/132(0%)
Query  1    MVDAFCATWKLTDSQNFDEYMKALGVGFATRQVGNVTKPTVIISQEGGKVVIRTQCTFKN  60
```

```
             MV+AFCATWKLT+SQNFDEYMKALGVGFATRQVGNVTKPTVIISQEG KVVIRT  TFKN
Sbjct  4     MVEAFCATWKLTNSQNFDEYMKALGVGFATRQVGNVTKPTVIISQEGDKVVIRTLSTFKN  63

Query  61    TEINFQLGEEFEETSIDDRNCKSVVRLDGDKLIHVQKWDGKETNCTREIKDGKMVVTLTF  120
             TEI+FQLGEEF+ET+ DDRNCKSVV LDGDKL+H+QKWDGKETN  REIKDGKMV+TLTF
Sbjct  64    TEISFQLGEEFDETTADDRNCKSVVSLDGDKLVHIQKWDGKETNFVREIKDGKMVMTLTF  123

Query  121   GDIVAVRCYEKA  132
             GD+VAVR YEKA
Sbjct  124   GDVVAVRHYEKA  135
```

OPTIONAL EXERCISE:

Load some of the solved structures and compare them to the model(s.)

8 MODELLER tutorials online

8.1 Official web site

The `MODELLER` web site offers tutorials with different levels of difficulty https://salilab.org/modeller/tutorial/ :

1. Basic Modeling. *Model a sequence with high identity to a template.* This exercise introduces the use of MODELLER in a simple case where the template selection and target-template alignments are not a problem.
2. Advanced Modeling. *Model a sequence based on multiple templates and bound to a ligand.* This exercise introduces the use of multiple templates, ligands and loop refinement in the process of model building with MODELLER.
3. Iterative Modeling. *Increase the accuracy of the modeling exercise by iterating the 4 step process.* This exercise introduces the concept of MOULDING to improve the accuracy of comparative models.
4. Difficult Modeling. Model a sequence based on a low identity to a template. This exercise uses resources external to MODELLER in order to select a template for a difficult case of protein structure prediction.
5. Modeling with cryo-EM. Model a sequence using both template and cryo-EM data. This exercise assesses the quality of generated models and loops by rigid fitting into cryo-EM maps, and improves them with flexible EM fitting.

8.2 Other courses

Virtual Proteomics Laboratory - Experiment 10: Homology Modelling - http://iitb.vlab.co.in/?sub=41&brch=118&sim=657&cnt=2

REFERENCES

Berman, H. M., J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. 2000. "The Protein Data Bank." *Nucleic Acids Res.* 28 (1): 235–42.

Fiser, A., R. K. Do, and A. Sali. 2000. "Modeling of loops in protein structures." *Protein Sci.* 9 (9): 1753–73.

Marti-Renom, M. A., A. C. Stuart, A. Fiser, R. Sanchez, F. Melo, and A. Sali. 2000. "Comparative protein structure modeling of genes and genomes." *Annu Rev Biophys Biomol Struct* 29: 291–325.

Sali, A., and T. L. Blundell. 1993. "Comparative protein modelling by satisfaction of spatial restraints." *J. Mol. Biol.* 234 (3): 779–815.

Webb, B., and A. Sali. 2016. "Comparative Protein Structure Modeling Using MODELLER." *Curr Protoc Bioinformatics* 54 (June): 1–5.

Young, A. C., G. Scapin, A. Kromminga, S. B. Patel, J. H. Veerkamp, and J. C. Sacchettini. 1994. "Structural studies on human muscle fatty acid binding protein at 1.4 A resolution: binding interactions with three C18 fatty acids." *Structure* 2 (6): 523–34.