

Tutorial para predição de estrutura de proteínas

V Escola de Modelagem Molecular em Sistemas Biológicos

Priscila V. Z. Capriles Goliatt – capriles@lncc.br

Fábio Lima Custódio- flc@lncc.br

Comandos Básicos do Linux:

cd diretório/	Entra na pasta diretório/
ls diretório/	Lista as subpastas e arquivos existentes em diretório/
cd ..	Retorna para a pasta anterior
mkdir diretório/	Cria a pasta diretório/
rm arquivo.txt	Remove arquivo.txt (.txt ou outro formato de arquivo)
rmdir diretório/	Remove a pasta diretório/
cp diretório1/arquivo.txt diretório2/	Copia arquivo.txt no diretório1/ para diretório2/

* Neste curso os arquivos já estão salvos pastas mencionadas no Tutorial.

Parte 3: Predição via método híbrido: modelagem comparativa + *ab initio*

Exemplo 1: Modelagem com baixa taxa de identidade

Passo 1: Construindo o modelo via MHOLline

- 1.1 Acessar o site do MHOLline: <http://www.mholline.lncc.br>
- 1.2 Clicar em “Submit Your Job Now” e em “Submit Now”.
- 1.3 Selecionar todos os programas
- 1.4 Fazer o upload do arquivo /home/vemmsb/modelagem/sodef/sequences/SODEF17.fasta
Atenção: não submeter o job!!!!
- 1.5 Acesse o resultado através do link:

<http://www.mholline.lncc.br/result.php?id=5cb7ed4858e19b731cdd6b71f9ce0bba>

Atenção: Os resultados do MHOLline já estão salvos no diretório:
‘/home/vemmsb/modelagem/sodef/jobsodef17’

Passo 2: Analisar o modelo construído via MHOLline

- 2.1 Visualizar o resultado do BLAST: abrir com um navegador *web* (e.g. firefox, iceweasel, konqueror, chrome) o arquivo `/home/vemmsb/modelagem/sodef/jobsodef17/jobsodef17aa.html`
- 2.2 Visualizar o resultado do HMMTOP: abrir com um editor de texto (e.g. gedit, ooffice) o arquivo `/home/vemmsb/modelagem/sodef/jobsodef17/hmmtop.out`
- 2.3 Visualizar o resultado do BATS: abrir com um editor de texto (e.g. gedit, ooffice) o arquivo `/home/vemmsb/modelagem/sodef/jobsodef17/bats_results.txt`
- 2.4 Visualizar o resumo do resultado do BATS: abrir com um editor de texto (e.g. gedit, ooffice) o arquivo `/home/vemmsb/modelagem/sodef/jobsodef17/bats_log.txt`
- 2.5 Visualizar as estruturas classificadas em G1 no site do PDB (acesse o site: <http://www.pdb.org/pdb/home/home.do>)
- 2.6 Fazer o download das estruturas no formato FASTA e PDB.

Atenção: Os arquivos FASTA e PDB já estão salvos no diretório:
`‘/home/vemmsb/modelagem/sodef/sequences’`

Passo 3: Construindo o modelo via Modeller

Modelagem baseada em mais de um molde (template):

Preparando os arquivos de entrada:

- 3.1 Realizar o alinhamento múltiplo entre os possíveis templates obtidos com o MHOLline e a sequência alvo através do site: <http://www.ebi.ac.uk/Tools/es/cgi-bin/clustalw2>
- 3.2 Na opção “OUTPUT FORMAT”, selecionar o formato “PIR” (formato usado pelo MODELLER) e em “OUTPUT ORDER”, selecionar input
- 3.3 Abaixo, colar as seqüências FASTA dos templates e da sequência alvo contidas no arquivo: `/home/vemmsb/modelagem/sodef/sequences/multi.fasta`
- 3.4 Clicar em “RUN”, visualizar o resultado e salvar o arquivo de resultado no formato ALN.

Atenção: Os resultados do ClustalW já estão salvos no diretório:
`‘/home/vemmsb/modelagem/sodef/sequences’`

- 3.5 Abra o com um editor de texto e compare os arquivos `multi.aln` (gerado pelo ClustalW) e `SODEF17.ali` (já preparado no formato de entrada do Modeller).

Gerando o modelo da proteína alvo SODEF17 com o Modeller:

- 3.6 Digite na linha de comando do terminal: `mod9v8 model-multi.py`

Atenção: irá aparecer a mensagem: `'import site' failed; use -v for traceback`. Não se preocupe!

- 3.7 O *script* model-single.py lê o arquivo de alinhamento múltiplo SODEF17.ali, gera três modelos otimizados para a proteína alvo através da função automodel(), lista dentro do arquivo model-single.log a pontuação dos modelos gerados de acordo com os métodos DOPE (*Discrete Optimized Protein Energy*) e GA34, e retorna o nome da melhor estrutura gerada de acordo com a pontuação do DOPE.

Atenção: Os modelos gerados (SODEF17.B99990001.pdb, SODEF17.B99990002.pdb e SODEF17.B99990003.pdb) já encontram-se no diretório ./sequences/multipt/

Avaliando os modelos 3D gerados:

- 3.8 Abra com um editor de texto o arquivo /home/vemmsb/modelagem/sodef/model-multi.log
- 3.9 Vá até o final do arquivo e observe os resultados do DOPE.
- 3.10 OPS!!!!!!!!!!!!!!!!!!!! Não tem resultado do DOPE :-P
- 3.11 Será que eu fiz besteira?!?!?!?!?!?

Validando os modelos gerados:

- 3.12 Avaliar com o VMD os modelos gerados e comparar com a estrutura dos templates
- 3.13 Digite na linha de comando do terminal: procheck ../alvo.B99990001.pdb 1.5

Observação: o número 1.5 refere-se à resolução para a análise do modelo.

- 3.14 Abra com o “gv” os gráficos de Ramachandran gerados para a análise dos resultados

Atenção: Os resultados já encontram-se nos diretórios:
./sequences/multipt/procheckresult

OPS!!! AS REGIÕES C- E N-TERMINAL NÃO ESTÃO MODELADAS. O QUE EU FAÇO AGORA?!?!

Procurando por região de peptídeo sinal:

- 1.1 Acessar o site do SignalP: <http://www.cbs.dtu.dk/services/SignalP-3.0>
- 1.2 Colar a sequência FASTA da SODEF17 e definir os parâmetros de busca.
- 1.3 Analisar os resultados no link:

<http://www.cbs.dtu.dk/cgi-bin/webface?jobid=signalp,4C7532BE02FC1C4C&opt=none>

Aula 4: Completando modelos com o Rosetta

Introdução

Ao utilizar a modelagem comparativa é possível que regiões da sua estrutura fiquem sem estrutura devido à falta de estruturas de referencia. Nessa parte do tutorial iremos utilizar o Rosetta para modelar tais regiões partindo da estrutura gerada por modelagem comparativa.

Para modelar regiões restritas de um alvo existem duas opções de protocolo:

1. Protocolo “Floppy Tail”
2. Protocolo “Loop Modeling”

A aplicação de cada um dos dois protocolos depende do alvo sendo estudado, bem como da combinação correta de parâmetros. Neste tutorial iremos exemplificar a aplicação dos dois protocolos para modelar partes do modelo incompleto gerado com o modeller.

Protocolo 1: “Floppy Tail”

Este é um protocolo experimental do Rosetta e sua melhor aplicação é a modelagem longos **segmentos terminais** da estrutura que não puderam ser observados experimentalmente ou por modelagem comparativa. Esse protocolo apresenta um desempenho especialmente bom para sequências flexíveis.

Passo 1: Geração dos fragmentos

Assim como na predição da estrutura completa necessitamos dos arquivos de fragmentos.

<http://rosetta.bakerlab.org/fragmentsubmit.jsp>

Os arquivos de fragmentos já foram fornecidos para esse tutorial.

Crie um diretório de trabalho:

```
$ mkdir ~/floppy1  
$ cd ~/floppy1
```

Copie os arquivos de fragmentos para o diretório de trabalho:

```
$ cp ~/abinitio/sodef_frgs/* ~/floppy1
```

Abra o arquivo pdb com a estrutura incompleta para identificar as regiões a serem modeladas

```
$ vmd full.pdb
```

Duas regiões ficaram sem estrutura. Note a falta das cadeias laterais completas.

Região 1: resíduos 1 a 8

Região 2: resíduos 58 a 71

Essas são regiões terminais da estruturas (N terminal e C terminal). Isso torna possível a aplicação do protocolo “floppy tail”. Abra o arquivo de predição de estrutura secundária e observe essas regiões.

```
$ cat SODEF.psipred
```

Passo 2: Construção dos modelos

Os diversos protocolos do Rosetta são implementados em executáveis separados.

O nome (com o caminho) do executável executa o protocolo "loop modeling" é:

```
$ ~/rosetta3/rosetta_source/bin/FloppyTail.linuxgccrelease
```

O Rosetta permite que se grave os parâmetros em um arquivo texto para uma maior organização que pode ser passado ao executável. Monte um arquivo texto, chamado "ntail.txt", com os seguintes parâmetros (ou monte diretamente a linha de comando):

```
$ gedit ntail.txt
```

```
-database /home/vemmsb/rosetta3/rosetta_database
-flexible_start_resnum 1
-flexible_stop_resnum 8
-flexible_chain A
-s full.pdb
-short_tail_off 0
-short_tail_fraction 1.0
-shear_on .33333333333333333333
-AnchoredDesign::perturb_temp 0.8
-AnchoredDesign::perturb_cycles 3000
-AnchoredDesign::refine_temp 0.8
-AnchoredDesign::refine_cycles 500
-AnchoredDesign::refine_repack_cycles 30
-in::file::frag3 aaSODE_03_05.200_v1_3
-in::file::frag9 aaSODE_09_05.200_v1_3
-loops::relax shortrelax
-abinitio:fastrelax
-overwrite
-nstruct 1
-run::min_type dfpmin_armijo_nonmonotone
-run:use_time_as_seed
```

Agora faça um arquivo chamado ctail.txt para modelar a porção terminal.

```
$ cp ntail.txt ctail.txt
$ gedit ctail.txt
```

Mude as linhas:

```
-flexible_start_resnum 1
-flexible_stop_resnum 8
```

Para:

```
-flexible_start_resnum 58
-flexible_stop_resnum 71
```

Para executar o rosetta com o arquivo de parâmetros rode então:

Cada arquivo de entrada irá instruir o Rosetta para modelar uma porção terminal diferente. Por isso, devemos entre uma etapa e outra copiar a saída para o arquivo de entrada... Siga os comandos:

```
~/rosetta3/rosetta_source/bin/FloppyTail.linuxgccrelease @nterm.txt  
cp full_0001.pdb full.pdb  
~/rosetta3/rosetta_source/bin/FloppyTail.linuxgccrelease @cterm.txt  
cp full_0001.pdb full.pdb  
~/rosetta3/rosetta_source/bin/FloppyTail.linuxgccrelease @nterm.txt  
cp full_0001.pdb full.pdb  
~/rosetta3/rosetta_source/bin/FloppyTail.linuxgccrelease @cterm.txt
```

Passo 3: Análise dos resultados

Abra no vmd a estrutura original e a nova em moléculas separadas:

```
$ vmd -m full_ori.pdb S_0001.pdb
```

Vamos escolher uma boa visualização para as estruturas:

1. Vá em “Graphics > Representation”.
2. Selecione a molécula 1 (full_ori.pdb).
3. Mude o “Drawing Method” para “New Cartoon”.
4. Mude o “Coloring Method” para “Chain”.
5. Mude o “Material” para “Transparent”.
6. Selecione a molécula 0 (S_0001.pdb).
7. Mude a representação para “New Cartoon” com cores do tipo “Secondary Structure”.

Vamos agora alinhar as duas estruturas pelo centro conhecido:

1. Vá em “Extensions > Analysis > RMSD Calculator”.
2. Mude os resíduos selecionados para “residue 10 to 55”.
3. Clique em alinhar.

Protocolo 2: Loop Modeling

Passo 1: Geração dos fragmentos

Assim como na predição da estrutura completa a modelagem de voltas necessita dos arquivos de fragmentos.

<http://rosetta.bakerlab.org/fragmentsubmit.jsp>

Os arquivos de fragmentos já foram fornecidos para esse tutorial.

Crie um diretório de trabalho:

```
$ mkdir ~/loopm  
$ cd ~/loopm
```

Copie os arquivos de fragmentos para o diretório de trabalho:

```
$ cp ~/abinitio/sodef_frgs/* ~/loopm
```

Abra o arquivo pdb com a estrutura incompleta para identificar as regiões a serem modeladas

```
$ vmd full_ori.pdb
```

Duas regiões ficaram sem estrutura. Note a falta das cadeias laterais completas.

Região 1: resíduos 1 a 8

Região 2: resíduos 58 a 71

Essas são regiões terminais da estruturas (N terminal e C terminal). Isso torna possível a aplicação do protocolo “floppy tail” para complementar/comparar com o resultado do protocolo “loop modeling”.

Abra o arquivo de predição de estrutura secundária e observe essas regiões.

```
$ cat SODEF.psipred
```

Passo 2: Criação do arquivo de “loops”

Para que o Rosetta necessita de um arquivo contendo informações sobre quais regiões devem ser modeladas.

Edite um arquivo de texto chamado “loop_file.txt”.

```
$ gedit loop_file.txt
```

Esse arquivo deve seguir o seguinte formato:

- coluna1 "LOOP": A palavra LOOP
- coluna2 "inteiro": Resíduo inicial
- coluna3 "inteiro": Resíduo final
- coluna4 "inteiro": Resíduo de corte, deixe 0
- coluna5 "real": deixe 0.0
- coluna6 "booleano": Estender ou não o “loop”

Por exemplo:

LOOP	1	8	0	0.0	1
LOOP	58	71	0	0.0	1

Salve o arquivo em sua pasta de trabalho.

Passo 3: Construção dos modelos

Os diversos protocolos do Rosetta são implementados em executáveis separados. O nome (com o caminho) do executável executa o protocolo “loop modeling” é:

```
$ ~/rosetta3/rosetta_source/bin/loopmodel.linuxgccrelease
```

O Rosetta permite que se grave os parâmetros em um arquivo texto para uma maior organização que pode ser passado ao executável. Monte um arquivo texto, chamado “**params.txt**”, com os seguintes parâmetros (ou monte diretamente a linha de comando):

```
-in::file::fullatom
-in::file::psipred_ss2 SODE_.psipred_ss2
-loops::input_pdb full.pdb
-loops::loop_file loop_file.txt
-loops::frag_sizes 9 3
-loops::frag_files aaSODE_09_05.200_v1_3 aaSODE_03_05.200_v1_3
-loops::ccd_closure
-loops::random_loop
-loops::refine refine_ccd
-loops::remodel quick_ccd_moves
-loops::relax fastrelax
-out:pdb
-out:nstruct 1
-mute core.io.database
-database /home/vemmsb/rosetta3/rosetta_database
-run:use_time_as_seed
```

Para executar o rosetta com o arquivo de parâmetros rode então:

```
$ ~/rosetta3/rosetta_source/bin/loopmodel.linuxgccrelease @params.txt
```

Passo 4: Análise dos resultados

Abra no vmd a estrutura original e a nova em moléculas separadas:

```
$ vmd -m full_ori.pdb S_0001.pdb
```

Vamos escolher uma boa visualização para as estruturas:

8. Vá em “Graphics > Representation”.
9. Selecione a molécula 1 (full.pdb).
10. Mude o “Drawing Method” para “New Cartoon”.
11. Mude o “Coloring Method” para “Chain”.
12. Mude o “Material” para “Transparent”.
13. Selecione a molécula 0 (S_0001.pdb).
14. Mude a representação para “New Cartoon” com cores do tipo “Secondary Structure”.

Vamos agora alinhar as duas estruturas pelo centro conhecido:

4. Vá em “Extensions > Analysis > RMSD Calculator”.
5. Mude os resíduos selecionados para “residue 10 to 55”.
6. Clique em alinhar.