

Rosetta Software Suite Protein Design

Amanda Duran
Rosetta Workshop 2014
April 2014

amanda.m.duran@vanderbilt.edu

Overview

Introduction to Protein Design

How it works:

- The Packer

- Interaction Graph

- Resfiles

Rosetta Design Applications:

- Novel Folds

- Protein-Protein interactions

- Protein-Ligand interactions

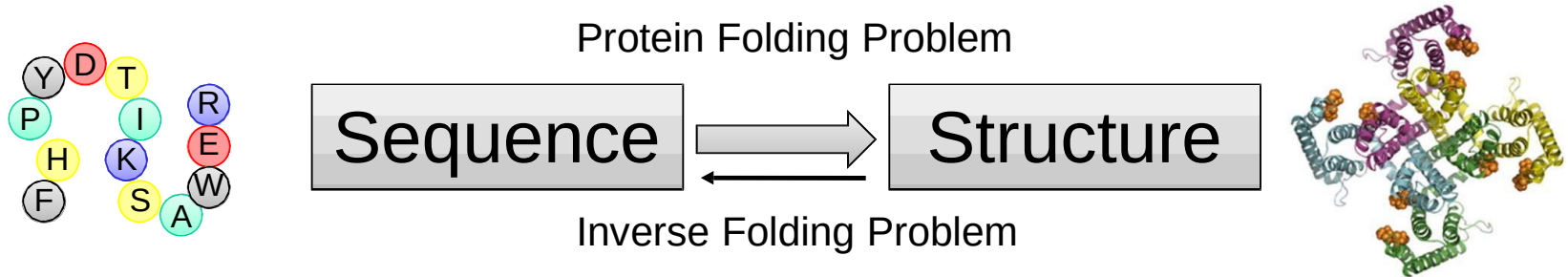
- Therapeutic

Overview of the Tutorial:

- File Formats

- Conclusions

Protein Design is the Inverse Protein Folding Problem



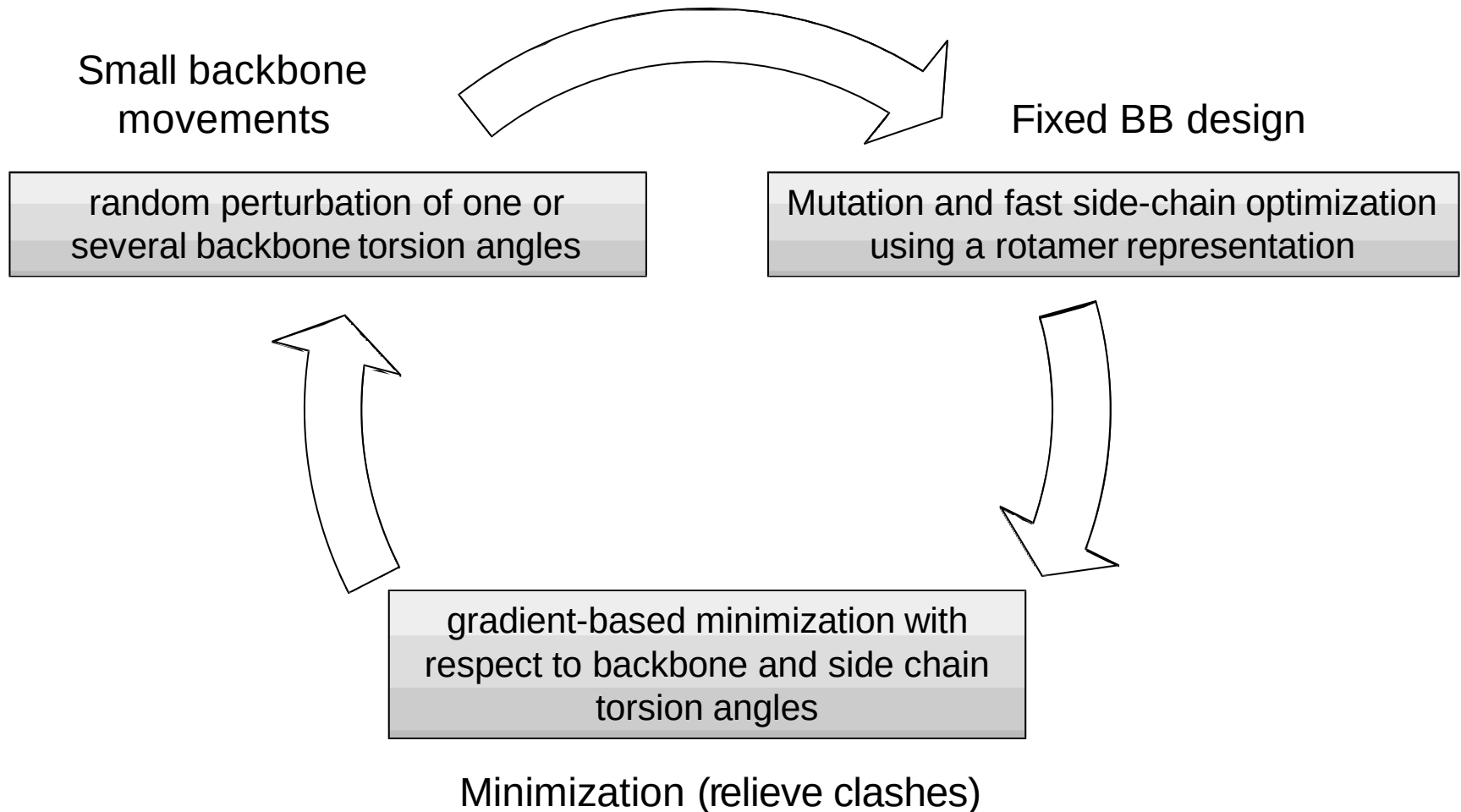
- Given a protein fold – which primary sequence(s) can fold into it?
- Rosetta uses the same sampling algorithms + scoring functions determine the inverse protein folding problem

Rosetta Fixed Backbone Design

Native Sequences are nearly optimal

- For this study, Rosetta Design had no knowledge of native sequences
- 108 proteins were designed
- Able to recover 51% of native sequence in the core
- Lennard-Jones and solvation terms dominate the potential
 - These terms typically indicated native sequences were optimal

Iterative Optimization of Sequence and Conformation in Rosetta

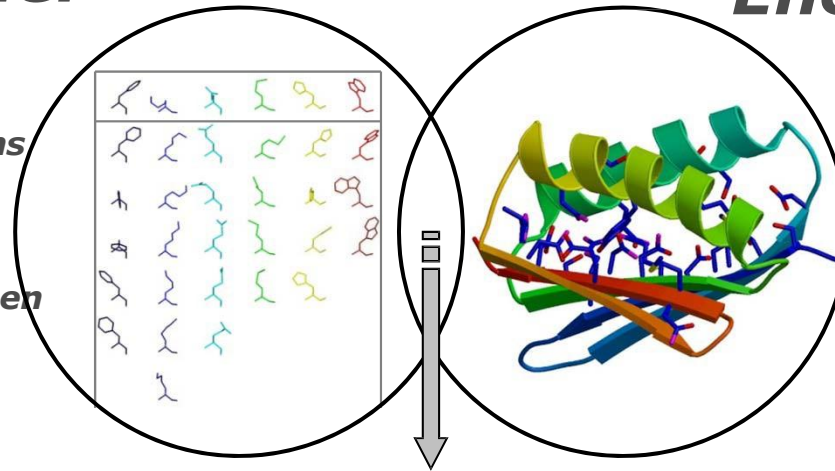


How Rosetta Design works...

Protein Design Uses the Rosetta Energy Function and Local Rotamer Libraries

Local Rotamer Bias

Approximate interactions within sidechain using the distribution of sidechain conformations (rotamers) seen in known protein structures

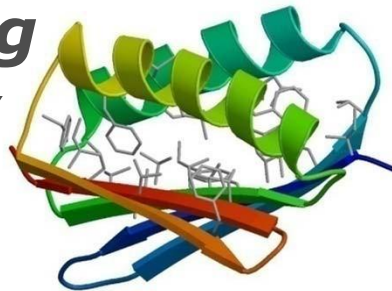


Energy function

Statistically derived potential function

- VDW interaction
- solvation
- hydrogen bonding potential
- pair wise interactions
- rotamer probability

Simulated Annealing Monte Carlo energy minimization



The Packer

In general, the purpose is to optimize rotamers on a fixed backbone.

Native (repack)

All Amino Acids (design)

Specified Amino Acids (guided design)

Steps of the packer:

1. Detect neighbors
2. Build rotamers
3. Calculate energies
4. Simulated annealing

The Packer-What it's Actually Doing

Random rotamer substitution

- set of rotamers to be considered are specified

Evaluate energy

- simulated annealing from high to low temperature

Accept/Reject

- Monte Carlo criterion

Return best energy

The Interaction Graph

The packer is guided by an interaction graph (IG). The IG is evaluated by the same scoring functions we have been talking about...

$$W_{rot}E_{rot} + W_{aa|phi,psi}E_{aa|phi,psi} + W_{rama}E_{rama} + W_{atr}E_{atr} + W_{solv}E_{solv} + W_{pair}E_{pair} + W_{bb_hbond}E_{bb_hbond} + W_{sc_hbond}E_{sc_hbond} + W_{sc_bb_hbond}E_{sc_bb_hbond} + W_{pair}E_{pair} - E_{ref}$$

With one exception:

For a given fixed backbone – They are **precomputed** with the Rosetta Scoring function at each amino acid position

It is recalculated upon backbone movements

The Reality of the Interaction Graph...

It is complicated, and represents the entire protein structure in nodes (one-body) and edges (two-body)

Packing is a combinatorial problem, so the more positions you allow for design to happen on, and the more rotamers at each position, means more time!.

The IG will also rapidly grow as a result, meaning more memory usage.

If you need to reduce memory usage, add the following to your command-line to enable a linear IG:

```
-linmem_ig 10
```

How to Control the Packer...

The Residue File Guides Design

```
ALLAA #allow all 20 amino
ALLAAwc #allow all 20 amino(default)
ALLAAxc #allow all amino acids except cysteine
POLAR #allow only canonical polar amino acids
APOLAR #allow only canonical non-polar amino acids
NOTAA #disallow only the specified amino acids
PIKAA #allow only the specified amino acids
NATAA #allow only the native amino acid (repack)
NATRO #preserve input rotamer
EMPTY #disallow all canonical amino acids
NC <ResidueTypeName> #allow the specific non canonical residue
```

You can also combine commands (see tutorial). Combining commands on a single line means the packer will only allow residues that intersect

```
25 A POLAR NOTAA K
```

When on separate lines (say at first a range, then at specific residues):

```
5-20 A ALLAA
15 A PIKAA Y
```

Residue 15 on chain A will only sample Y, whereas 5-14 and 16-20 will sample all

https://www.rosettacommons.org/manuals/archive/rosetta3.5_user_guide/d1/d97/resfiles.html

Basic Format of Residue Files

```
<Header> #instructions for all positions not specified in body
#The header can also use commands such as EX 1, EX 2, and
USE_INPUT_SC to apply to all positions not specified below
START #keyword
<Body> #instructions for specific chains and identifiers
<PDBNUM> <CHAIN> <COMMANDS>
#Basic format for lines in body
* <CHAIN> <COMMANDS>
#used to specify a command for an entire chain
```

For example, a resfile that does nothing:

```
NATRO #keeps all input rotamers (and hence identity)
START
```

Now it's your turn to predict what the
Packer would do...

NATRO

START

* A NATAA

Another chance to predict what the Packer would do...

NATRO

START

30 A PIKAA FY

20-35 A ALLAAxc

One last time!

NATAA

EX 1

START

25 A ALLAAxC

27 A POLAR

35 A PIKAA KR

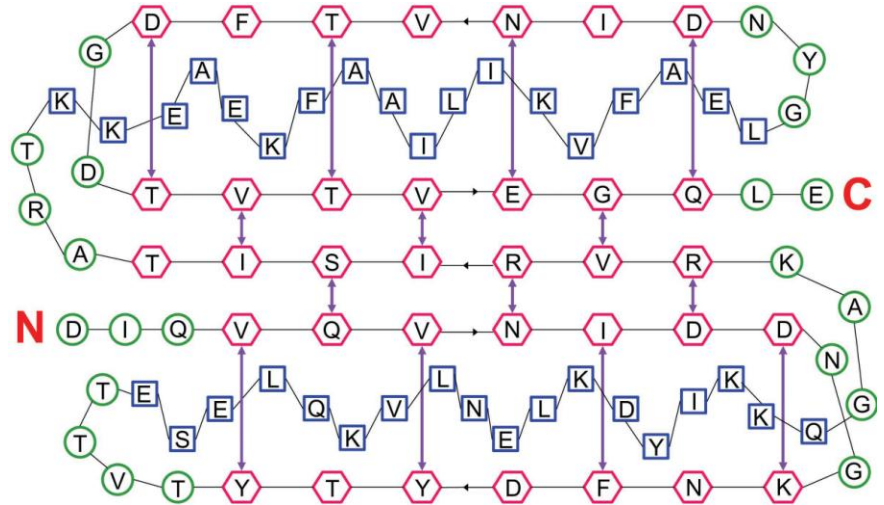
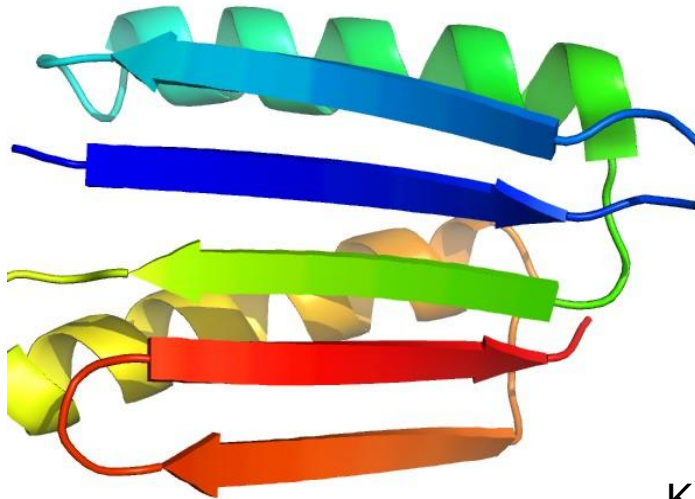
37 A ALLAA

Rosetta Protein Design Applications

De Novo Design of a Novel Fold

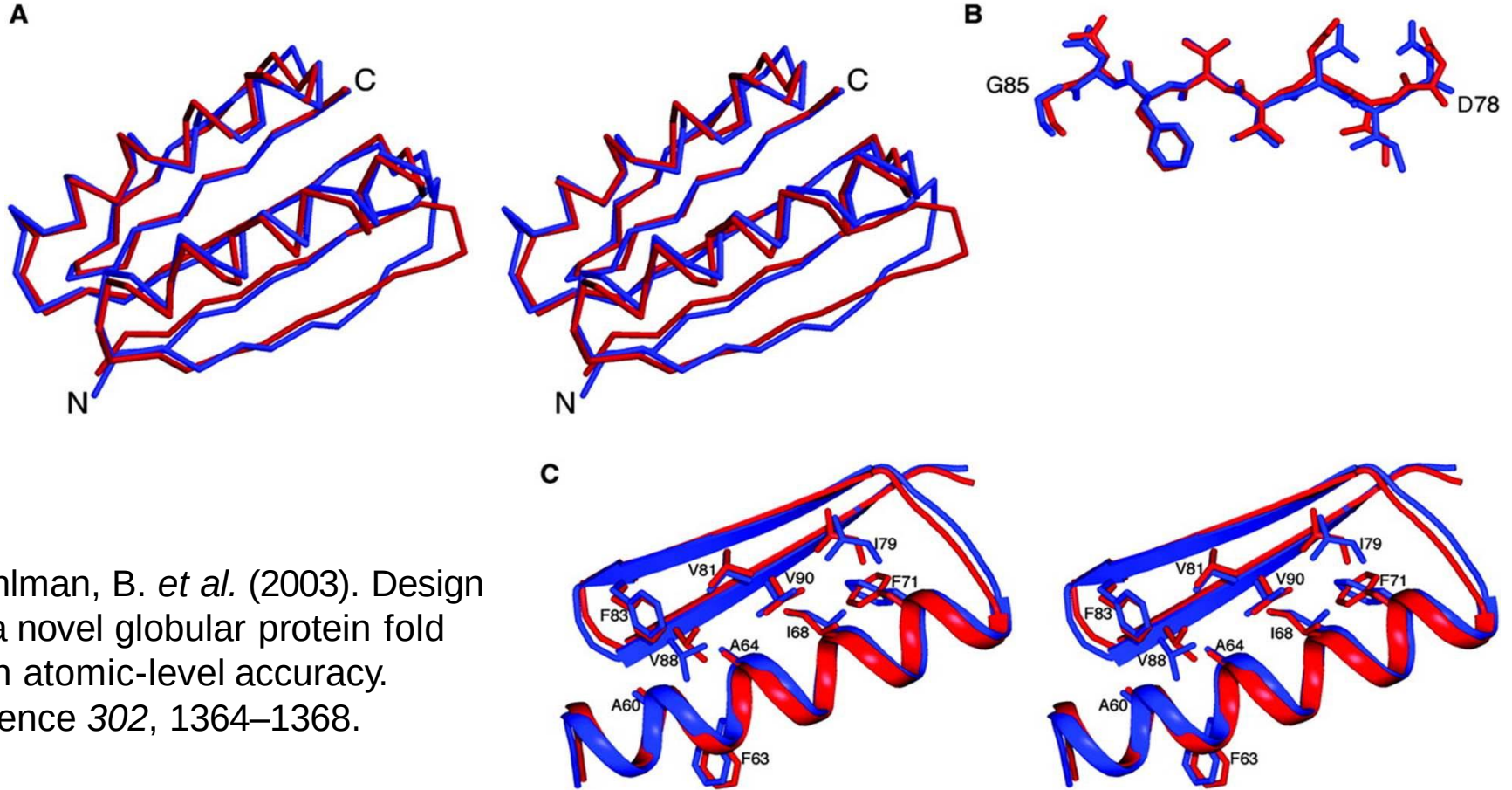
Top7 “back-of-the envelope”
drawn topology not found in the
PDB at time of design

Iterative fixed backbone design +
backbone perturbations



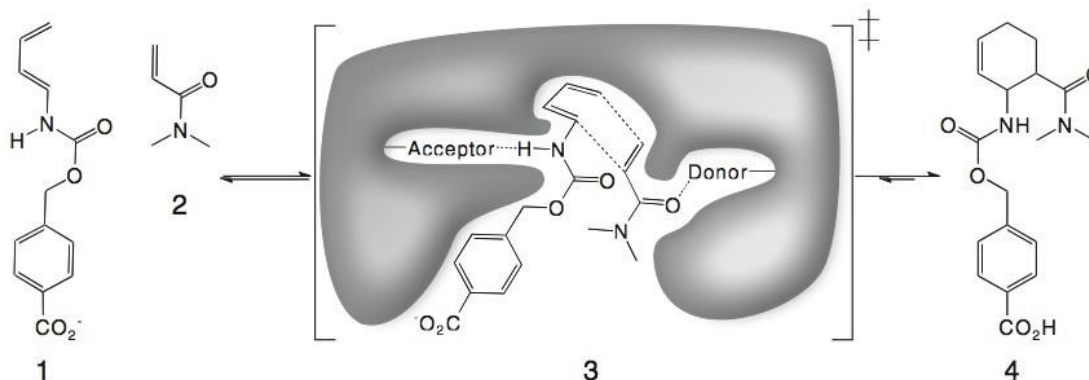
Kuhlman, B. *et al.* (2003). Design of a novel
globular protein fold with atomic-level accuracy.
Science 302, 1364–1368.

Atomic Level Accuracy of Design (blue) to X-ray structure (red)



Kuhlman, B. *et al.* (2003). Design of a novel globular protein fold with atomic-level accuracy. *Science* 302, 1364–1368.

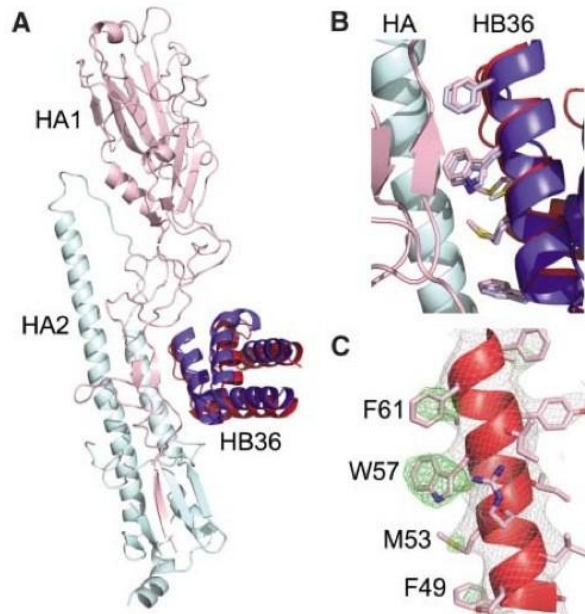
Novel Enzyme Design



RosettaMatch to search for a stable protein scaffold.

RosettaDesign to maximize transition state binding

Novel Protein Therapeutic Design



Design of interfaces with high shape complementarity

Optimized residue interactions at core region.

Blue is computational model

Design of Protein-Ligand Interfaces

RosettaMatch to identify stable backbone

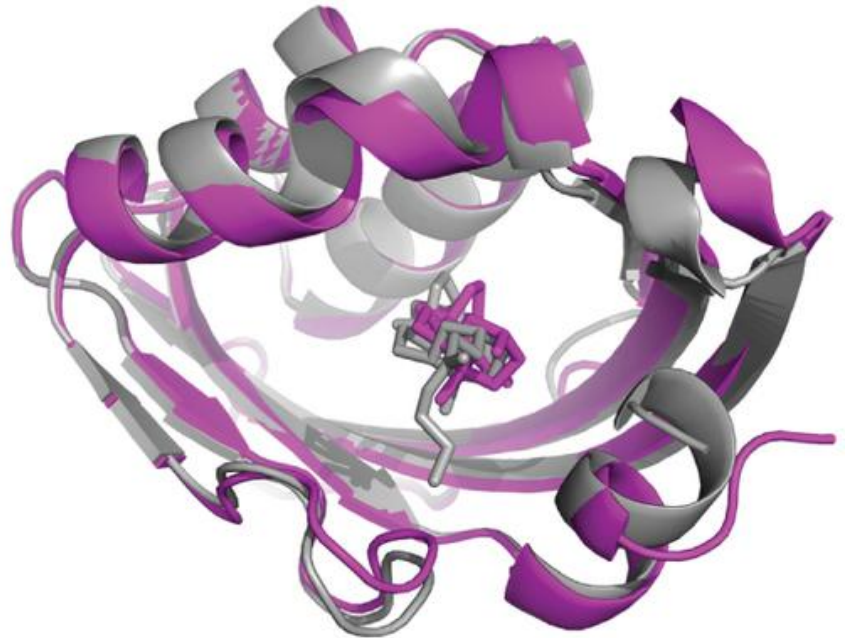
Sequence Design:

Round 1 = maximize binding affinity for ligand

Round 2 = protein stabilization

Computational model = grey

X-ray structure = purple



Tinberg, C.E., Khare, S.D., Dou, J., Doyle, L., Nelson, J.W., Schena, A., Jankowski, W., Kalodimos, C.G., Johnsson, K., Stoddard, B.L., Baker, D. (2013). Computational design of ligand-binding proteins with high affinity and selectivity. *Nature* 501 212-216

Additional Design Applications

- **Design of symmetric superfolds to understand protein folding evolution**

Fortenberry, C. *et al.* (2011). Exploring symmetry as an avenue to the computational design of large protein domains. *J. Am. Chem. Soc.* 133, 18026–18029.

- **Design of self-assembling proteins as nanomaterials**

King, N.P., Sheffler, W., Sawaya, M.R., Vollmar, B.S., Sumida, J.P., Andre, I., Gonen, T., Yeates, T.O., Baker, D. (2012). Computational Design of Self-Assembling Protein Nanomaterials with Atomic Level Accuracy. *Science* 336 1171-1174

- **Design of a thermally stabilize enzyme**

Korkegian, A., Black, M.E., Baker, D., and Stoddard, B.L. (2005). Computational thermostabilization of an enzyme. *Science* 308, 857–860.

- **Rational epitope design**

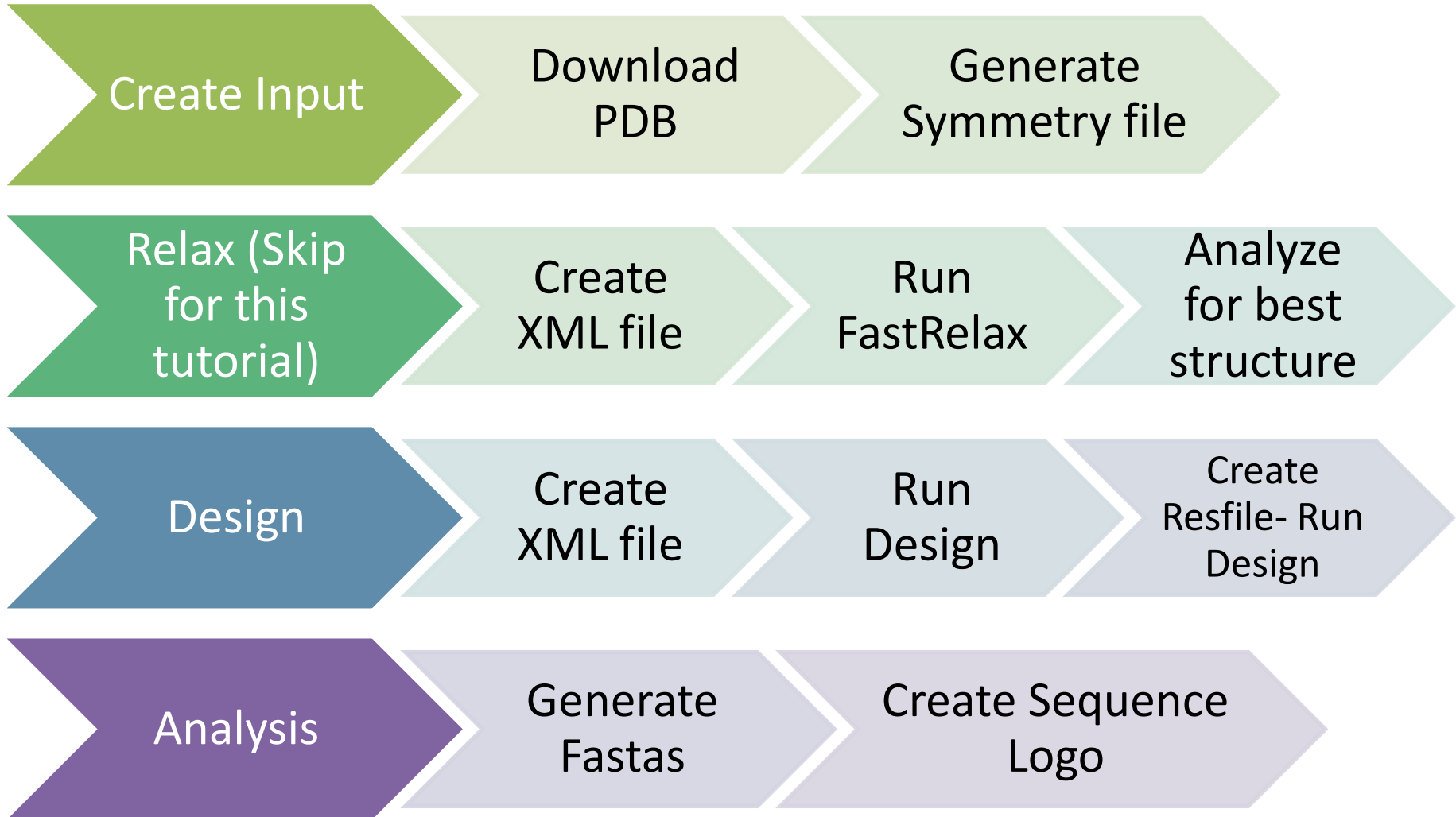
Wu, X., et al. (2010). Rational design of envelope identifies broadly neutralizing human monoclonal antibodies to HIV-1. *Science* 329, 856– 861.

- **Rational vaccine design**

Jardine, J., et al. (2013). Rational HIV Immunogen Design to Target Specific Germline B Cell Receptors. *Science*.

Tutorial Overview-Rosetta Design: Design of a Dimeric Membrane Protein

Protocol Overview



Rosetta Symmetry Definition File

symmetry_name 3UKM__2

$E = 2 * \text{VRT0_base} + 1 * (\text{VRT0_base} : \text{VRT1_base})$

anchor_residue COM

COM = center of mass, this is where
the subunit jump is anchored

virtual_coordinates_start

xyz VRT0 -0.9342759,-0.1279366,-0.3328074 0.1356705,-0.9907540,0.0000000 20.3383043,-
34.0374754,57.1640488

xyz VRT0_base -0.9342759,-0.1279366,-0.3328074 0.1356705,-0.9907540,0.0000000 28.3810913,-
32.9361230,60.0290476

xyz VRT1 0.9342759,0.1279366,0.3328074 -0.2876739,-0.2809746,0.9155856 20.3383043,-
34.0374754,57.1640488

xyz VRT1_base 0.9342759,0.1279366,0.3328074 -0.2876739,-0.2809746,0.9155856 12.2955173,-
35.1388277,54.2990500

xyz VRT 1.0000000,0.0000000,0.0000000 0.0000000,1.0000000,0.0000000 21.3383043,-
34.0374754,57.1640488

virtual_coordinates_stop

Each set of virtual coordinate is made of three triples. This represents unit vectors of X, Y
and Origin for X, Y and Origin. Named by VRT__ or whatever you'd like to name it

Energy equation for a dimer.
Considers energy of single
protomer X 2, then
interaction of a protomer
with its neighbor

Rosetta Symmetry Definition File Cont...

```
connect_virtual JUMP0_to_com VRT0 VRT0_base
connect_virtual JUMP0_to_subunit VRT0_base SUBUNIT
connect_virtual JUMP1_to_com VRT1 VRT1_base
connect_virtual JUMP1_to_subunit VRT1_base SUBUNIT
connect_virtual JUMP0 VRT VRT0
connect_virtual JUMP1 VRT0 VRT1
set_dof JUMP0_to_com x(8.60857814358724) angle_x
set_dof JUMP0_to_subunit angle_x angle_y angle_z
```

Encodes the various jumps to create the correctly calculated symmetric structure

```
set_jump_group JUMPGROUP2 JUMP0_to_com
JUMP1_to_com
set_jump_group JUMPGROUP3 JUMP1_to_subunit
JUMP0_to_subunit
```

Sets jump groups, essentially the protocol

The Residue File (example)

Resfile that mix and match packing tasks

<header>

NATAA #keep native amino acid at all positions (not specified below)

START

<body>

#position chain command

29 A ALLAA #allow all amino acids to be sampled (including cys)

62 A APOLAR #sample apolar amino acids only

65 A PIKAA VI

66 A PIKAA LVS

67 A POLAR #sample polar amino acids only

68 A PIKAA A

69 A APOLAR

70 A PIKAA NRG

94 A PIKAA TI

95 A PIKAA TIV

96 A PIKAA G

97 A PIKAA YFL

99 A PIKAA HNYD #sample H,N,Y, and D at position 99 (in chain A, but since we are in symmetry mode, it will apply to all chains)

RosettaScripts – Symmetry + Fixed BB Design

```
<ROSETTASCRIPTS>
  <SCOREFXNS>
    <mem_highres weights="membrane_highres_Menv_smooth.wts" symmetric="1"/>
      #made existing scoring function symmetric
  </SCOREFXNS>
  <TASKOPERATIONS>
    <InitializeFromCommandline name="ifcl"/>
  </TASKOPERATIONS>
  <MOVERS>
    <SetupForSymmetry name="setup_symm" definition="3UKM.symm" />
      #create symmetric oligomer
    <SymPackRotamersMover name="sym_pack" scorefxn="mem_highres"
task_operations="ifcl"/>
      #enable symmetric version of PackRotamer
  </MOVERS>
  <PROTOCOLS>
    <Add mover_name="setup_symm" />
      #First, set up symmetry
    <Add mover_name="sym_pack" />
      #Then, design
  </PROTOCOLS>
</ROSETTASCRIPTS>
```

RosettaScripts – Symmetry + Fixed BB Design

```
<ROSETTASCRIPTS>
  <SCOREFXNS>
    <mem_highres weights="membrane_highres_Menv_smooth.wts" symmetric="1"/>
      #made existing scoring function symmetric
  </SCOREFXNS>
  <TASKOPERATIONS>
    <InitializeFromCommandline name="ifcl"/>
  </TASKOPERATIONS>
  <MOVERS>
    <SetupForSymmetry name="setup_symm" definition="3UKM.symm" />
      #create symmetric oligomer
    <SymPackRotamersMover name="sym_pack" scorefxn="mem_highres"
task_operations="ifcl"/>
      #enable symmetric version of PackRotamer
  </MOVERS>
  <PROTOCOLS>
    <Add mover_name="setup_symm" />
      #First, set up symmetry
    <Add mover_name="sym_pack" />
      #Then, design
  </PROTOCOLS>
</ROSETTASCRIPTS>
```

RosettaScripts – Symmetry + Fixed BB Design

```
<ROSETTASCRIPTS>
  <SCOREFXNS>
    <mem_highres weights="membrane_highres_Menv_smooth.wts" symmetric="1"/>
      #made existing scoring function symmetric
  </SCOREFXNS>
  <TASKOPERATIONS>
    <InitializeFromCommandline name="ifcl"/>
  </TASKOPERATIONS>
  <MOVERS>
    <SetupForSymmetry name="setup_symm" definition="3UKM.symm" />
      #create symmetric oligomer
    <SymPackRotamersMover name="sym_pack" scorefxn="mem_highres"
task_operations="ifcl"/>
      #enable symmetric version of PackRotamer
  </MOVERS>
  <PROTOCOLS>
    <Add mover_name="setup_symm" />
      #First, set up symmetry
    <Add mover_name="sym_pack" />
      #Then, design
  </PROTOCOLS>
</ROSETTASCRIPTS>
```

RosettaScripts – Symmetry + Fixed BB Design

```
<ROSETTASCRIPTS>
  <SCOREFXNS>
    <mem_highres weights="membrane_highres_Menv_smooth.wts" symmetric="1"/>
      #made existing scoring function symmetric
  </SCOREFXNS>
  <TASKOPERATIONS>
    <InitializeFromCommandline name="ifcl"/>
  </TASKOPERATIONS>
  <MOVERS>
    <SetupForSymmetry name="setup_symm" definition="3UKM.symm" />
      #create symmetric oligomer
    <SymPackRotamersMover name="sym_pack" scorefxn="mem_highres"
task_operations="ifcl"/>
      #enable symmetric version of PackRotamer
  </MOVERS>
  <PROTOCOLS>
    <Add mover_name="setup_symm" />
      #First, set up symmetry
    <Add mover_name="sym_pack" />
      #Then, design
  </PROTOCOLS>
</ROSETTASCRIPTS>
```

RosettaScripts – Symmetry + Fixed BB Design

```
<ROSETTASCRIPTS>
```

```
<SCOREFXNS>
```

```
<mem_highres weights="membrane_highres_Menv_smooth.wts" symmetric="1"/>  
#made existing scoring function symmetric
```

```
</SCOREFXNS>
```

```
<TASKOPERATIONS>
```

```
<InitializeFromCommandline name="ifcl"/>
```

```
</TASKOPERATIONS>
```

```
<MOVERS>
```

```
<SetupForSymmetry name="setup_symm" definition="3UKS.symm" />  
#create symmetric oligomer
```

```
<SymPackRotamersMover name="sym_pack" scorefxn="mem_highres"  
task_operations="ifcl"/>  
#enable symmetric version of PackRotamer
```

```
</MOVERS>
```

```
<PROTOCOLS>
```

```
<Add mover_name="setup_symm" />  
#First, set up symmetry
```

```
<Add mover_name="sym_pack" />  
#Then, design
```

```
</PROTOCOLS>
```

```
</ROSETTASCRIPTS>
```



Conclusions

- This protocol is VERY simplified. You may have to adjust your protocol to perform several iterations of backbone movements and design.
- Can you implement and/or combine other movers you have learned about this week to aid in designing an optimal sequence?
RosettaScripts is the perfect tool to wrap design in these other protocols.

*****Something for you to think about*****

What types of information /data can you use to guide design?

(Feel free to discuss this with one of us!)

For the complete user's guide:

https://www.rosettacommons.org/manuals/archive/rosetta3.5_user_guide/index.html

Basic ways to get to design:

FixBB design application

~/Rosetta/main/source/bin/fixbb.default.linuxgccrelease

Movers (in RosettaScripts):

```
<PackRotamersMover name=pack  
  scorefxn=myscfxn.wts  
  task_operations=ifcl,rrf/>
```

Sampling Amino Acid Sequences over a Fixed Backbone

<http://www.youtube.com/watch?v=oWNgtFH6g9Q>