Become a RosettaScript developer!

Rosetta@DTU
May 2016

Christoffer Norn
Lund University
ch.norn@gmail.com

Fleishman et al. 2011, PLOS ONE
Overview

• Rationale for Rosetta Scripts
• Movers and filters
• TaskOperations
• Large-scale simulations
• Modifying score functions and constraints
• Building useful and more complex protocols
• Multi-objective scoring functions
• Is Rosetta practically useful for your problem?
• Exercise: Make a protocol for DDG calculation
• Best practices in loop modeling
Rationale for the XML script

- Rosetta is a +3 million lines of C++ code monster.
- Most functionalities you need is in there.
- RosettaScripts provides access without having to understand the C++ code.
- Rapid protocol testing (no compilation time).
Movers and filters

Example: Protein-protein docking
Movers and filters

• All steps in a simulation can be described as either those that change the protein (Mover) or those that report a certain value (Filter)
• Filters doesn’t change the protein or its internal representation
• Movers are everything else

<table>
<thead>
<tr>
<th>Movers</th>
<th>Filters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Docking</td>
<td>Binding energy</td>
</tr>
<tr>
<td>Design</td>
<td>Total energy</td>
</tr>
<tr>
<td>Loop model</td>
<td>Mutational scans</td>
</tr>
</tbody>
</table>
Movers and filters

read structure

Mover 1

Filter 1

pass

Mover N

Filter N

pass

write structure

fail

fail
RosettaScript XML code is like a recipe

<ROSETTASCRPT>
  <FILTERS> Ingredients
  </FILTERS>
  <MOVERS> Ingredients
  </MOVERS>
  <PROTOCOLS> Order of operations
  </PROTOCOLS>
</ROSETTASCRPT>
MinMover

Does minimization over sidechain and/or backbone

```
<MinMover name="&string" scorefxn=(score12 &string) chi=(&bool) bb=(&bool)
```

- MinMover is also sensitive to a MoveMap block, just like FastRelax.
- scorefxn: scorefunction to use during minimization
- chi: minimize sidechains?
- bb: minimize backbone?
Docking with RosettaScripts

<ROSETTASCRIPTS>
  <FILTERS>
    <Ddg name=ddg confidence=1 threshold=-12/>
  </FILTERS>

  <MOVERS>
    <Prepack name=prepack/>
    <Docking name=dock fullatom=1 local_refine=1/>
    <MinMover name=min bb=0 chi=1 jump=1/>
  </MOVERS>

  <PROTOCOLS>
    <Add mover=prepack/>
    <Add mover=dock/>
    <Add mover=min/>
    <Add filter=ddg/>
  </PROTOCOLS>
</ROSETTASCRIPTS>
Docking with RosettaScripts

<ROSETTASCRIPITCS>
  <FILTERS>
    <Ddg name=ddg confidence=1 threshold=-12/>
  </FILTERS>

  <MOVERS>
    <Prepack name=prepack/>
    <Docking name=dock fullatom=1 local_refine=1/>
    <MinMover name=min bb=0 chi=1 jump=1/>
  </MOVERS>

  <PROTOCOLS>
    <Add mover=prepack/>
    <Add mover=dock/>
    <Add mover=min/>
    <Add filter=ddg/>
  </PROTOCOLS>
</ROSETTASCRIPITCS>

Get the format right!
Your most-favorite protein-complex of the day: Lysozyme-Ab (1MLC)

**Prepare your protein**
- Remove ligands (or make param files)
- Remove waters
- (Reorder chains)
- Renumber residues
DIY: Run a simple docking script

• Go to your home directory:
  – ssh xx
  – cd rosettaWorkshop/exercise_1
• Have a look at the flags file in the in/
• Check that you understand what run.sh contains and run it:
  ./run.sh
Let’s add some refinement

• Off-rotamer minimization of side-chains is important in docking. Let’s add that (RotamerTrialsMinMover).

https://www.rosettacommons.org/docs/latest/scripting_documentation/RosettaScripts/RosettaScripts

• How is the energy affected?

Docking with RosettaScripts

<ROSETTASCRIPITS>
  ...

  <MOVERS>
    <Prepack name=prepack/>
    <Docking name=dock fullatom=1 local_refine=1/>
    <MinMover name=min bb=0 chi=1 jump=1/>
    <RotamerTrialsMinMover name=rtmin/>
  </MOVERS>

  <PROTOCOLS>
    <Add mover=prepack/>
    <Add mover=dock/>
    <Add mover=rtmin/>
    <Add mover=rtmin/>
    <Add mover=rtmin/>
    <Add mover=rtmin/>
    <Add mover=min/>
    <Add filter=ddg/>
  </PROTOCOLS>
</ROSETTASCRIPITS>
Task Operations

Example: Protein-protein interface design
Protein sequence design

**Goal**
Optimize affinity in antibody-antigen complex, by Ab design.

**Note:**
- Target only interface residues on Ab.
The sequence design mover

<_packRotamersMover name=design task_operations=(str, str) />
TaskOperations governs movers

<ROSETTASCRIPTS>
  <TASKOPERATIONS>
  </TASKOPERATIONS>
  <FILTERS>
  </FILTERS>
  <MOVERS>
  </MOVERS>
  <PROTOCOLS>
  </PROTOCOLS>
</ROSETTASCRIPTS>

Default residue behavior
Packable + designable!

Exercise 2
Use the taskOperation ProteinInterfaceDesign to allow:
  • Lysozyme: Repacking of interface residues
  • Ab: Repacking and design of interface residues (chain 2)
  • Interface = 10 Å cutoff.

-PackRotamersMover name=design task_operations=(str, str)/>
Antibody design with taskOPs

<ROSETTASCRIPTS>

<TASKOPERATIONS>

<ProteinInterfaceDesign name=pido
    design_chain1=0 design_chain2=1
    interface_distance_cutoff=10/>

</TASKOPERATIONS>

<MOVERS>

-PackRotamersMover name=design
    task_operations=pido/>

</MOVERS>

<PROTOCOLS>

-Add mover=design

</PROTOCOLS>

</ROSETTASCRIPTS>
Antibody design with taskOPs

<ROSETTASCRIPITTS>

<TASKOPERATIONS>
    <ProteinInterfaceDesign name=pido
design_chain1=0 design_chain2=1
interface_distance_cutoff=10/>
</TASKOPERATIONS>

<MOVERS>
    <PackRotamersMover name=design
task_operations=pido/>
</MOVERS>

<PROTOCOLS>
    <Add mover=design>
</PROTOCOLS>

</ROSETTASCRIPITTS>

Flag file
-ex1
-ex2
-use_input_sc
Antibody design with taskOPs

<ROSETTASCRIPTS>

<TASKOPERATIONS>

<ProteinInterfaceDesign name=pido
design_chain1=0 design_chain2=1
interface_distance_cutoff=10/>
<InitializeFromCommandLine name=init>
</TASKOPERATIONS>

<MOVERS>

<PackRotamersMover name=design
task_operations=pido,init/>
</MOVERS>

<PROTOCOLS>

<Add mover=design>
</PROTOCOLS>

</ROSETTASCRIPTS>

Flag file
-ex1
-ex2
-use_input_sc
Recap: Properties of TaskOperations

• Provides easy control of movers.
• Applied only when requested, so if binding orientation changes during simulation, they will be accordingly up-to-date.
• When applying multiple taskOPs together, the most restrictive interpretation is chosen.

<ProteinInterfaceDesign name=pido design_all_aas=1/>
<RestrictAbsentCanonicalAAS name=nopro keep_aas="ACDEFGHIKLMQRTVWY"/>
Recap: Properties of TaskOperations

- Provides easy control of movers.
- Applied only when requested, so if binding orientation changes during simulation, they will be accordingly up-to-date.
- When applying multiple taskOPs together, the most restrictive interpretation is chosen.

```
<ProteinInterfaceDesign name=pido design_all_aas=1/>
<RestrictAbsentCanonicalAAS name=nopro keep_aas="ACDEFGHIKLMQRSTVWY"/>
```

Filter: `<DesignableResidues name=(&string) task_operations=(comma-separated list)>`: Write to outstream the status of all residues when applying the listed task_operations.
Large-scale simulations

Example: Antibody design
Appropriate Number of Structures ("nstruct")

$10^0$ deterministic algorithms: minimize, score

$10^1$ structure preparation: tightly constrained relax, repack a shell of residues after making a point mutation on a good structure

$10^2$ internal sampling: coupled moves, fixed backbone design

$10^3$ structure preparation: relax with looser constraints, optimal for production-scale runs; repack and dock peptidomimetics

$10^4$ dock-design algorithms: repack a protein and dock/design a binding partner, perhaps with minimization at the end

$10^5$ design plus flexibility: FastRelax, Backrub, or some kind of loop modeling combined with design, Complex composite protocols

$10^6$ structure determination: ab initio folding, homology modeling from poor MSA

rosettacommons.org
One trajectory is not enough

- Modify the flags file to generate 100 structures
  - -nstruct 100
- Modify the flags file to generate silent files
  - -out:file:silent pdbs/structures.out
  - -out:file:silent_struct_type binary
- Mute out stream
  - -mute all
- ./run.sh on 1000 CPUs (or not)
- Extract the best scored structure (use the precomputed file).
A warning about Rosetta

• Look at the best model in your favorite viewer. Does everything look alright?

DEMONSTRATION
Rosetta makes choices that nature never makes: A natural antibody

Red = choices that occur in extremely rarely
Blue = choices that occur often
Rosetta makes choices that nature never makes: A designed antibody

Red = choices that occur in extremely rarely
Blue = choices that occur often
Think hard about your design goals!

How do you avoid kinetic traps?

Energy landscape of fast folding protein vs Energy landscape with kinetic traps

Graphics of Ken Dill lab
Watters et al. 2007, Cell (TOP7 has metastable intermediate)
Think hard about your design goals!

How do you avoid alternative states?

Energy landscape of fast folding protein

vs

Energy landscape with alternative conformation

Graphics of Ken Dill lab
Karanicolas et al. 2011, Molecular Cell (a failed attempt at protein-protein interaction design)
Helpful pointers

Use PSSMs to bias your scoring function!

Figure showing preliminary results, showing that expression level (~stability) for antibodies is increased by the use of PSSMs in design

Norn & Baran & Fleishman, unpublished
PSSMs implicitly encode negative design elements

Norn & Baran & Fleishman, unpublished
Helpful pointers

- Use PSSMs to bias your scoring function
- Use negative design (explicit/implicit)
- Forward fold filters (does the energy landscape funnel?)
Score functions
Modifying scorefxns can assist sampling

Customize your scoring function?

Current default behavior is Talaris [see ref.], fully recapitulates H-bond preferences from QM simulations.

Modifications are relevant for sampling purposes.

O’meara et al. 2015, JCTC
Modifying scorefxns can assist sampling

Reduced penalty for vdw clashes
How to add PSSM seq constraints

<ROSETTASCRIPITXS>
  <SCOREFXNS>
    <soft_rep_w_csts weights=soft_rep>
      <Reweight scoretype=res_type_constraint weight=0.2/>
    </soft_rep_w_csts>
  </SCOREFXNS>

  <MOVERS>
    </MOVERS>

  <PROTOCOLS>
    </PROTOCOLS>
  </PROTOCOLS>
</ROSETTASCRIPITXS>
How to add PSSM seq constraints

<ROSETTAScripts>
  <ScoreFxnS>
    <soft_rep_w_csts weights=soft_rep>
      <Reweight scoretype=res_type_constraint weight=0.2/>
    </soft_rep_w_csts>
  </ScoreFxnS>
  
  <Movers>
    <FavorSequenceProfile name=add_PSSM pssm=PSSM.txt/>
  </Movers>
  
  <Protocols>
    <Add mover=add_PSSM/>
  </Protocols>
</ROSETTAScripts>
How to add PSSM seq constraints

<ROSETTASCRIPITS>
  <SCOREFXNS>
    <soft_rep_w_csts weights=soft_rep>
      <Reweight scoretype=res_type_constraint weight=0.2/>
    </soft_rep_w_csts>
  </SCOREFXNS>

  <MOVERS>
    <FavorSequenceProfile name=add_PSSM pssm="%%pssm_path%%"/>
  </MOVERS>

  <PROTOCOLS>
    <Add mover=add_PSSM/>
  </PROTOCOLS>
</ROSETTASCRIPITS>

-subparser:script_vars foo='X'
subtistutes %foo in XML
How to add PSSM seq constraints

<ROSETTASCRIPTS>
  <SCOREFXNS>
    <soft_rep_w_csts weights=soft_rep>
      <Reweight scoretype=res_type_constraint weight=0.2/>
    </soft_rep_w_csts>
  </SCOREFXNS>
  <MOVERS>
    <FavorSequenceProfile name=add_PSSM pssm="%%pssm_path%%"/>
    <PackRotamersMover name=design_soft score_fxn=soft_rep_w_csts/>
  </MOVERS>
  <PROTOCOLS>
    <Add mover=add_PSSM/>
    <Add mover=design_soft/>
  </PROTOCOLS>
</ROSETTASCRIPTS>
How to add coordinate constraints

<ROSETTASCRIPTS>
  <SCOREFXNS>
    <soft_rep_w_csts sweights=soft_rep>
      <Reweight scoretype=coordinate_constraint weight=0.1/>
    </soft_rep_w_csts>
  </SCOREFXNS>

  <MOVERS>
    <TaskAwareCsts name=add_csts/> mean=0, sd=1
    <MinMover name=bb_min bb=1 score_fxn=soft_rep_w_csts/>
    <ClearConstraintsMover name=clear_csts/>
  </MOVERS>

  <PROTOCOLS>
    <Add mover=add_csts/> acts on pose till rm’ed
    <Add mover=bb_min/>
    <Add mover=clear_csts/>
  </PROTOCOLS>
</ROSETTASCRIPTS>
How to add coordinate constraints 2

<ROSETTASCRPTS>
  <SCOREFXNS>
    <soft_rep_w_csts weights=soft_rep>
      <Reweight scoretype=coordinate_constraint weight=0.1/>
    </soft_rep_w_csts>
  </SCOREFXNS>

  <MOVERS>
    <ConstraintSetMover name=add_csts add_constraints=1
      cst_file=%%%filepath%%%/>  see doc for cst file format
    <MinMover name=bb_min bb=1 score_fxn=soft_rep_w_csts/>
    <ClearConstraintsMover name=clear_csts/>
  </MOVERS>

  <PROTOCOLS>
    <Add mover=add_csts/> acts on pose till rm’ed
    <Add mover=bb_min/>
    <Add mover=clear_csts/>
  </PROTOCOLS>
</ROSETTASCRPTS>
Parsed protocols and MC movers
Parsed protocols contains blocks of movers

<ROSETTASCRIPTS>
  <MOVERS>
    ...
    <ParsedProtocol name=proper_design>
      <Add mover=design_softrep/>
      <Add mover=soft_min/>
      <Add mover=hard_min/>
      <Add mover=design_hard/>
      <Add mover=rtmin_x_3/>
      <Add mover=hard_min/>
    </ParsedProtocol>
  </MOVERS>
  <PROTOCOLS>
    <Add mover=proper_design/>
  </PROTOCOLS>
</ROSETTASCRIPTS>
parsed protocols contains blocks of movers

<ROSETTASCRIPTS>
    <MOVERS>
        ...
        <ParsedProtocol name=proper_design>
            <Add mover=design_softrep/>
            <Add mover=soft_min/>
            <Add mover=hard_min/>
            <Add mover=design_hard/>
            <Add mover=rtmin_x_3/>
            <Add mover=hard_min/>
        </ParsedProtocol>
        <GenericMonteCarlo name=mc_design
            mover_name=proper_design trials=30 temperature=1/>
    </MOVERS>

    <PROTOCOLS>
        <Add mover=mc_design/>
    </PROTOCOLS>
</ROSETTASCRIPTS>
Multi-objective scoring functions

Design goals:
• Fraction of folded Ab.
• Fraction bound antigen
Fractional occupancies of the target state

\[ U \Leftrightarrow F \]

\[ f = \frac{[F]}{[U]+[F]} \]

\[ K_{eq} = \frac{[F]}{[U]} = e^{-\Delta G/RT} \]

\[ f = \frac{1}{1+e^{\Delta G/RT}} \]

\[ f = \frac{1}{1+e^{(\text{energy-offset)/RT}}} \]

Warszawski et al. 2014, JMB
Fractional occupancies from stability

Warszawski et al. 2014, JMB
Balancing different design goals

\[ \Delta G_{\text{folding}}, \ \Delta G_{\text{binding}}, \ \text{etc...} \]

\[ f_1, \ f_2, \ \ldots, \ f_n \]

Optimization objective function \[ = f_1 \land f_2 \lor \ldots \lor \neg f_n \]

Simulated Annealing Monte Carlo

Warszawski et al. 2014, JMB
Balancing different design goals

<ROSETTASCRIPITS>
  <FILTERS>
    <Sigmoid name=s_bind filter=ddg
      offset=-5 steepness=1/>
    <Sigmoid name=s_stability filter=stability
      offset=-150 steepness=1/>
  <FILTERS/>

  <MOVERS>
  </MOVERS>

  <PROTOCOLS>
  </PROTOCOLS>
</ROSETTASCRIPITS>

(This protocol makes little sense in reality)
Balancing different design goals

<ROSETTASCRPTS>
<FILTERS>
  <Sigmoid name=s_bind filter=ddg
    offset=-5 steepness=1/>
  <Sigmoid name=s_stability filter=stability
    offset=-150 steepness=1/>
  <Operator name=b_x_s operation=PRODUCT
    filters=s_bind,s_stability negate=1 logarithm=1/>
</FILTERS>

<MOVERS>
</MOVERS>

<PROTOCOLS>
</PROTOCOLS>
</ROSETTASCRPTS>

(this protocol makes little sense in reality)
Balancing different design goals

<ROSETTASCRIPTS>
  <FILTERS>
    <Sigmoid name=s_bind filter=ddg
      offset=-5 steepness=1/>
    <Sigmoid name=s_stability filter=stability
      offset=-150 steepness=1/>
    <Operator name=b_x_s operation=PRODUCT
      filters=s_bind,s_stability negate=1 logarithm=1/>
  </FILTERS>

  <MOVERS>
    <ParsedProtocol name=proper_design> ... </ParsedProtocol>
    <GenericMonteCarlo name=mc mover_name=proper_design
      filter_name=b_x_s trials=30 temperature=1/>
  </MOVERS>

  <PROTOCOLS>
    <Add mover=mc/>
  </PROTOCOLS>
</ROSETTASCRIPTS>

These need to be specified as well

(this protocol makes little sense in reality)
Is Rosetta practically useful for your problem?
Factors affecting Rosetta performance

- Energy gap size
- Folding
- Loop conformations
- Small molecule binding mode
- Alternative states

Fleishman & Baker, 2012, Cell
Factors affecting Rosetta performance

- Energy gap size
- Folding
- Loop conformations
- Small molecule binding mode
- Alternative states

Structure inaccuracy
- De novo predicted models
- Homology models
- Crystal structures
Factors affecting Rosetta performance

- Energy gap size
  - Folding
  - Loop conformations
  - Small molecule binding mode
  - Alternative states

<table>
<thead>
<tr>
<th>Structure inaccuracy</th>
<th>Force field inaccuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>De novo predicted models</td>
<td>BB potentials</td>
</tr>
<tr>
<td>Homology models</td>
<td>H-bond networks</td>
</tr>
<tr>
<td>Crystal structures</td>
<td>Core packing</td>
</tr>
</tbody>
</table>
Factors affecting Rosetta performance

- Energy gap size
- Folding
- Loop conformations
- Small molecule binding mode
- Alternative states

Structure inaccuracy:
- De novo predicted models
- Homology models

Force field inaccuracy:
- H-bond networks
- BB potentials
- Core packing
- Crystal structures
Augmenting poor FF/models with mutational data

Norn et al. 2015, Structure
Augmenting poor FF/models with NMR data

Ramen et al. 2010, Science.
Dealing with ligands
Dealing with ligands

- Small molecules, e.g., ATP, are not defined by default in Rosetta and require user input.

- If the ligand is in a PDB file, go to pymol, select the ligand, extract to object, then save the object with file name suffix .mol. (more accurate ways of parametrizing ligands exist).

- This will produce a mol file, containing the ligand's atoms and connectivities. Next, from the Rosetta directory run, src/python/apps/public/molfile_to_params.py <MOL_FILE>

- This will produce a Rosetta recognized .params file and a PDB file for the ligand. Remove the ligand atoms from the original PDB file and replace them with the PDB file produced by this script. Now you're ready to run whatever Rosetta protocol with this ligand:

  - rosetta_scripts... -extra_res_fa < .params file >