Practicals

Design your protein :-)

Thomas Schiex

November/December 2017

Cargèse, Corsica, France

Thanks

Rosetta, PyRosetta and beta_nov16

- Rosetta: RosettaCommons, long list of people (D. Baker PI)
- PyRosetta: Python bindings to Rosetta (Sergey Lyskov, Graylab, John Hopkins University)
- beta_nov16: Frank Di Maio (UW, not IPD: huge work, seems to do very well).

Planned program

Virtual machine: PyRosetta, scripts

- Preparing your system (minimal, PyRosetta)
- Computing energy matrices beta/PyRosetta (AMBER/EEF1/Osprey: See `SpeedUp2`)
- Solving the SCP problem with Pyrosetta and `toulbar2`
- Designing with PyRosetta and `toulbar2`
- Enumerating sequence·conformations, sequences only
- Incorporating fitness in the energy function.
- Affinity: $\Delta\Delta G$ and $\Delta\Delta E$

Missing: Forward folding

David Simoncini, Thomas Schiex, and Kam YJ Zhang. "Balancing exploration and exploitation in population-based sampling improves fragment-based de novo protein structure prediction". In: *Proteins: Structure, Function, and Bioinformatics* 85.5 (2017), pp. 852–858

Assumptions, trust

It would be nice to know

- physics, atoms, amino-acids, bonds, proteins, X-ray cristallography. . .
- Linux/Unix (shells)
- Python3
- toulbar2
- I cannot say you should know Rosetta (infeasible)

Assumptions, trust

It would be nice to know

- physics, atoms, amino-acids, bonds, proteins, X-ray cristallography. . .
- Linux/Unix (shells)
- Python3
- toulbar2
- I cannot say you should know Rosetta (infeasible)

- the provided Python scripts are part of a currently under revision submission.
- please do not distribute them.

Building energy matrices

Using either

- the design score function beta (Rosetta)
- the physical (MD) force field AMBER+EEF1 (`SpeedUp2`)

Building energy matrices

Using either

- the design score function beta (Rosetta)
- the physical (MD) force field AMBER+EEF1 (`SpeedUp2`)

All based on existing work

- AMBER/Osprey: Seydou Traoré et al. "A new framework for computational protein design through cost function network optimization". In: *Bioinformatics* 29.17 (2013), pp. 2129–2136
- Beta/Rosetta: David Simoncini et al. "Guaranteed Discrete Energy Optimization on Large Protein Design Problems". In: *Journal of chemical theory and computation* 11.12 (2015), pp. 5980–5989 for design, Clément Viricel et al. "Cost Function Network-based Design of Protein-Protein Interactions: predicting changes in binding affinity". In: *Under revision* (2017) for affinity.
- It is possible to design real proteins with this, already.

Usual preparation

Preparing structures

- X-ray cristallography/MNR/CryoEM have weaknesses
- Missing data: atoms (hydrogens or more)
- Precision: unrealistic positions (strained bonds, steric clashes,. . .)

Usual preparation

Preparing structures

- X-ray cristallography/MNR/CryoEM have weaknesses
- Missing data: atoms (hydrogens or more)
- Precision: unrealistic positions (strained bonds, steric clashes,. . .)

Preparing structures

- fill-in missing H (protons) at least
- adjust positions to fit with existing knowledge (radiuses, distances, angles)
- ideally using the force field you'll use to design

Minimization/relaxation

What is the difference?

- Minimization: continuous local optimisation of energy (cartesian coordinates or angles/distances), gradient based mostly.
- Relaxation: cycles of minimization and Monte-Carlo based Side-Chain Packing (SCP)
- energy usually biased by “harmonic potentials” to remain close to experimental data

Let’s do it first

```
cd TSc/single
ls
make clean
make showpars
make 1aho.rlx
```

Let’s do it first

```
cd TSc/single
ls
make clean
make showpars
make 1aho.rlx
```

Let’s dig a bit

- the Rosetta messages (disulfide bridges,. . .)
- the PDB files `1aho.pdb`, `1aho.rlx` (`pymol` both)
- the parameters (`pars`, all of them)
- the python script (`tb2cpd.py`: just the load/relax parts)
- the `Makefile`

Let's do it first

```
cd TSc/single
ls
make clean
make showpars
make 1aho.rlx
```

Let's dig a bit

- the Rosetta messages (disulfide bridges,. . .)
- the PDB files `1aho.pdb`, `1aho.rlx` (`pymol` both)
- the parameters (`pars`, all of them)
- the python script (`tb2cpd.py`: just the load/relax parts)
- the `Makefile`

Download another PDB and relax it. Error messages?

Side-Chain Packing

- Explain the `1aho.resfile`
- Edit the resfile to do “Side-Chain Päcking” only
- SCP 1ah0: `make 1aho.opt`
- Explore: `pymol 1aho.opt`, `less 1aho.show`

Side-Chain Packing

- Explain the `1aho.resfile`
- Edit the resfile to do "Side-Chain Päcking" only
- SCP 1ah0: `make 1aho.opt`
- Explore: `pymol 1aho.opt`, `less 1aho.show`

What happened: `Makefile`

- the relaxed PDB exists
- The energy matrix is computed (`.wcsp` format)
- `toulbar2` is there, so not downloaded using `git` (cpd branch)
- `toulbar2` is there, so not compiled
- an upper bound is computed using `fixbb` (often useless)
- `toulbar2` solves the `.wcsp` file and outputs the GMEConformation
- the conformation is used to create the associated PDB+stats

Experiment a bit

- choose your favorite monomer structure (PDB)
- change parameters (extended rotamers,. . .)
- side-chain pack it (resfile)

Experiment a bit

- choose your favorite monomer structure (PDB)
- change parameters (extended rotamers,. . .)
- side-chain pack it (resfile)

`toulbar2` should be able to optimally pack large proteins (¿1 000 AAs), and this even using the ex2 rotamer library. The largest we measured defined a space of size 10^{927} conformations. Takes more time.

Dig the Python script a bit

- how is the `wcsp` file extracted?
- how is `toulbar2` called? Which options?
- look into `toulbar2` options (just execute `toulbar2` with `-h`)
- how do we reconstruct the mutated PDB?

Dig the Python script a bit

- how is the wcsp file extracted?
- how is toulbar2 called? Which options?
- look into toulbar2 options (just execute toulbar2 with -h)
- how do we reconstruct the mutated PDB?

(Py)Rosetta

Very touchy. Needs suitable mantras and RotSets (sizes and indices in them) are context sensitive (pose, score function).

Designing one sequence·conformation

Just a matter of changing the resfile

- edit `1aho.resfile` and add mutable positions (`PIKAA/ALLAA`)
- problems are getting harder, pay attention!
- `make 1aho.gmec`, `make 1aho.opt` or other targets.

Listing all sequence·conformation within δ of GMEC

- Do this directly with `toulbar2` (in the exes directory)
- Choose a (small) threshold δ and compute an upper bound for `toulbar2`
- look into the `1aho.shft` file: energy shift and resolution.
- `./exes/toulbar2 -a -s -ub <ub> 1aho.wcsp` (HBFS)
- have a look to `toulbar2` web site.

Listing all sequences within δ of GMEC

- check the threshold and other parameters (`make showpars`)
- `make 1aho.enum`
- This uses DFS (not HBFS) + SCP-branching

Incorporating "fitness"

- Evolution of natural similar proteins give us indications on what matters beyond stability as the score function describes it (catalysis, agreggation, flexibility. . .).
- Recruit "similar" proteins using Psi-blast (in practice, some cleaning may be useful)
- Produce a "position specific score matrix" (see www.ncbi.nlm.nih.gov/books/NBK2590)
- check parameters.
- redesign with `1aho.pssm`
- Alternatively the native and a protein similarity matrix can be used.

Affinity in PPI

Install `toulbar2` (other version)

```
cd  ~/TSc; tar xvfz EasyE-JayZ.tar.gz
cd easy_jayz/exes
sh toulbar2-install.sh
```

Affinity in PPI

Install `toulbar2` (other version)

```
cd ~/TSc; tar xvfz EasyE-JayZ.tar.gz
cd easy_jayz/exes
sh toulbar2-install.sh
```

Estimating affinity by “potential” energy difference

```
cd ../Example
../exes/EasyE.py --pdb 1CBW.pdb --seq 1CBW.seq \
                 --partner FGH_I
```

Affinity in PPI

Install toulbar2 (other version)

```
cd  ~/TSc; tar xvfz EasyE-JayZ.tar.gz
cd easy_jayz/exes
sh toulbar2-install.sh
```

Estimating affinity by "potential" energy difference

```
cd ../Example
../exes/EasyE.py --pdb 1CBW.pdb --seq 1CBW.seq \
                 --partner FGH_I
```

Explanations

- EasyE: does $\Delta\Delta E$ computations
- –pdb: a PDB file with more than 1 chain
- –seq: the mutations that will be considered
- –partner: the two sides of the interaction
- results in associated directory

Affinity in PPI

Partition function based

```
../exes/JayZ.py --pdb 1CBW.pdb --seq 1CBW.seq \
                --partner FGH_I
```

Affinity in PPI

Partition function based

```
../exes/JayZ.py --pdb 1CBW.pdb --seq 1CBW.seq \
                --partner FGH_I
```

Affinity in PPI

Partition function based

```
../exes/JayZ.py --pdb 1CBW.pdb --seq 1CBW.seq \
                --partner FGH_I
```

Explanations

- JayZ: does $\Delta\Delta G$ computations
- similar syntax and output
- Much slower. Z computed only on residues with atoms within 3Å of mutable residues and after a global SCP. Largest integrated space: 10^{28}.

References I

[1] David Simoncini, Thomas Schiex, and Kam YJ Zhang. “Balancing exploration and exploitation in population-based sampling improves fragment-based de novo protein structure prediction”. In: *Proteins: Structure, Function, and Bioinformatics* 85.5 (2017), pp. 852–858.

[2] David Simoncini et al. “Guaranteed Discrete Energy Optimization on Large Protein Design Problems”. In: *Journal of chemical theory and computation* 11.12 (2015), pp. 5980–5989.

[3] Seydou Traoré et al. “A new framework for computational protein design through cost function network optimization”. In: *Bioinformatics* 29.17 (2013), pp. 2129–2136.

[4] Clément Viricel et al. “Cost Function Network-based Design of Protein-Protein Interactions: predicting changes in binding affinity”. In: *Under revision* (2017).