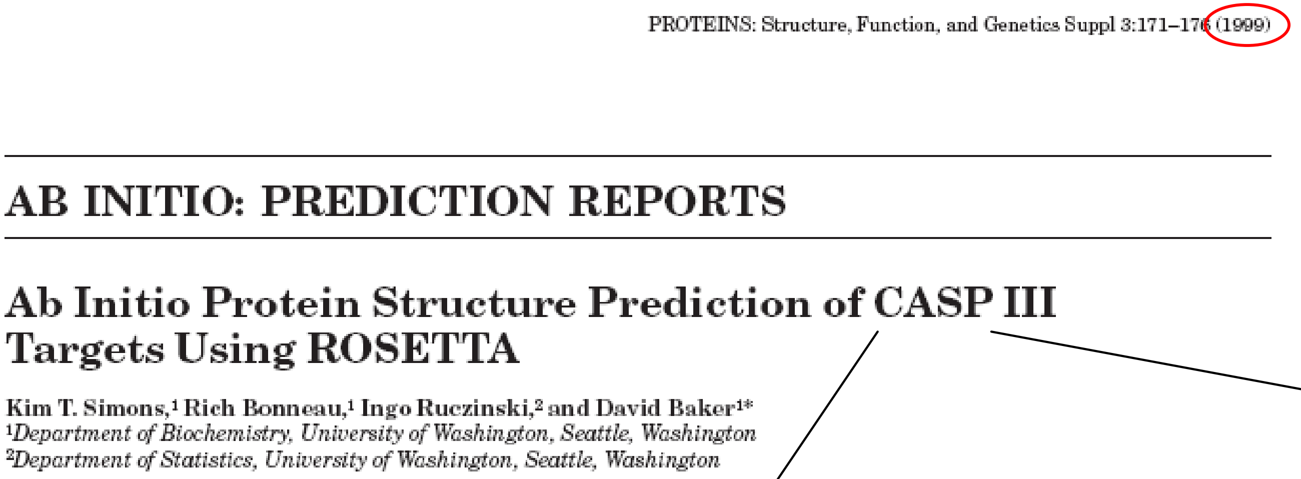
A Framework for Understanding Rosetta

Xavier Ambroggio

Rosetta Design Group

- ➥ Origin of Rosetta
- ➥ Introduction to Basic Rosetta Methodology
- ➥ Overview of Rosetta Implementation

Rosetta: an algorithm for *ab initio* structure prediction



PROTEINS: Structure, Function, and Genetics Suppl 3:171–176 (1999)

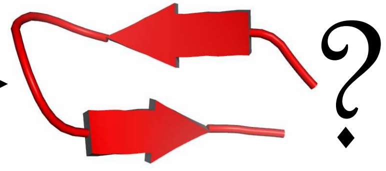
AB INITIO: PREDICTION REPORTS

Ab Initio Protein Structure Prediction of CASP III Targets Using ROSETTA

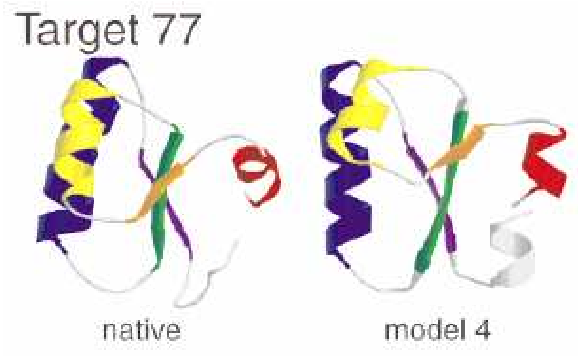
Kim T. Simons,[1] Rich Bonneau,[1] Ingo Ruczinski,[2] and David Baker[1]*
[1]Department of Biochemistry, University of Washington, Seattle, Washington
[2]Department of Statistics, University of Washington, Seattle, Washington

Critical **A**ssessment of Techniques for Protein **S**tructure **P**rediction

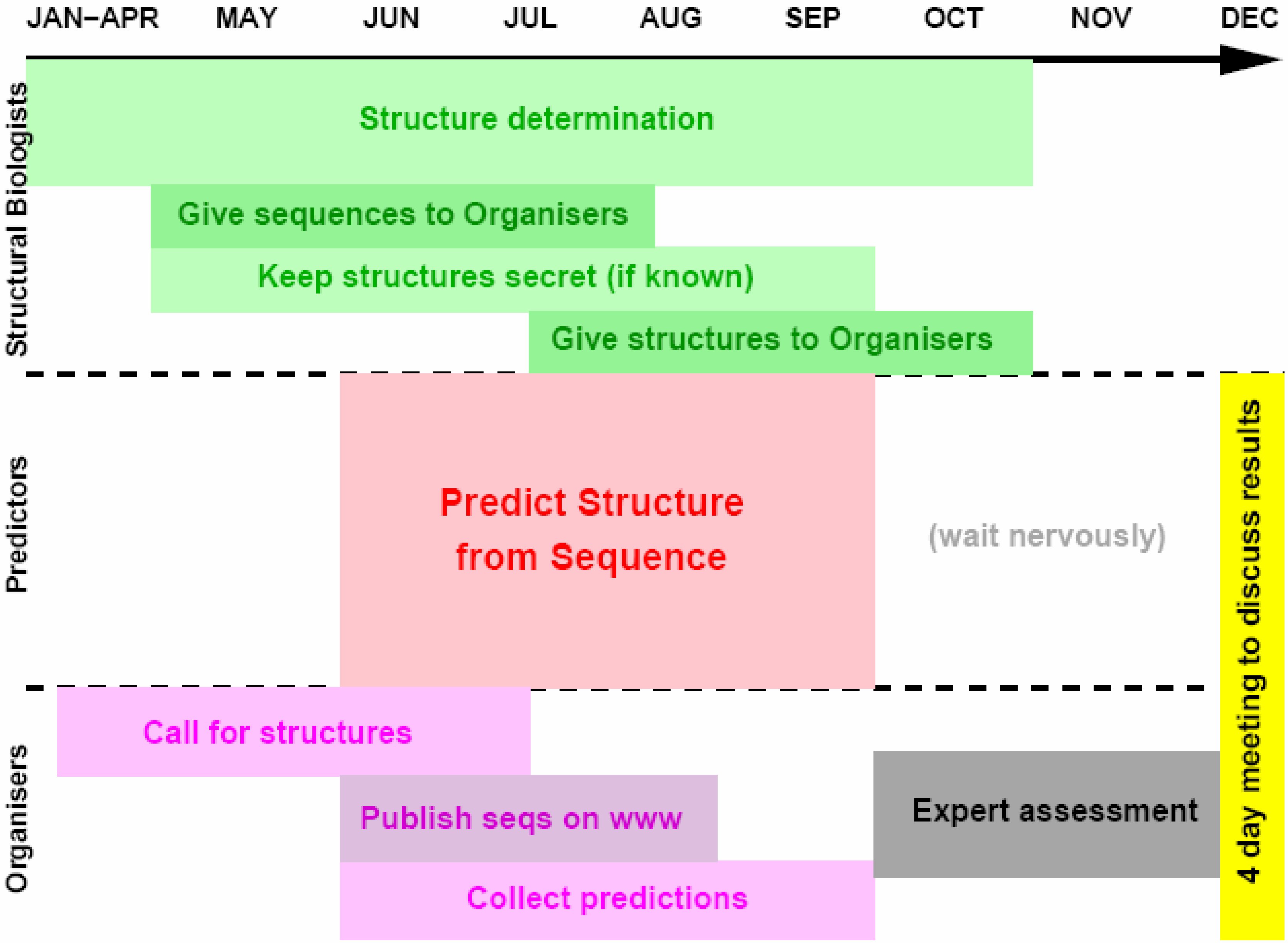
CTWEGNKLTC →



protein folding problem



CASP



JAN–APR MAY JUN JUL AUG SEP OCT NOV DEC

Structural Biologists

Structure determination

Give sequences to Organisers

Keep structures secret (if known)

Give structures to Organisers

Predictors

Predict Structure from Sequence

(wait nervously)

Organisers

Call for structures

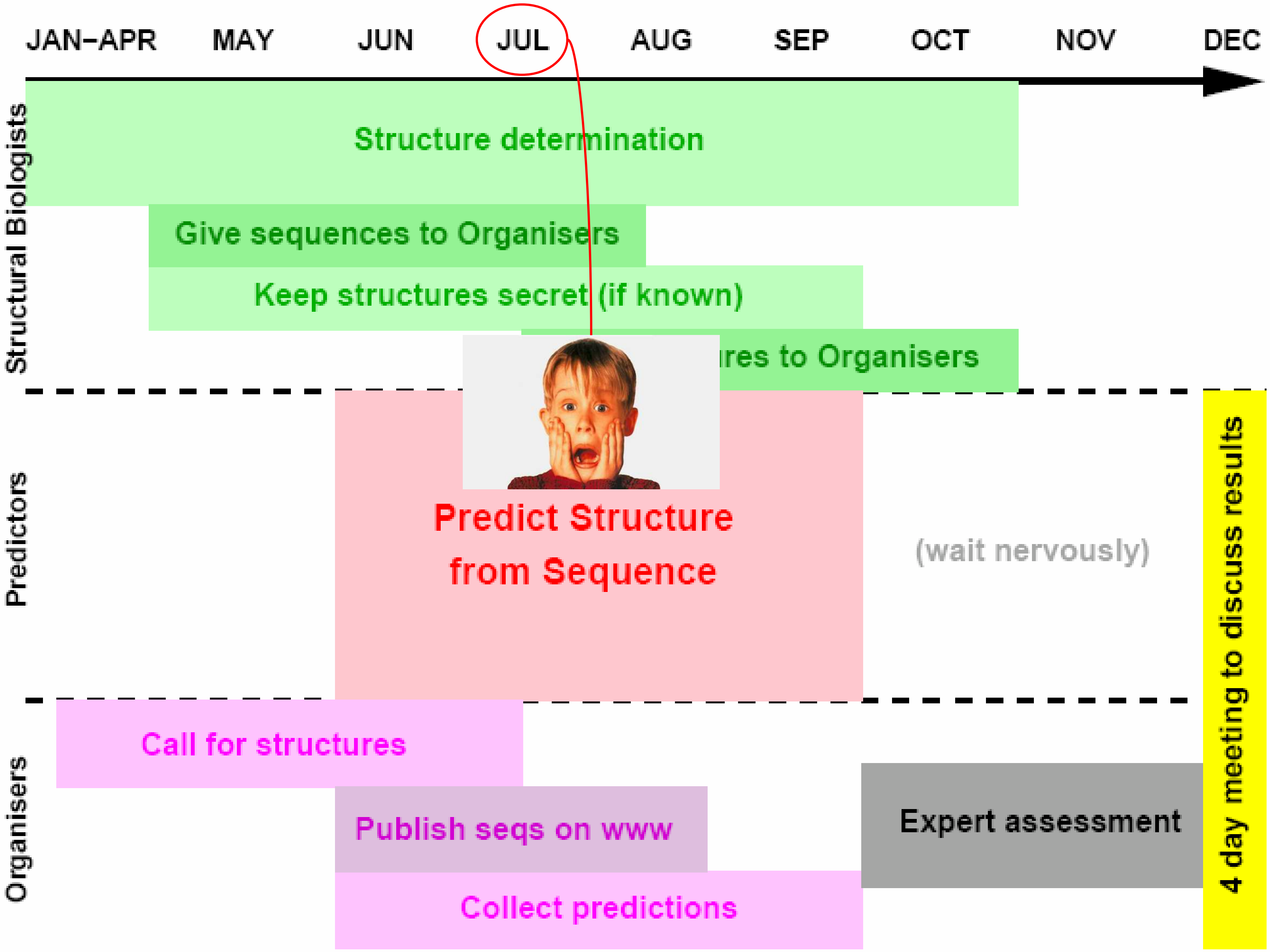
Publish seqs on www

Collect predictions

Expert assessment

4 day meeting to discuss results

CASP8



JAN–APR MAY JUN JUL AUG SEP OCT NOV DEC

Structural Biologists

Structure determination

Give sequences to Organisers

Keep structures secret (if known)

res to Organisers

Predictors

Predict Structure from Sequence

(wait nervously)

Organisers

Call for structures

Publish seqs on www

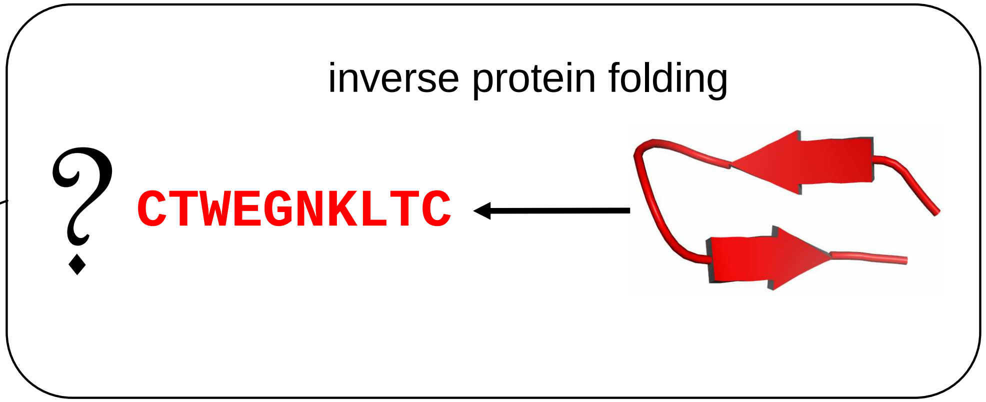
Collect predictions

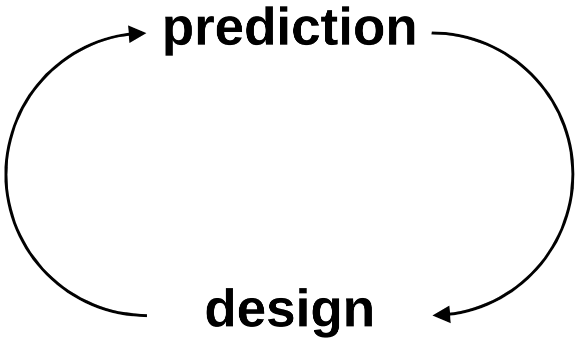
Expert assessment

4 day meeting to discuss results

Functional expansion of Rosetta algorithms

- *ab initio* folding
- design
- docking
- protein-protein interactions
- ligand docking
- enzyme design
- etc.



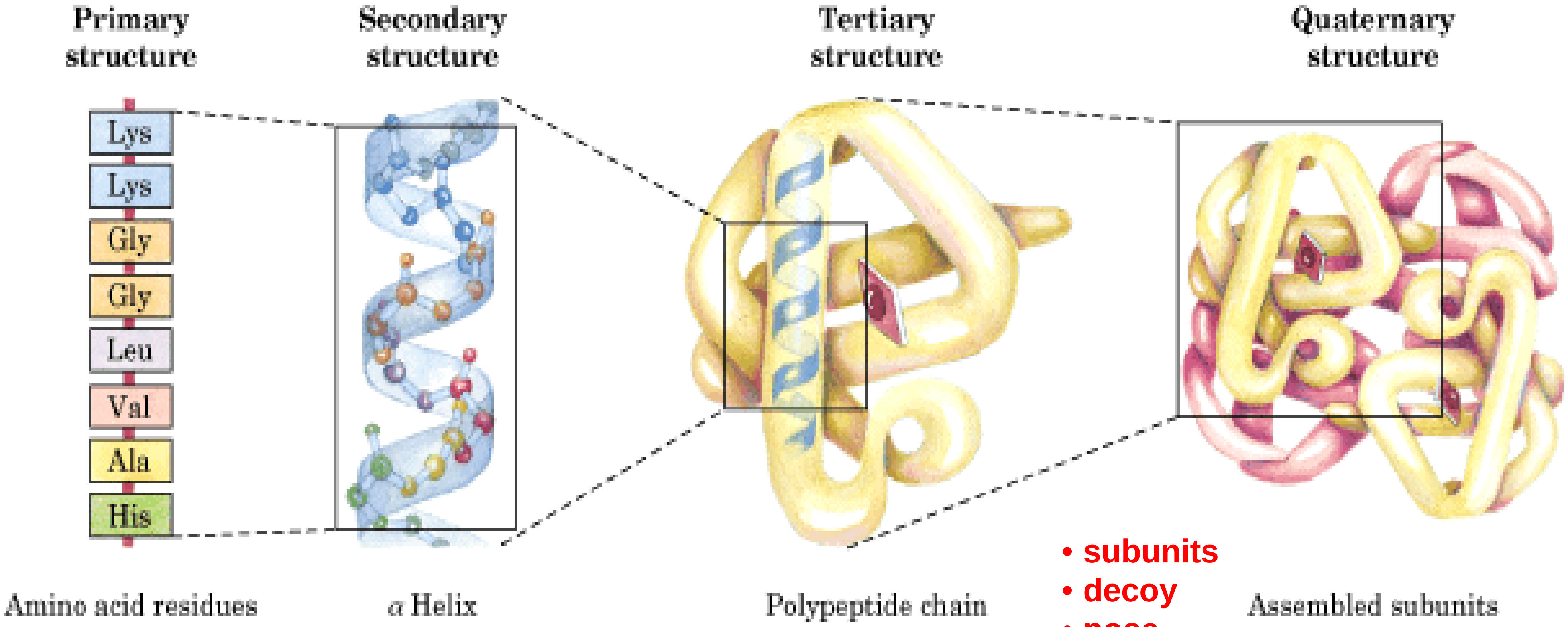


Introduction to Basic Rosetta Methodology

- ➥ States & State Changes
- ➥ Scoring Functions
- ➥ Search & Optimization Routines
- ➥ Output

States Used in **Rosetta**

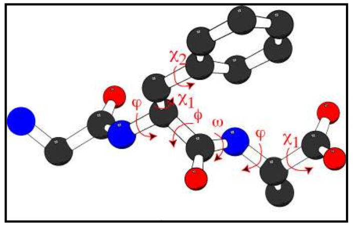
State = Discrete Conformational Unit



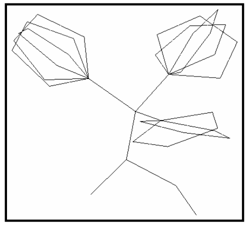
sequence

fragment

- **subunits**
- **decoy**
- **pose**
- **ligand**



dihedral, torsion angle



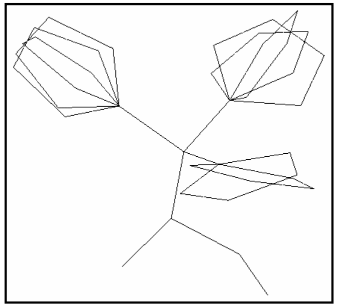
rotamer

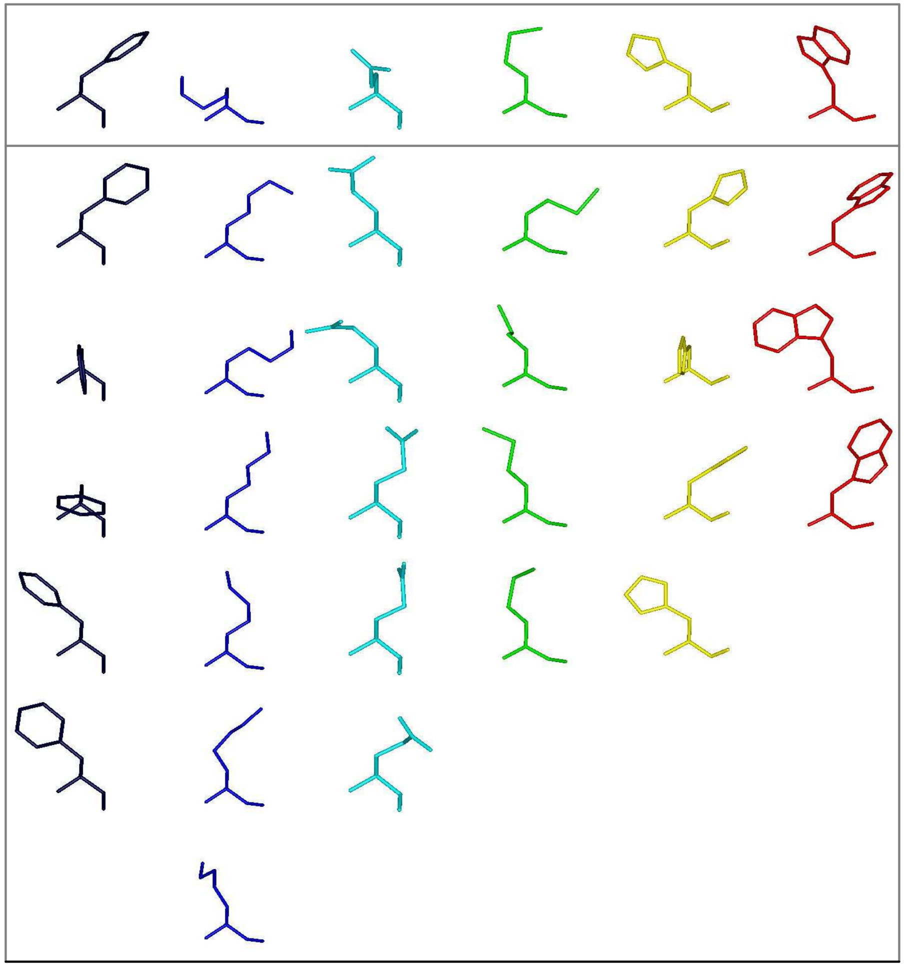
States & State Changes

- *sequences*
  - static state for folding & loop modeling
  - amino acid substitutions in parallel design
- rotamers
- dihedrals
- fragments
- ligands
- protein subunits
- pose & fold trees

Rotamers

States for full-atom scoring and design





rotamer move = substitution

rotamers (rotational isomers):

- highly populated combinations of side-chain dihedral angles.
  - low energy side-chain conformations.
- a small library of about 100-150 rotamers can cover 96-97% of the conformations found in protein structures.

Dunbrack rotamer libraries:

Backbone dependent and independent libraries.

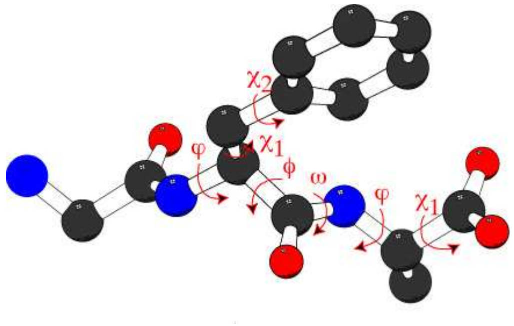
`rosetta_database/bbdep02.May.sortlib`

Dihedrals

States used in most protocols

Small scale dihedral moves (i.e. refinement, minimization)

- Random torsion angle perturbation
  - “small” = randomly perturb paired phi, psi
  - “shear” = randomly perturb phi, equal & opposite perturbation to preceding psi
- *fragment insertion*
- rapid torsion angle optimization to offset global perturbations
  - “wobble” = continuous variation of phi, psi near perturbation to minimize downstream MSD
- gradient descent = dE / dPhi,Psi evaluated, followed by...
  - linmin (line searches):
    - find minimum in direction of steepest descent and stop
    - not the best way to explore a complex landscape
  - dfpmin (Davidson, Fletcher, Pal - quasi-Newton method):
    - the core minimization routine
    - iterations of moves and derivative calculations
    - smarter than steepest descent

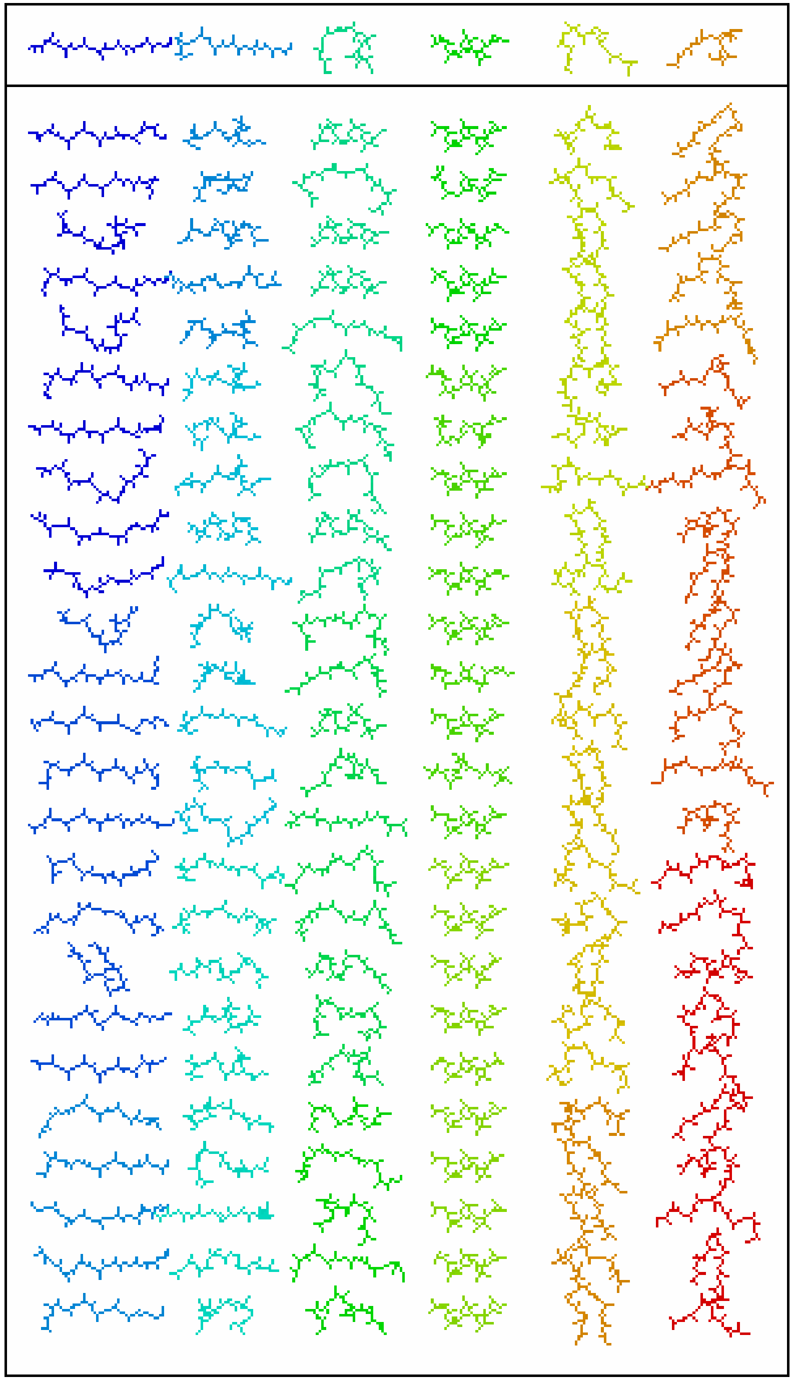


Fragments

- ➥definition
- ➥fragment moves

Fragments

States for ab initio and loop modeling



- 3 and 9 residue fragments
- database created from crystal structures
  - < 2.5Å resolution
  - < 50% sequence identity
- `rosetta_fragments/nnmake_database/vall.dat.2006-05-05`
- *custom fragment database possible*
- low resolution modeling
  - centroid representation of side chains

Making Fragment Libraries

Overview

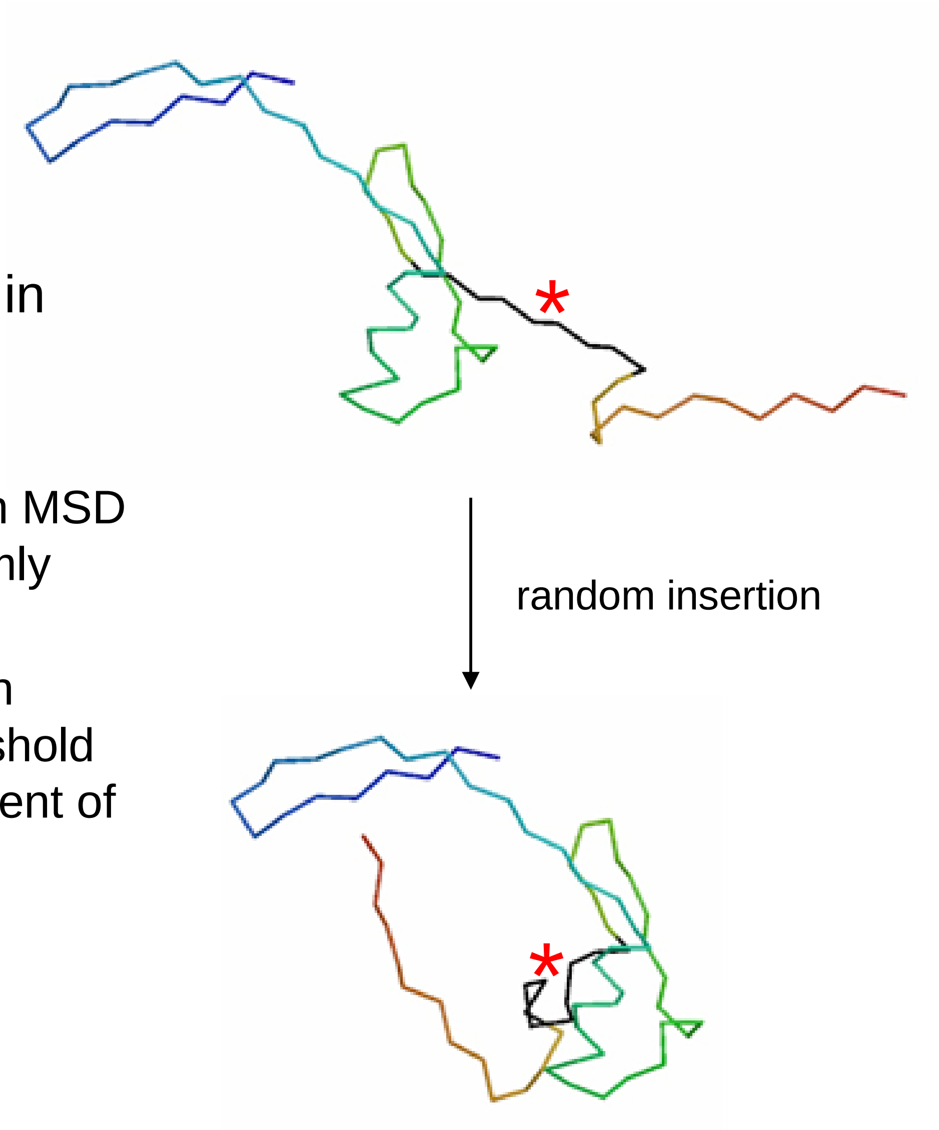
- Fragments are selected from database and ranked according to:
  - input amino acid sequence
    - FASTA format
    - *possible* to use only secondary structure information
  - secondary structure predictions
    - programs
      - PSI-PRED
        - default and predictions carry largest weight
      - JUFO
      - SAM
      - PROF
    - more = better
    - manual

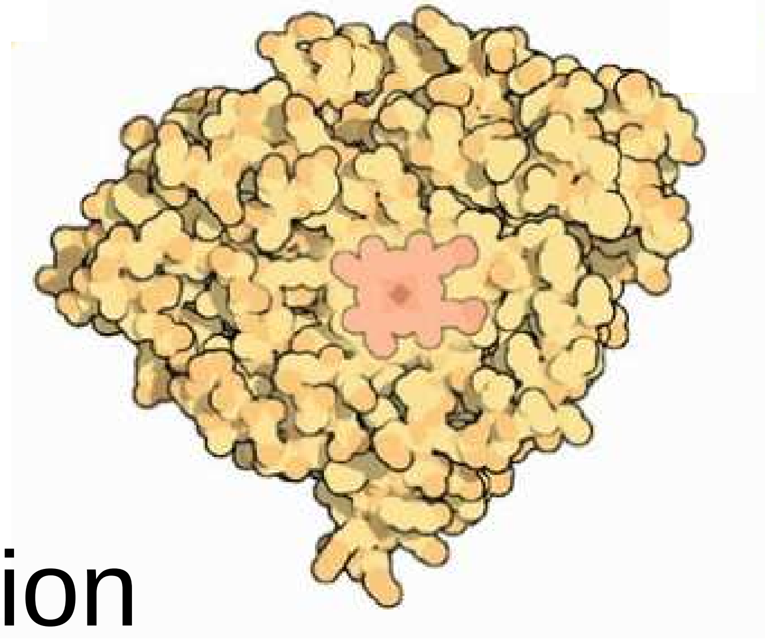
Note: we are leaving “Rosetta”

Fragment Moves

Fragment insertion

- conformation modification occurs in torsion space
- small changes in dihedrals
  - “chuck” = fragments that result in MSD of atoms below threshold randomly inserted (Cartesian)
  - “Gunn” = fragments that result in translation & rotation below threshold are randomly inserted (independent of coordinate system)
  - “crank” = “chuck” + “wobble”



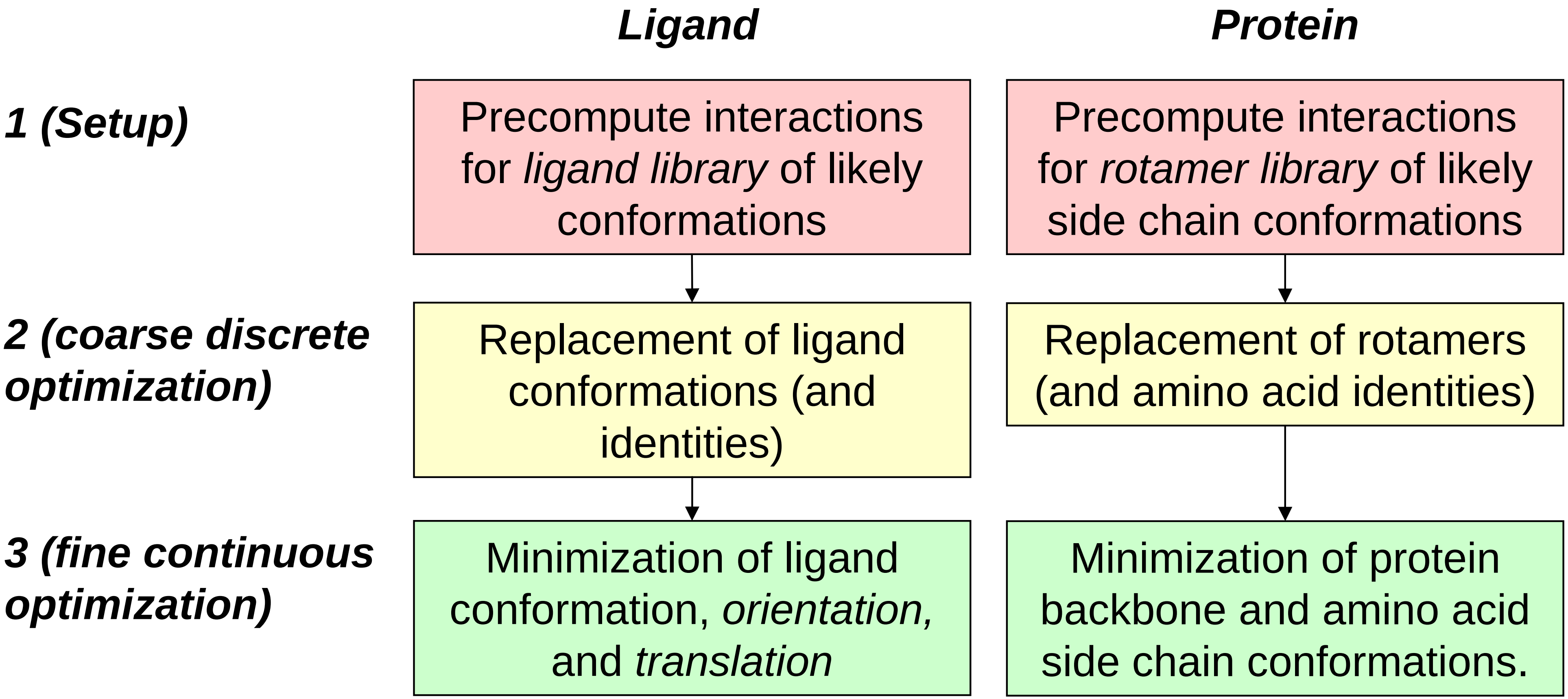


Ligands

- ➥*biochemical* definition
  - ➥metals, small-molecules, etc.
  - ➥(<200 non-hydrogen atoms)
- ➥ligand moves

Ligand Moves

analog of protein design with flexible backbone (& docking)



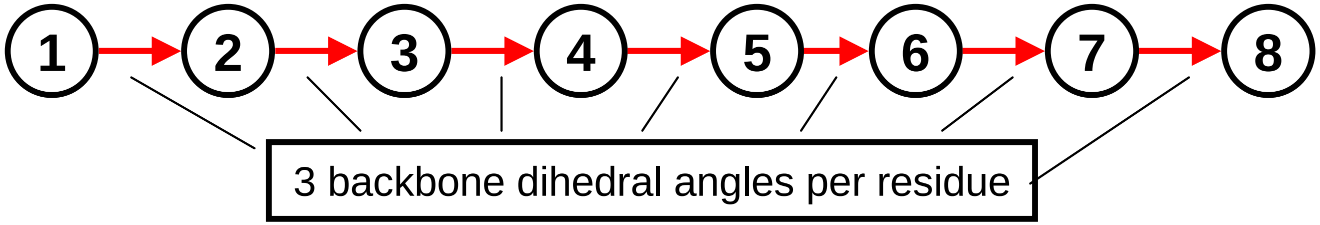
	Ligand	***Protein***
1 (Setup)	Precompute interactions for *ligand library* of likely conformations	Precompute interactions for *rotamer library* of likely side chain conformations
2 (coarse discrete optimization)	Replacement of ligand conformations (and identities)	Replacement of rotamers (and amino acid identities)
3 (fine continuous optimization)	Minimization of ligand conformation, *orientation,* and *translation*	Minimization of protein backbone and amino acid side chain conformations.

slide content credits:
Jens Meiler

Pose & Fold Trees

Methodological Inconvenience

Rosetta folding

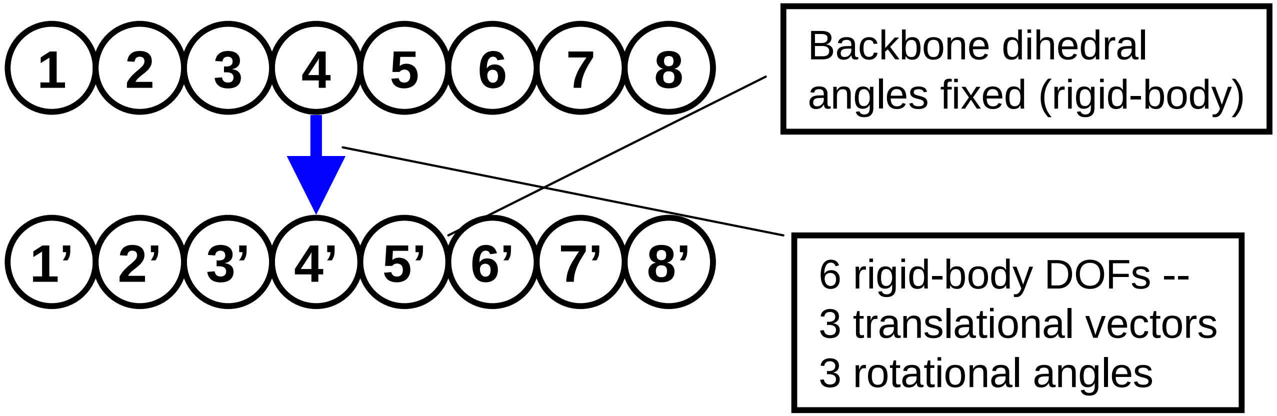


3 backbone dihedral angles per residue

Sampling and minimization in TORSIONAL space

Sampling and minimization in RIGID-BODY space

Rosetta docking



Backbone dihedral angles fixed (rigid-body)

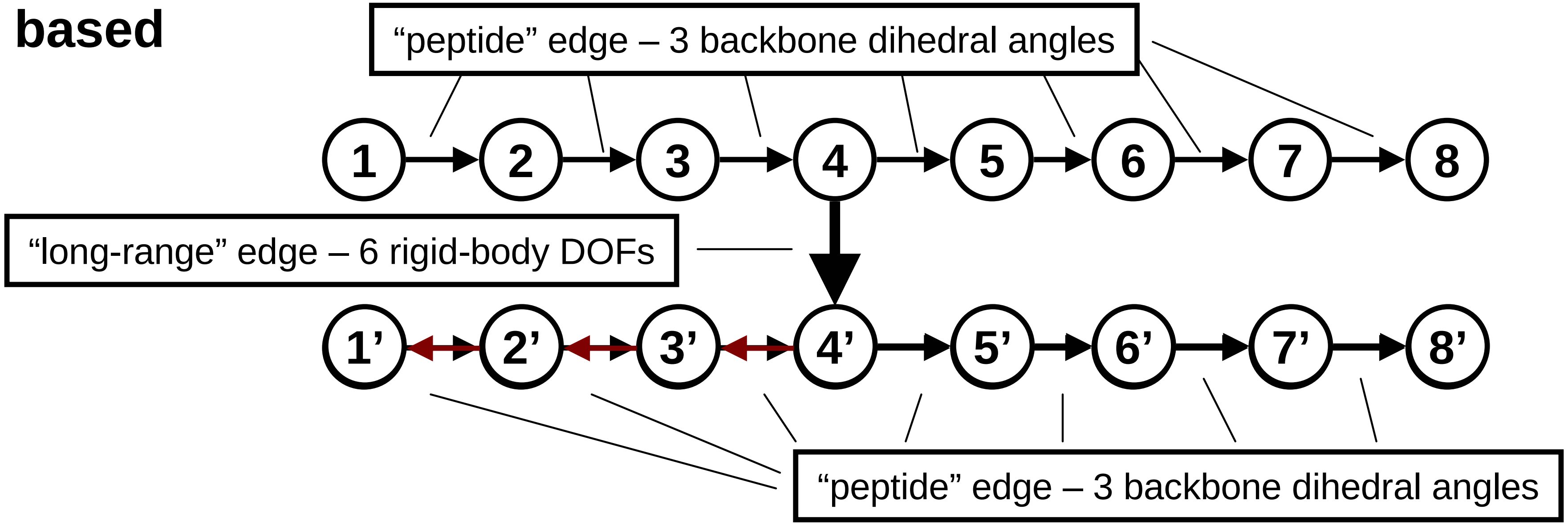
6 rigid-body DOFs --
3 translational vectors
3 rotational angles

Pose & Fold Trees

Fold tree representation

Allows simultaneous optimization of rigid-body and backbone/sidechain torsional degrees of freedom.

fold-tree based docking


“peptide” edge – 3 backbone dihedral angles
1
2
3
4
5
6
7
8
“long-range” edge – 6 rigid-body DOFs
1’
2’
3’
4’
5’
6’
7’
8’
“peptide” edge – 3 backbone dihedral angles

- Construct fold-trees to treat a variety of protein folding and docking problems.

Bradley and Baker, *Proteins* 2006

Energy Functions

- purpose: *score states*
- major classes
  - low resolution
  - high resolution

Major Classes of Energy Functions

- **Low resolution:** *reduced atom representation*
  - simplified energy function
  - used for aggressive search of state space
- **High resolution:** *full-atom representation*
  - detailed energy function
  - local search of state space
  - refinement and minimization

Rosetta Energy Function

Low resolution:

Atom Model

centroid reduction of side chains

Energy function terms

van der Waals repulsion

“pair” terms (electrostatics)

residue environment (prob of burial)

2^o structure pairing terms (H-bonds)

radius of gyration

packing density

In general …

Weighted linear combination

$$\mathit{Energy} = w_1{*}\mathit{term}_1 + w_2{*}\mathit{term}_2 + \ldots$$

Pair-wise decomposable

Heavily trained on PDB statistics

Discriminate “near native” vs “non native”

No single low resolution score

Several functions with different weights

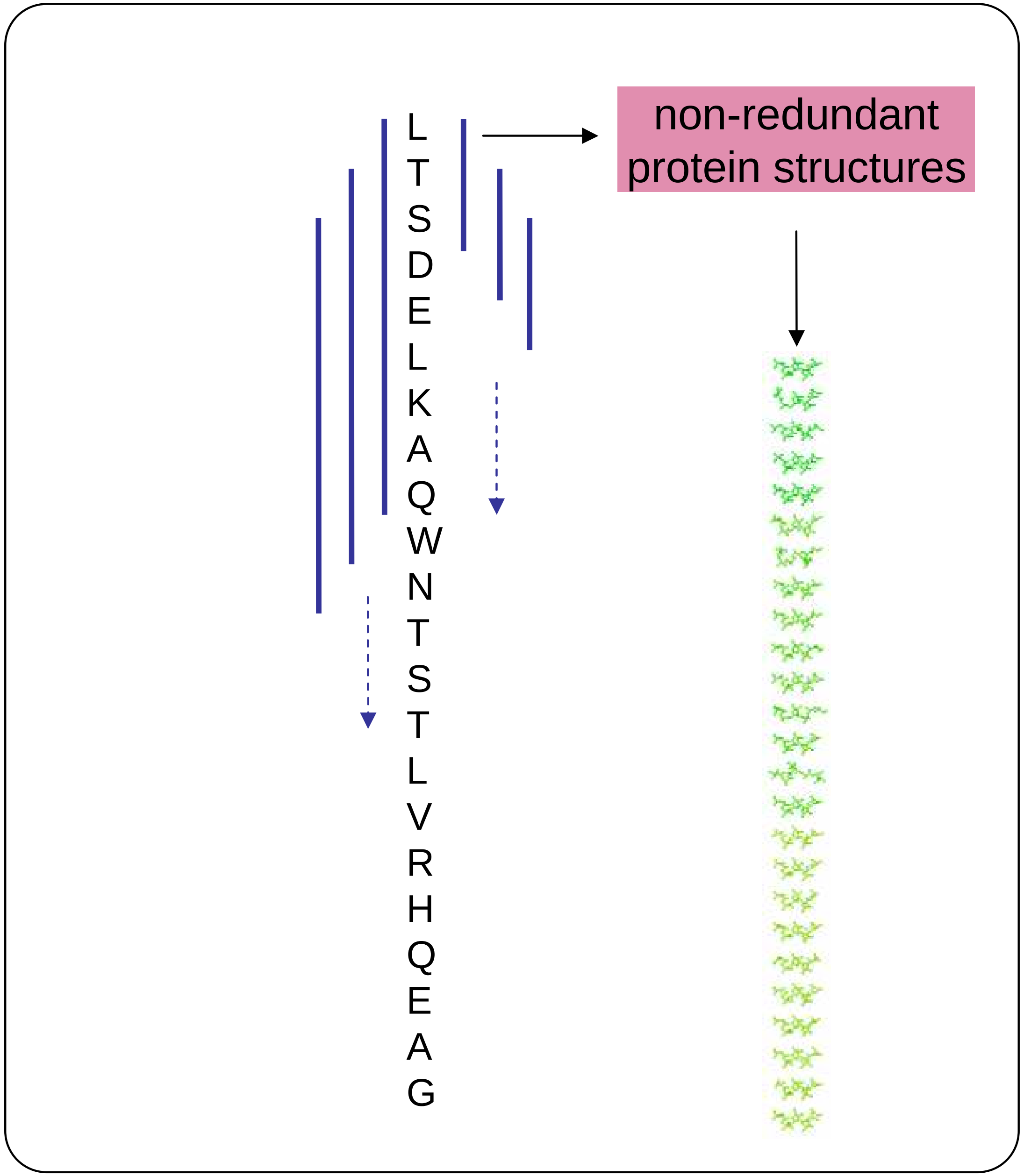
slide content credits:
Glenn Butterfoss

Rosetta Energy Function

Low resolution:

Implicit terms

fragments (local interactions)



non-redundant protein structures

slide content credits:
Glenn Butterfoss

Rosetta Energy Function

High resolution:

Atom Model

- full atom representation

Energy function terms

- Rotamer (Dunbrack)
- Ramachandran
- Solvation (Lazaridius Karplus)
- Hydrogen bonding
- Lennard-Jones
- Pair (electrostatic)
- Reference energies

In general …

Weighted linear combination

$Energy = w_1 * term_1 + w_2 * term_2 + \ldots$

Pair-wise decomposable

- Pre- tabulate energies

Hybrid Statistical / MM-like score

Weights trained for different applications

slide content credits:
Glenn Butterfoss

Search and Optimization

- size of state spaces
- algorithm(s)
  - Monte Carlo
  - simulated annealing
  - Metropolis

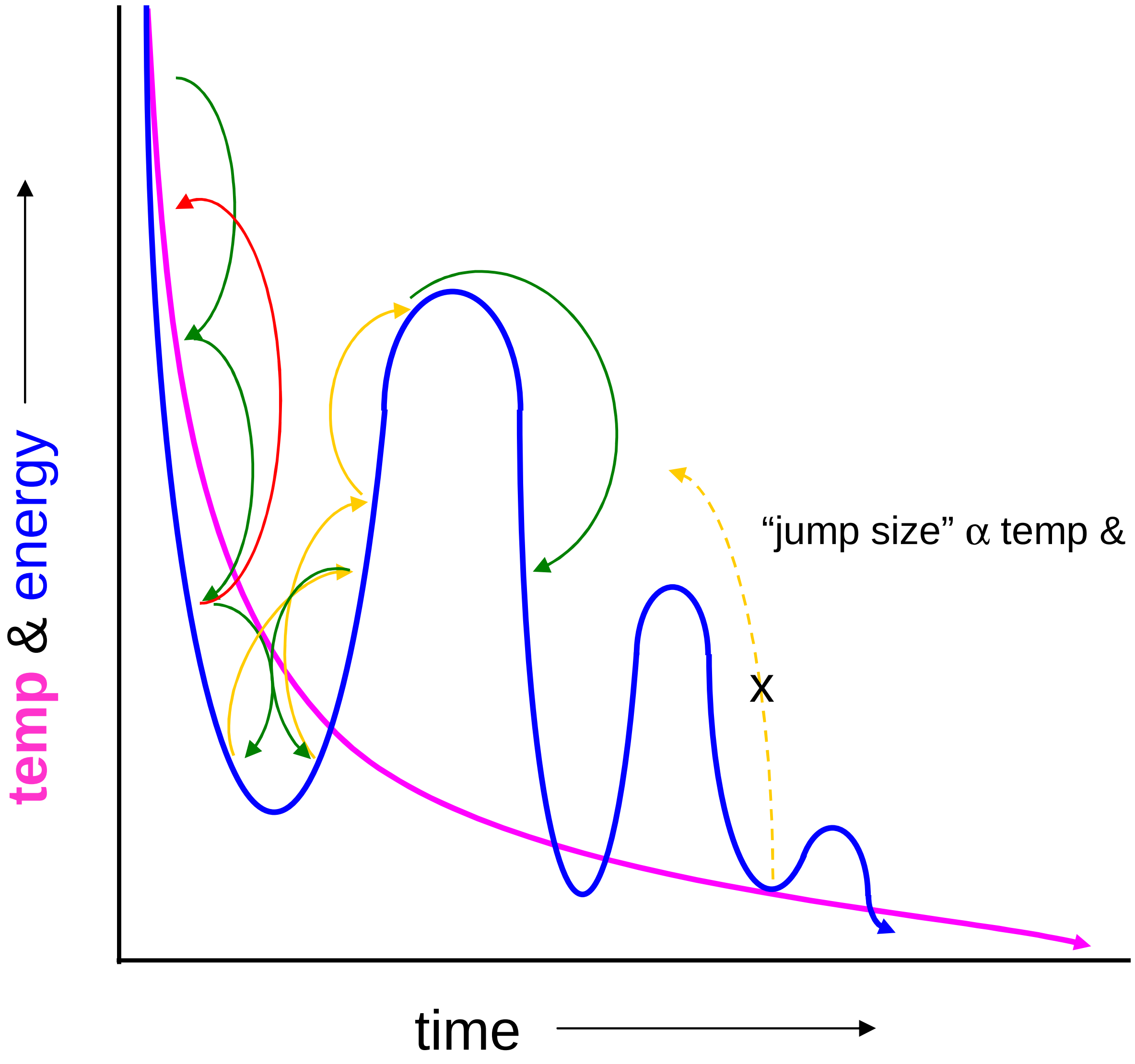
Approximate size of different state spaces

- **Folding**: given either alpha, beta, or loop conformation, for protein of *nres*, 3^{nres} possible conformations.
  - Levinthal paradox (*Cyrus Levinthal, J. Chim. Phys. 65, 44; 1968*):
    - If *nres* = 100, sampling a conformation every 10^{-13} seconds, it would take 10^{27} years to fold. Universe is 10^{10} years old.
  - Folding is non-random and cooperative.
- **Design**:
  - for protein of *nres*, 20^{nres} possible sequences
  - given 10 rotamers per fixed amino acid, 10^{nres} possible states
- **Docking**: 360^3 x Angstroms^3 (for 10 Angstroms, 4.6 x 10^{10} states)
- etc.

Basic Rosetta optimization algorithm

Monte Carlo search = *random state substitutions*

Simulated Annealing & Metropolis = *acceptance criterion*



Rosetta methodology in real time

NOTE: **MOVIES REPRESENT SINGLE TRAJECTORIES**
typical simulation involves 100-100000 trajectories

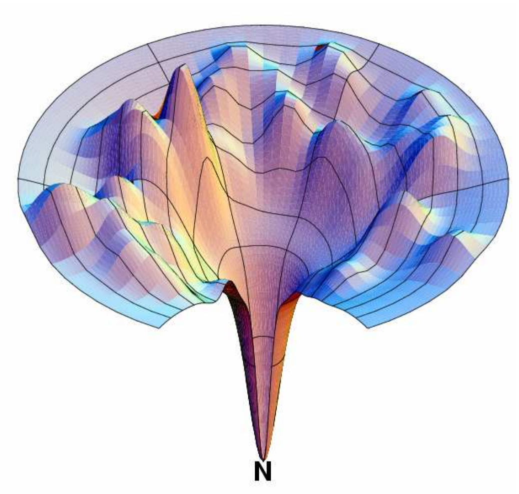
- ➥ design movie
- ➥ *ab initio* movie
- ➥ docking movie

Overview of Rosetta output

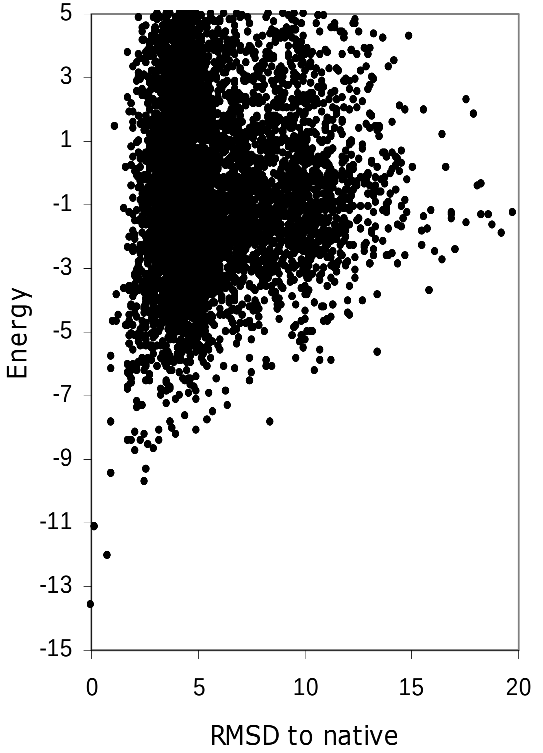
- ↪decoys and funnels
- ↪computational power versus accuracy
- ↪constraints
- ↪filters

Funnels: decoy RMSD to native versus energy
1 decoy/point = 1 trajectory

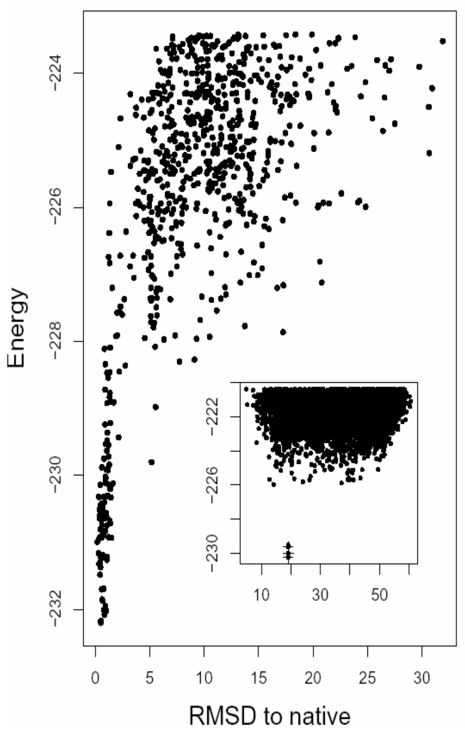
Energy landscape



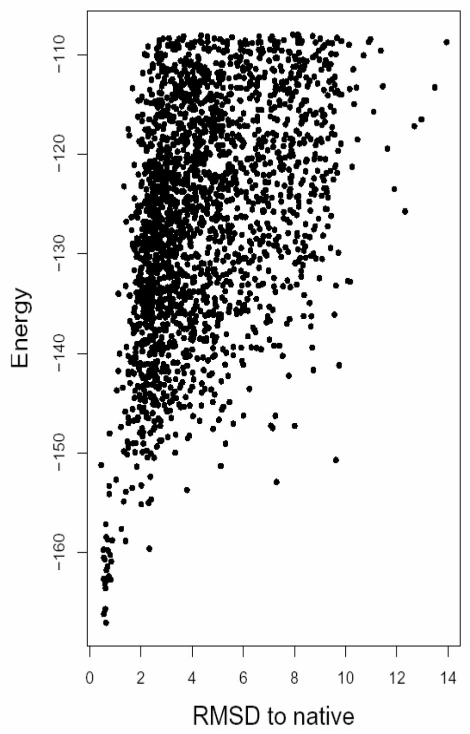
Ligand-protein energy landscape



Docking energy landscape



Folding energy landscape



Similar energy landscapes for Rosetta predictions:

- *energy function accurately scores states*
- *models can be selected by energy/score only*

slide content credits:
Ora Furman-Schueler
Ken Dill
Phil Bradley
Kristian Kaufmann

Constraint: *user input limitation of state space search*

- constraint methodology
  - violation of a constraint increases the decoy score
  - Implemented through files (.cst, .dpl, .dst)
- types of constraints
  - mainly apply to *ab initio* mode
  - NMR derived dipolar coupling constraints
  - barcode constraints (features like ss, phi/psi, etc.)
  - distance constraints (docking)
- future expansion to other modes

Filters: *absolute constraints*

- filter methodology
  - violation causes decoy to be discarded
  - implemented through command line options
- physical attributes
  - disulfides
  - knot
  - SASA
  - vdw
  - radius of gyration
  - score
  - etc.

Overview of Rosetta Implementation

- Implementation Details of Select Modes
- Brief Description of Select Modes
  - Loop Modeling Protocols
- Introduction to the Rosetta command line
- Flow-chart of Rosetta Execution

Brief Description of Select Modes

mode	description	main flag(s)	main code
ab initio	predict the structure from sequence	*none (original mode)* `-abrelax`	fold_abinitio.cc
relax	refine the structure using Rosetta energy functions	`-relax`	relax_structure.cc
idealize	replace bond geometries with ideal values	`-idealize`	idealize.cc
loop modeling	build and refine local structurally variable regions in context of a structural template	`-loops`	fold_loops.cc
design	optimize sequence given a structure	`-design`	design_structure.cc
docking	structure prediction for a protein-protein complex given subunits	`-dock`	dock_structure.c docking.cc
ligand	ligand docking, design	`-ligand`	ligand.cc
interface	ddG calculation for mutations made across a complex interface	`-interface`	analyze_interface_ddg.cc
scoring	score input conformations with Rosetta energy functions	`-score`	scorefxns.cc
domain assembly	fixed domains connected by variable regions	`-assemble`	assemble_domains.cc
pose	a set of algorithms which improve previous implementations	`-pose` `-pose_*`	pose_*.cc

Brief Description of Select Modes

mode	de		
ab initio	p		
relax			
idealize			
loop modeling			
design			
docking	c		
ligand	li		
interface	d co		
scoring	score input conformations with Rosetta energy functions	`-score`	scorefxns.cc
domain assembly	fixed domains connected by variable regions	`-assemble`	assemble_domains.cc
pose	a set of algorithms which improve previous implementations	`-pose` `-pose_*`	pose_*.cc

Loop modeling protocols

Protocol	Reference	General characteristics	Differing input files

“Classical”	Carol Rohl et al.
	Proteins 2004.
		classical *ab initio* fragment insertion with minimization	`(1pdbC.ssa)`
			- secondary structure assignments
			`1pdb.loops`
			- loop library

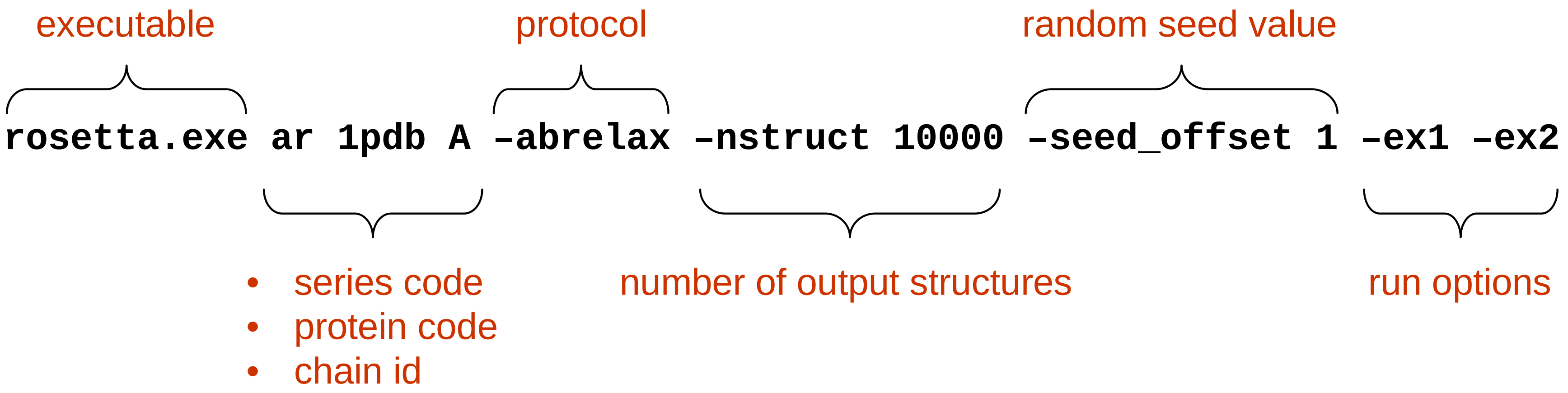
“Pose-based”	Chu Wang et al. *JMB* 2007	+ explicit cyclic coordinate descent for loop closure	`1pdbC.pose_loops` - loop definitions and options
“Loop relax”	Bin Qian et al. *Nature* 2007	+ full atom minimization	`1pdbC.loopfile` - loop definitions
“Termini”	Sood et al. *JMB* 2006	centroid based extension of protein termini	`1pdbC.loops` - special loop library
“Loop design”	Xiaozhen Hu et al. *PNAS* 2007	specialized flexible backbone design	(custom method and inputs, stay tuned...)

Introduction to the Rosetta command line

UNIX-like:

`executable –flags`

e.g. `ls -a`



executable

protocol

random seed value

```
rosetta.exe ar 1pdb A –abrelax –nstruct 10000 –seed_offset 1 –ex1 –ex2
```

- series code
- protein code
- chain id

number of output structures

run options