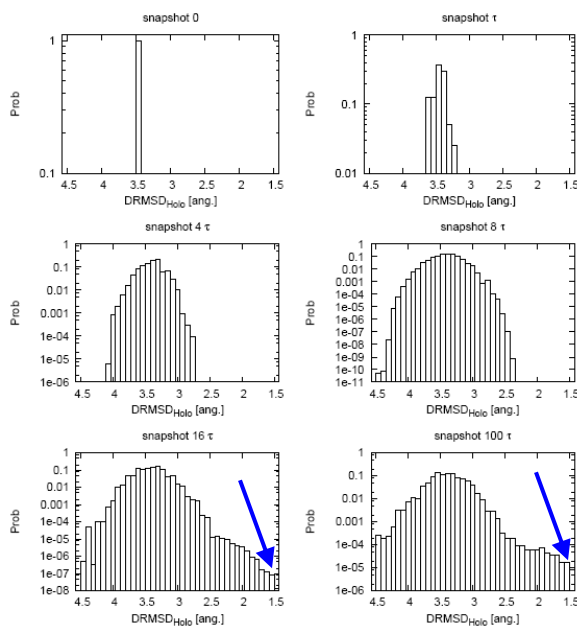


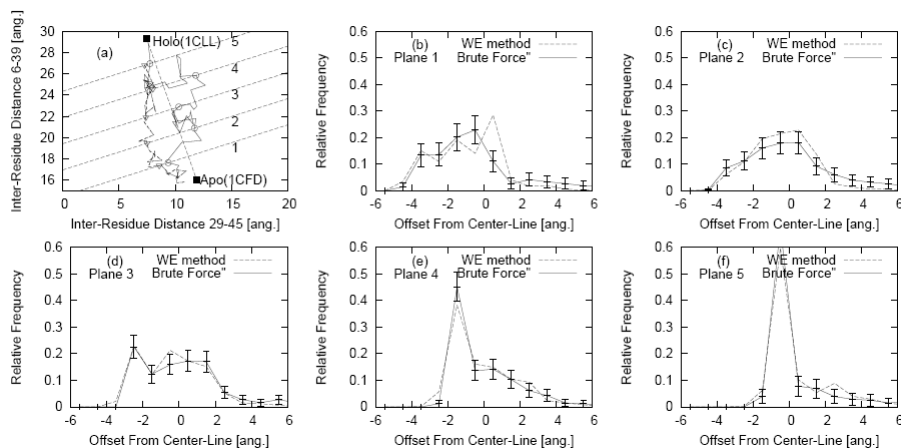
# Calmodulin: WE works as expected

- Initialized path ensemble
- Efficient; more so with higher barrier
- Details
  - 40 bins, 40 traj per bin
  - Fixed bins: DRMSD from target config
  - Target defined as **DRMSD < 1.5 Å**

[Zhang, Jasnow, Zuckerman, PNAS, 2007]



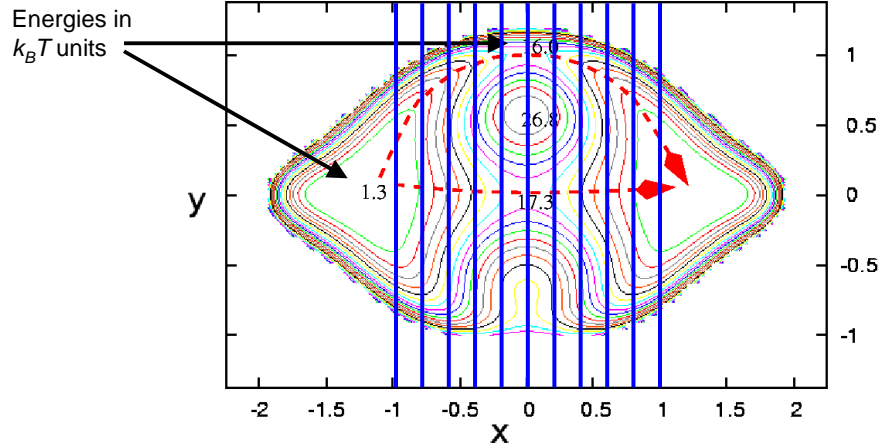
# Calmodulin: structurally correct path ensemble



- Rates also calculated correctly

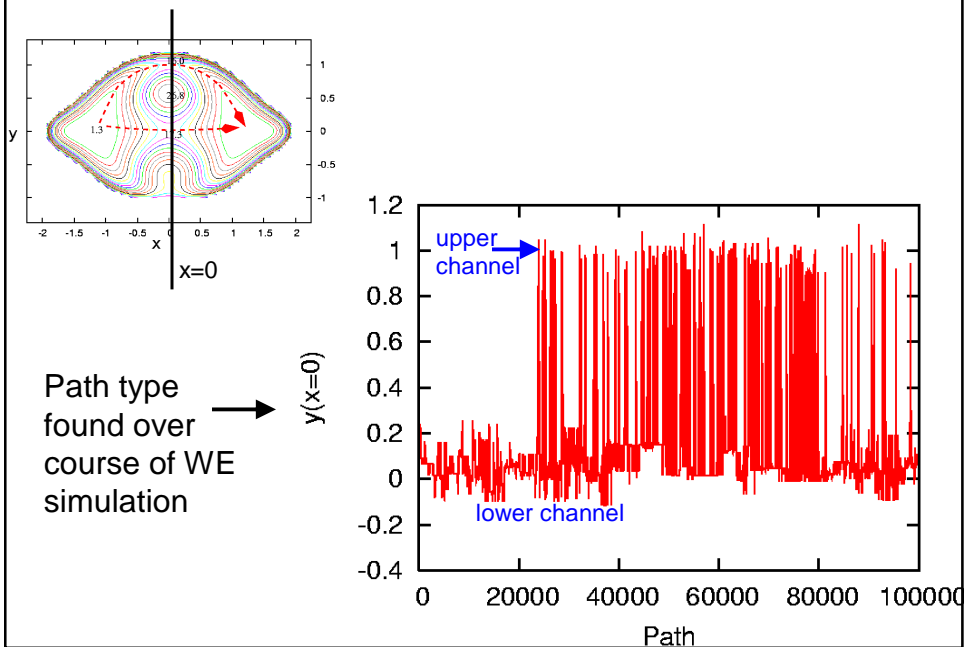
$$k_{WE} = (1.9 \pm 0.4) \cdot 10^{-10} \text{ per MC step} \sim k_{BF} = (2.1 \pm 0.2) \cdot 10^{-10} \text{ per MC step}$$

## Basic WE for 2D/two-channel problem



- Two-dimensional toy model
- A  $10 k_B T$  “mountain” separates the two channels
- Fixed bins as shown by vertical lines

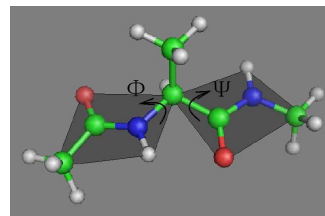
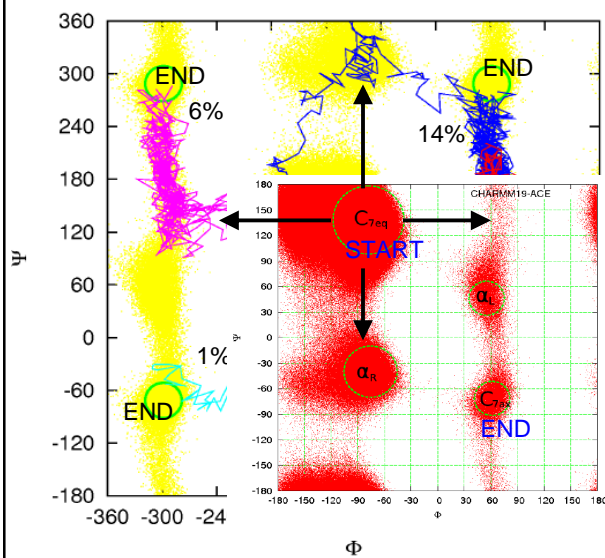
## Basic WE in 2D: Both channels are found



## Basic WE: Finding a target state

- WE can be run with fixed bins and unknown target state ...

## Alanine dipeptide: multiple paths, unknown target

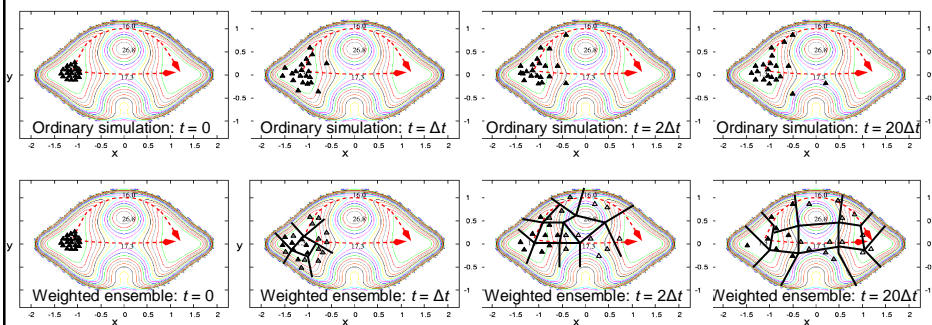


rational  
channel use

# Adaptive WE

- Bins are arbitrary and can be changed
- Idea: Change bins to “follow” evolving probability distribution

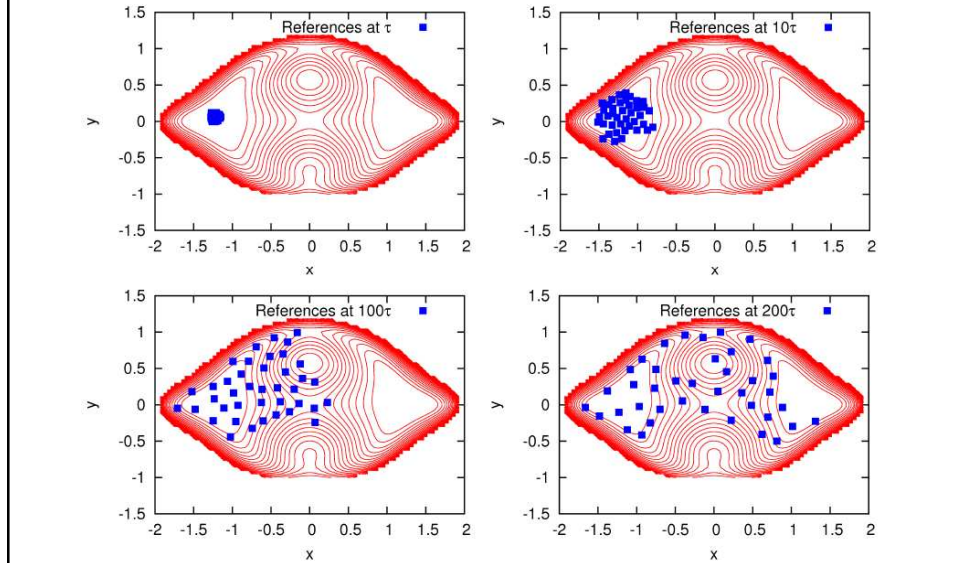
## Schematic adaptive example



- Adaptive procedure:
  - Run dynamics
  - Generate Voronoi cells
  - Split and combine – usual WE
  - Repeat

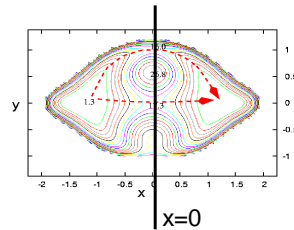
# Adaptive WE in two dimensions

- Reference configs for Voronoi cells, evolving

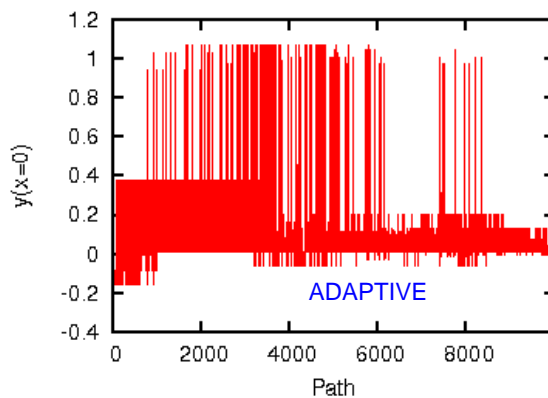
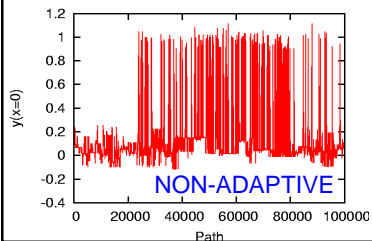


# Adaptive Voronoi: Results in 2D

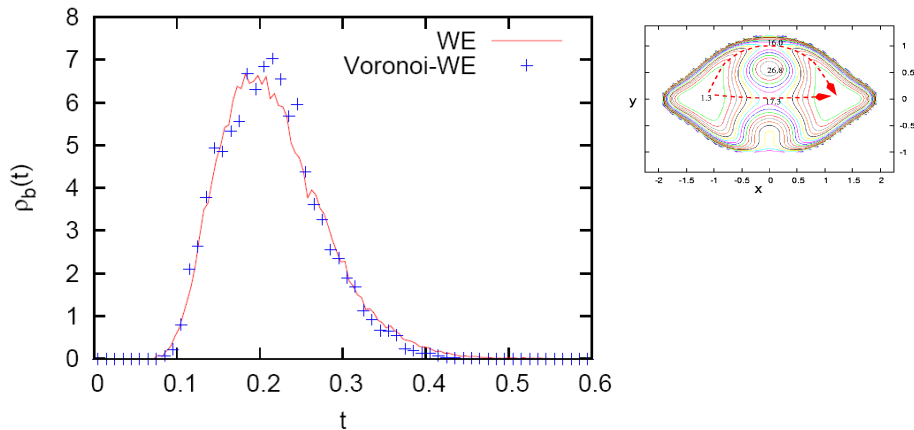
- Only single start-state used
- Redefine bins after each dynamics period
  - Keep them spread out using Voronoi procedure



Second channel found in **less than 1000 paths**, compared to  $> 20,000$  in simple (non-adaptive) WE



## Adaptive WE gives correct distribution



- Must have right balance of two channels to get true distribution

## Summary: WE for path sampling

- Rigorous: exact for any stochastic dynamics or thermostat
- Initialized or steady-state trajectory ensembles
- Single WE simulation provides path ensemble & rate
- Ideal for multiple channels
- Fixed or adaptive bins
- Easy to code/script/parallelize
  - Our C code is available to download

## Part II: Modeling

- Reminder: the high cost of sampling ...
  - To obtain 100 independent trajectories requires

$$\text{Computation time} : 100 \times t^{\text{traj}} \times m$$

$t^{\text{traj}}$  = avg trajectory duration

$m$  = inefficiency multiplier

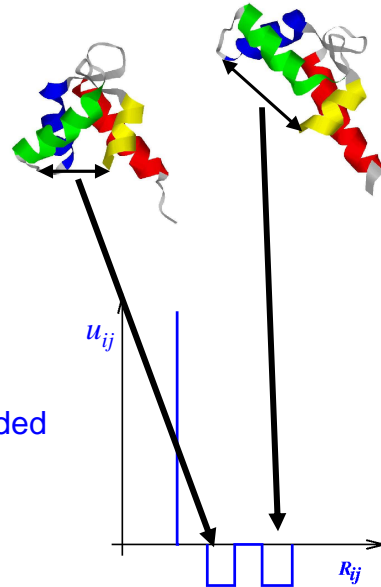
Depends  
on model:  
**MUST  
COARSE-  
GRAIN**

## Part II: Modeling

- What would be the perfect model for path sampling?
- Hamiltonian/forcefield: As complex as possible (but coarse-graining will be necessary)
- Programming: Flexible software, permitting global and local changes to resolution, as required by the system/questions and resources

## Coarse-grained models must stabilize states

- Coarse-grained (CG) model must be roughly correct
  - Protein geometry
  - Basic interactions
  - Key states must be stable
- No universal CG model
  - Nature uses 20 atomistic amino acids
- Double-Go guarantees bi-stability
  - Realistic interactions can be added and Go reduced
  - Used in many groups
  - [Zuckerman, *J Phys Chem B*, 2004]

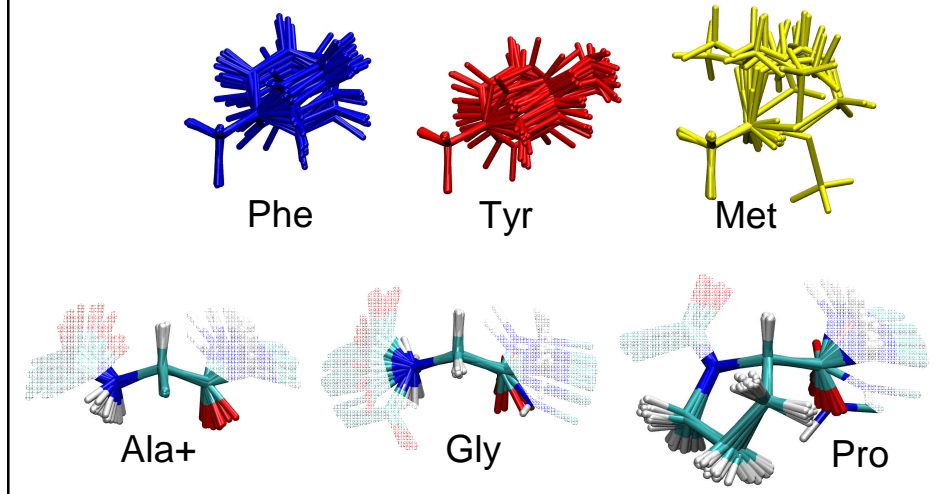


## Model Implementation

- Many ways to do it
- Ideally: adjustable resolution (globally)
  - Suit to changing resources
- Ideally: adjustable resolution (locally)
  - Suit to specific question: atomistic binding site
  - “Mixed model”

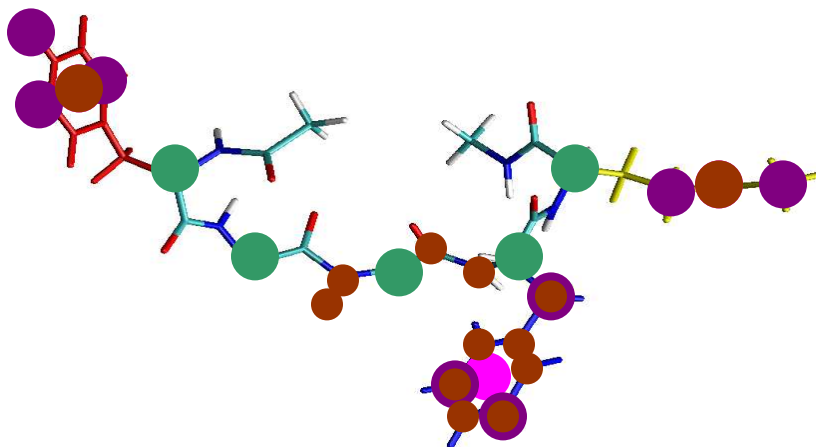
## Implementation: RAM is your friend

- 4 GB RAM = 1 billion floats = \$50
- Therefore, pre-calculate ...



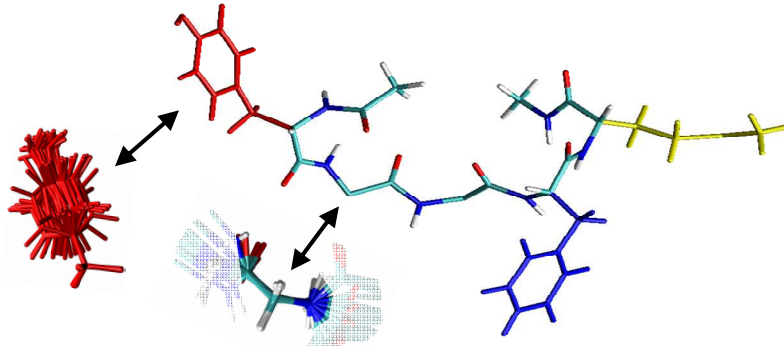
## Put the pieces together

- Store all atomic configurations
- Use a subset of sites for interactions



## Library-based Monte Carlo (LBMC)

- Trial move = swap one or more configurations with library/ies
- Energy internal to configuration cancels in acceptance criterion
- Neighbor lists can be used to increase acceptance and cause local, physically reasonable changes
- Used with any model: all-atom, coarse-grained, or mixed



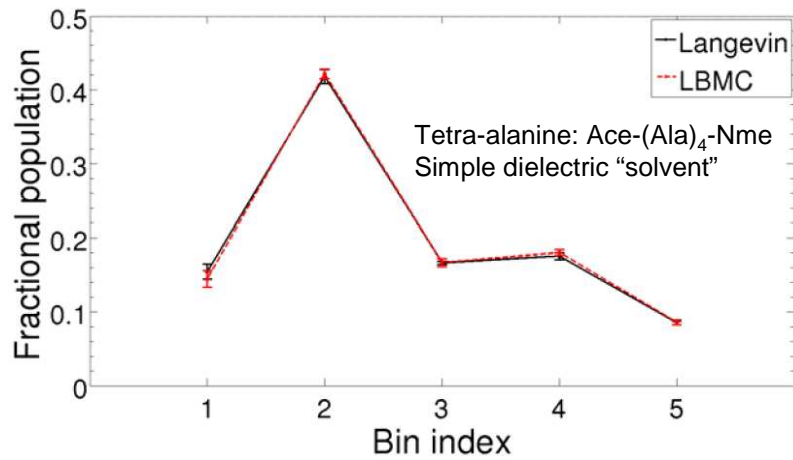
[Mamonov, Bhatt, Cashman, Ding, & Zuckerman, submitted; see also arxiv.org  
Libraries available on website]

## LBMC strengths

- LBMC permits fully flexible use of library-based models
  - Arbitrary/mixed resolution
- Some savings in energy calculation, but ...
  - Large system: most interactions not within (small) fragments
- However, LBMC is much more efficient than internal-coordinate MC or Langevin
  - Reason: **complex correlations stored in libraries**
  - Tested in all-atom peptides; OPLSAA forcefield (100 – 1,000 times faster than Langevin or int-coord MC)

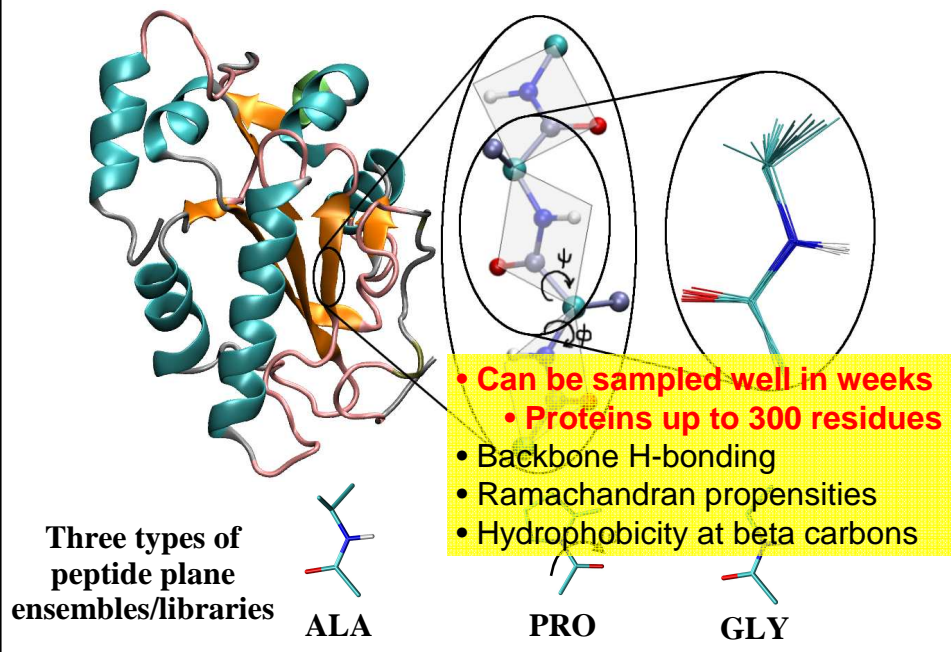
## LBMC: Efficiency in all-atom peptides

- LBMC: 10 sec
- Langevin: 16 hrs

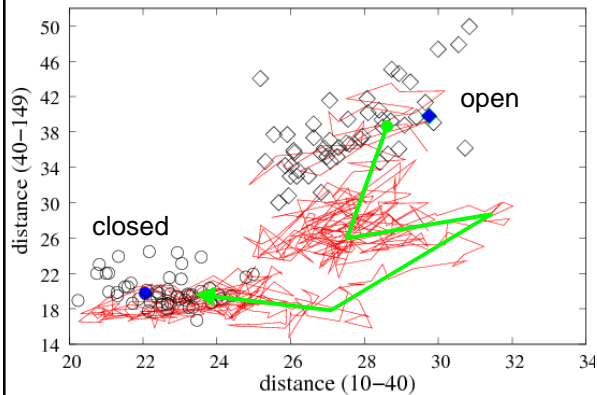


[Sample-size assessment: Lyman & Zuckerman, *J Phys Chem B*, 2007]

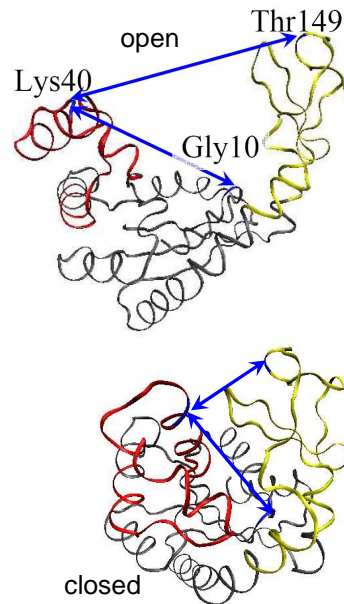
## Atomistic backbone at CG cost via libraries



## Basic WE for adenylate kinase

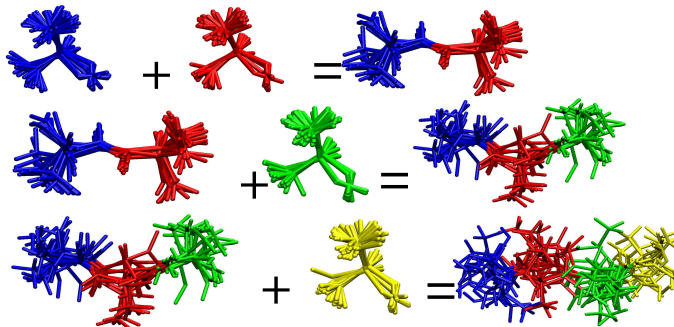


- Atomistic backbone + alpha-carbon interactions
- Open symbols = equilibrium ensembles
- Transitions start in a day of single-CPU simulation
- Steady state reached in **less than a week** (basic WE)



## Other uses for atomistic fragment libraries

- Multi-resolution simulation of proteins
- Polymer-growth sampling of all-atom peptides, small molecules
- Library-based MC sampling of all-atom molecules
- Absolute free energy (partition function) calculations via growth
  - Can grow ligand in flexible receptor
  - Receptor modeled via LBMC with atomistic binding site



## Overall summary

- Theory: Different path ensembles to sample
- Simulation: “Weighted ensemble” strategy is very general & flexible
  - Key for future: Better algorithms – adaptive and steady-state weighted ensemble simulation
- Modeling: Atomistic fragment libraries → models as complex as possible
  - Libraries available
  - Library-based Monte Carlo for atomistic or CG models

## Acknowledgments

- Bin Zhang (student/postdoc) – almost everything
  - Looking for postdoctoral position
- Divesh Bhatt (postdoc) – WE on adenylate kinase
- Ying Ding (student) – LBMC on all-atom peptides
- David Jasnow (collaborator) – keeps me honest
- Funding from NIH, NSF