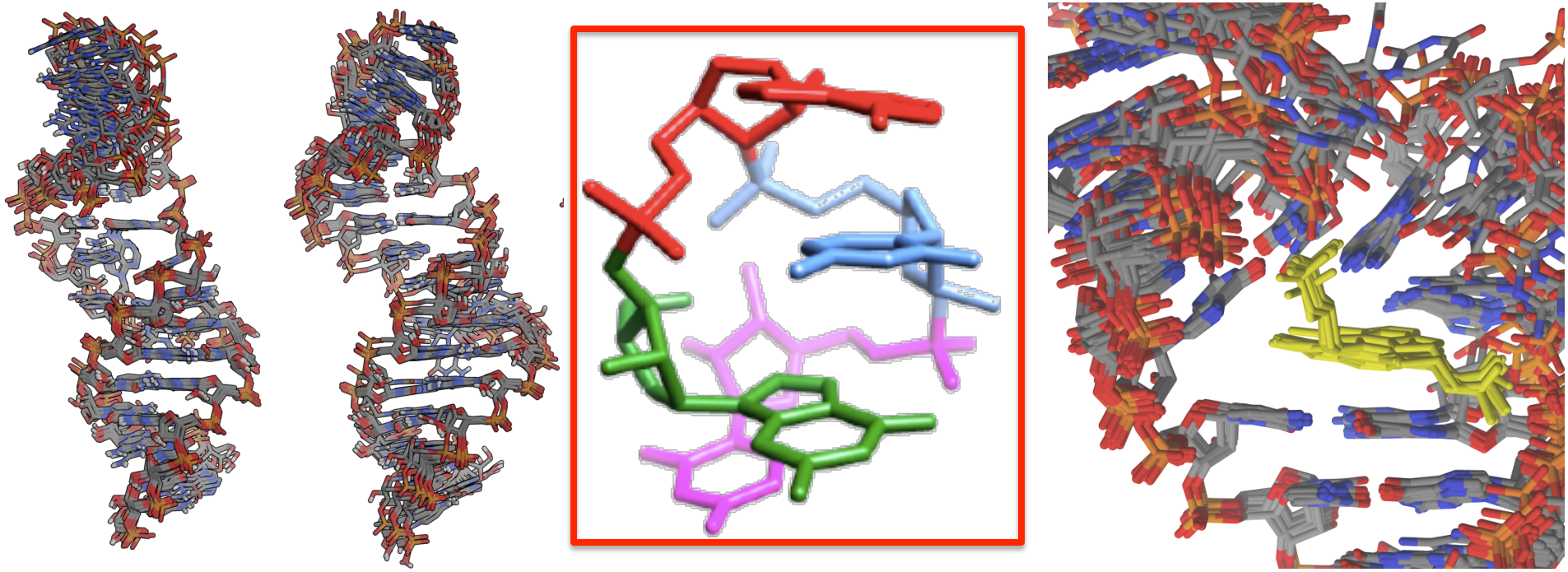
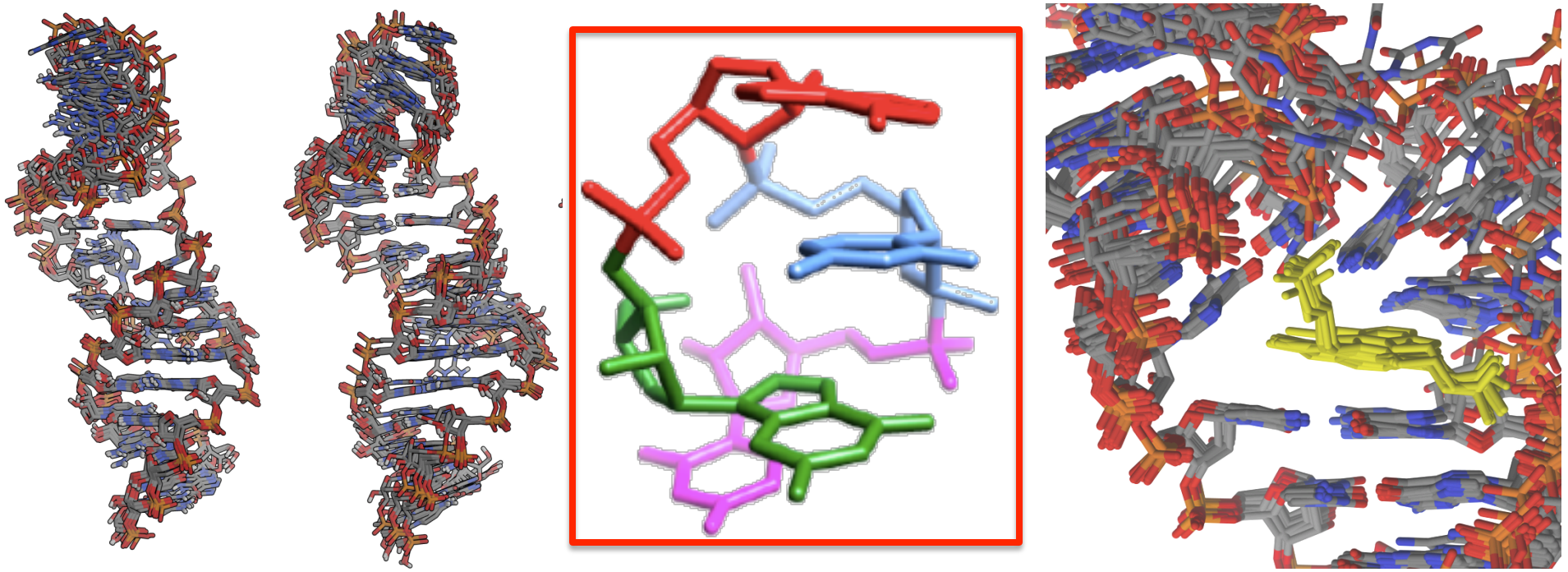
NSF OCI-1036208: PRAC – Hierarchical molecular dynamics sampling for assessing pathways and free energies of RNA catalysis, ligand binding, and conformational change”. NEIS-P2 update, May 2013



Thomas E. Cheatham III
Associate Professor
Department of Medicinal Chemistry
College of Pharmacy, University of Utah

Advances in computational power over the past two decades have transformed our understanding of biomolecular structure…



…we bring together an experienced team of **AMBER developer's** with expertise ranging from QM/MM methods to understanding of biomolecular structure to try to decipher the full landscape of RNA structure and function.

PI: Cheatham

Co-PIs: Carlos Simmerling (Stony Brook U), Adrian Roitberg (U Florida), Darrin York (Rutgers) and Ross Walker (UCSD).

AMBER leader: David Case (Rutgers)

BLUE WATERS
SUSTAINED PETASCALE COMPUTING

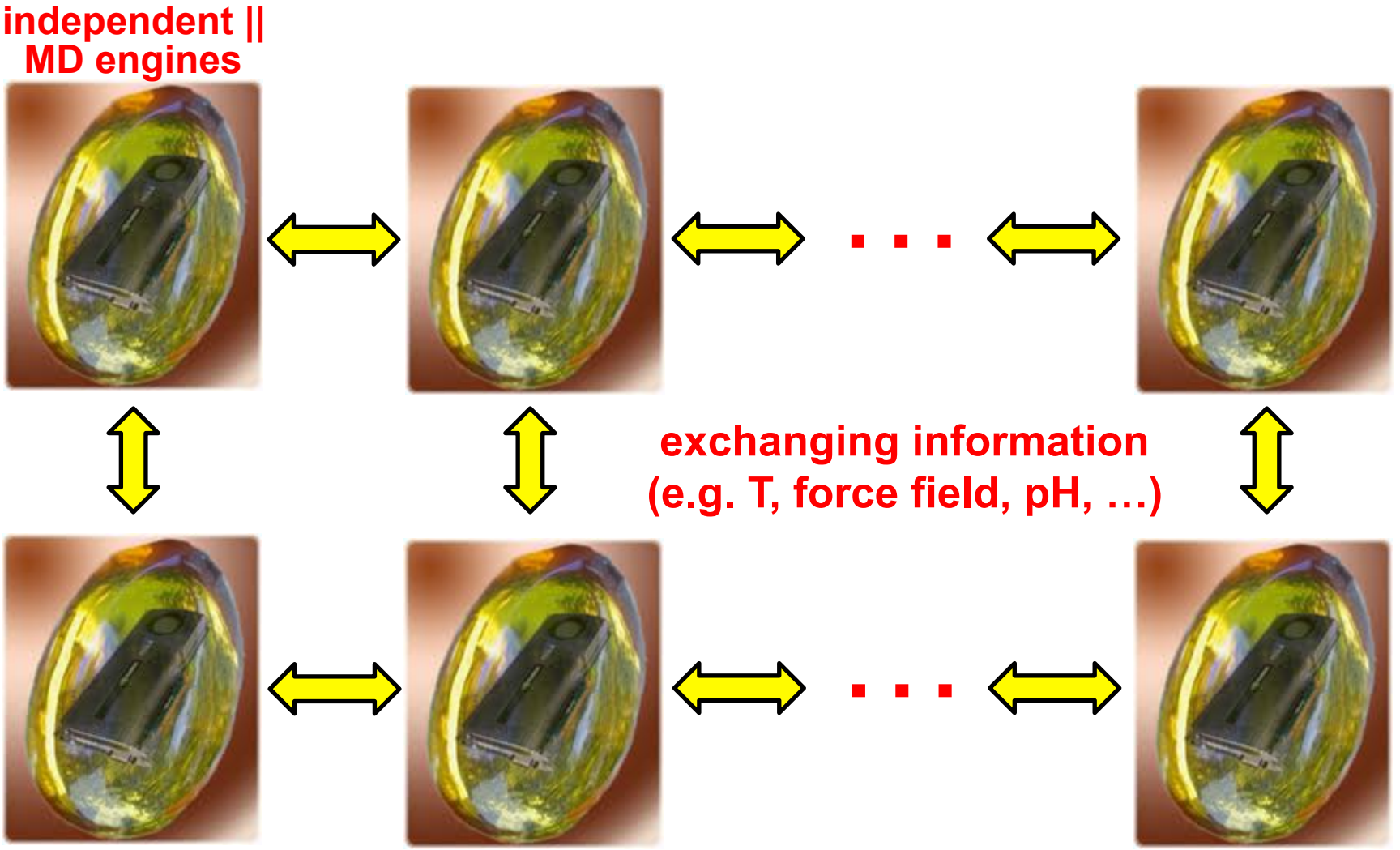
NEIS-P2 support
(split Utah / UCSD)

Utah: Dan Roe (PhD, staff / programmer)

UCSD: Romelia Salomon-Ferrer
(PhD, post-doc)

p.s. thanks B/W and NSF!!!

The main goals are to hierarchically and tightly couple a series of optimized molecular dynamics engines to fully map out the conformational, energetic and chemical landscape of RNA.



independent ||
MD engines =

~1978 - present

amber

Assisted Model Building with Energy-derived Restraints

~1978 - present

amber

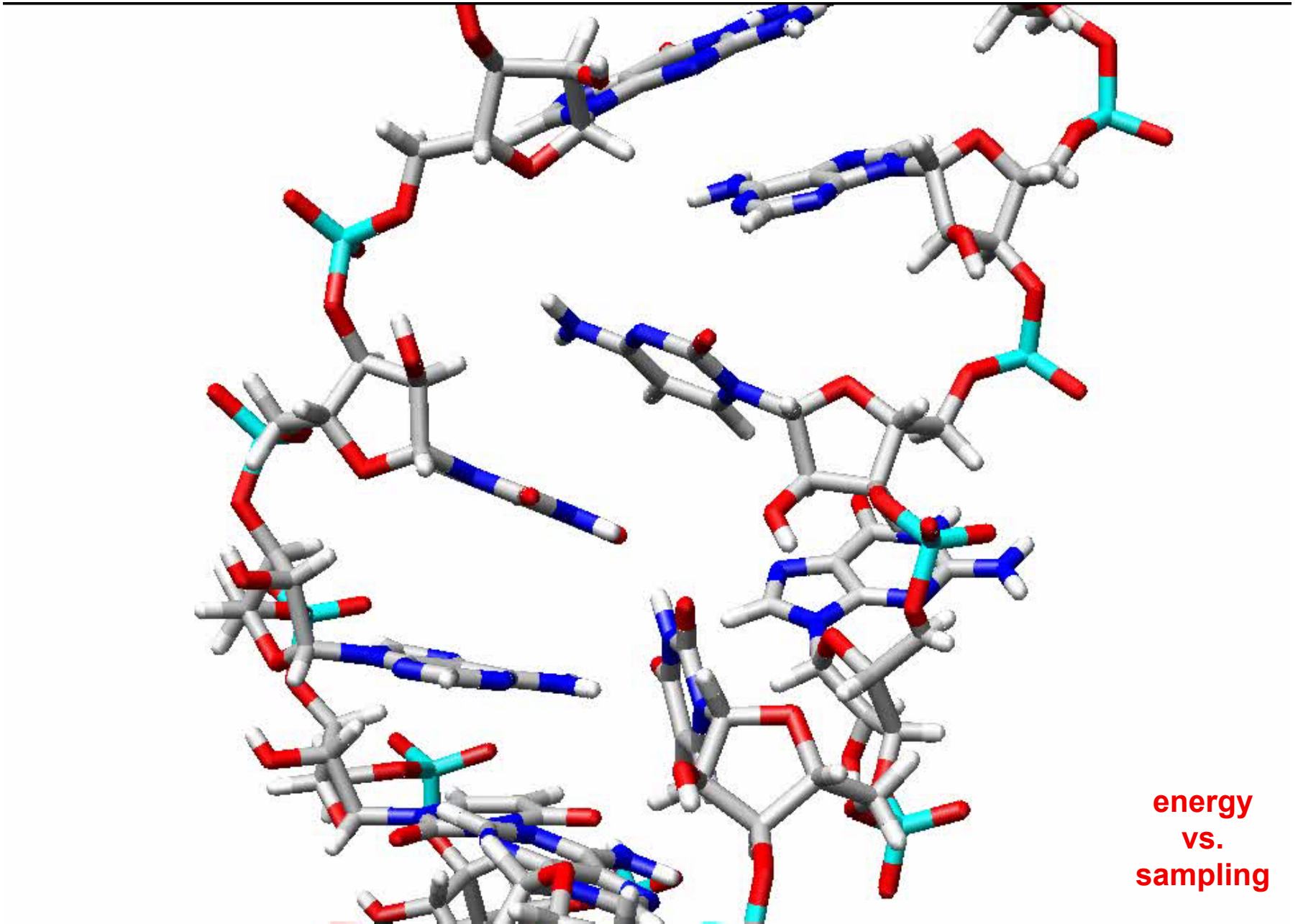
Assisted Model Building with Energy-derived Restraints

code vs. force field

the setup and calculation engines

the parameters and potentials

Science area: Simulation of RNA and proteins



energy
vs.
sampling

~1978 - present

amber

Assisted Model Building with Energy-derived Restraints

code vs. force field

the setup and calculation engines

the parameters and potentials

- Not really a professional code (some experts, some beginners)
- Not really software engineered (parts were, like GPU code, optimizations)
- It is continually evolving; one of the first “community codes”…
- Development efforts are not directly funded (except maybe GPU)

~1978 - present

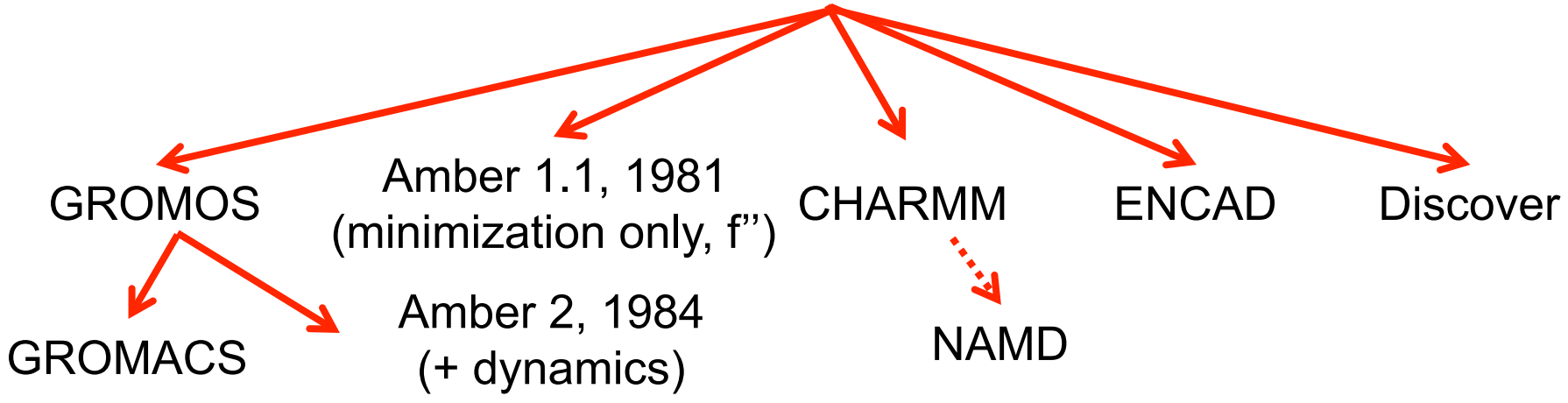
amber

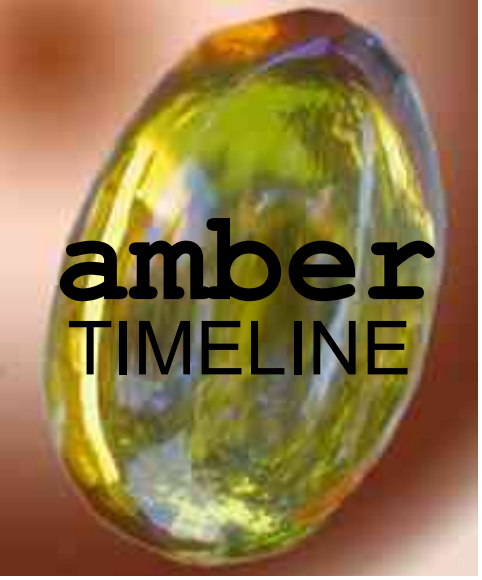
Assisted Model Building with Energy-derived Restraints

code vs. force field

late 60's: CFF (consistent force field) + early code
{Warshel, Levitt, Lifson}

1978: Bruce Gelin thesis @ Harvard {Karplus}





1986: amber3

ΔG, QM/MM, non-additivity

1989: amber3a

code cleanup, bug fixes
increased performance, portability
vectorization, || on hypercube,
shared memory
Intel Paragon 1/3 speed of Y-MP

1990-1994: SPASMS

?

(blue matter?)

1991: amber4.0

NMR refinement, normal modes, ΔG
serious code bifurcation
|| message passing
(TCGMSG, PVM, MPI, …)

1994: amber4.1

particle mesh Ewald ☺
more shared memory, MPI only
`#ifdef MPI`

early days: **ftp repository, makefiles (many), MACHINEFILE**

4.1-7.0: **CVS, C memory allocation move to F90, makefiles**
compile script recognizing MACHINEFILE
(fight w/ compiler for giganet vs. myrinet vs. …)

simplify, unify (as machines are becoming homogeneous)
drop vectorization, drop shared memory, drop machine specific opts

8.0: **introduce fast engine pmemd, configure scripts**

focus on fewer compilers: gnu, intel, pgi, pathscale
minimize #ifdefs to infrequently used code paths

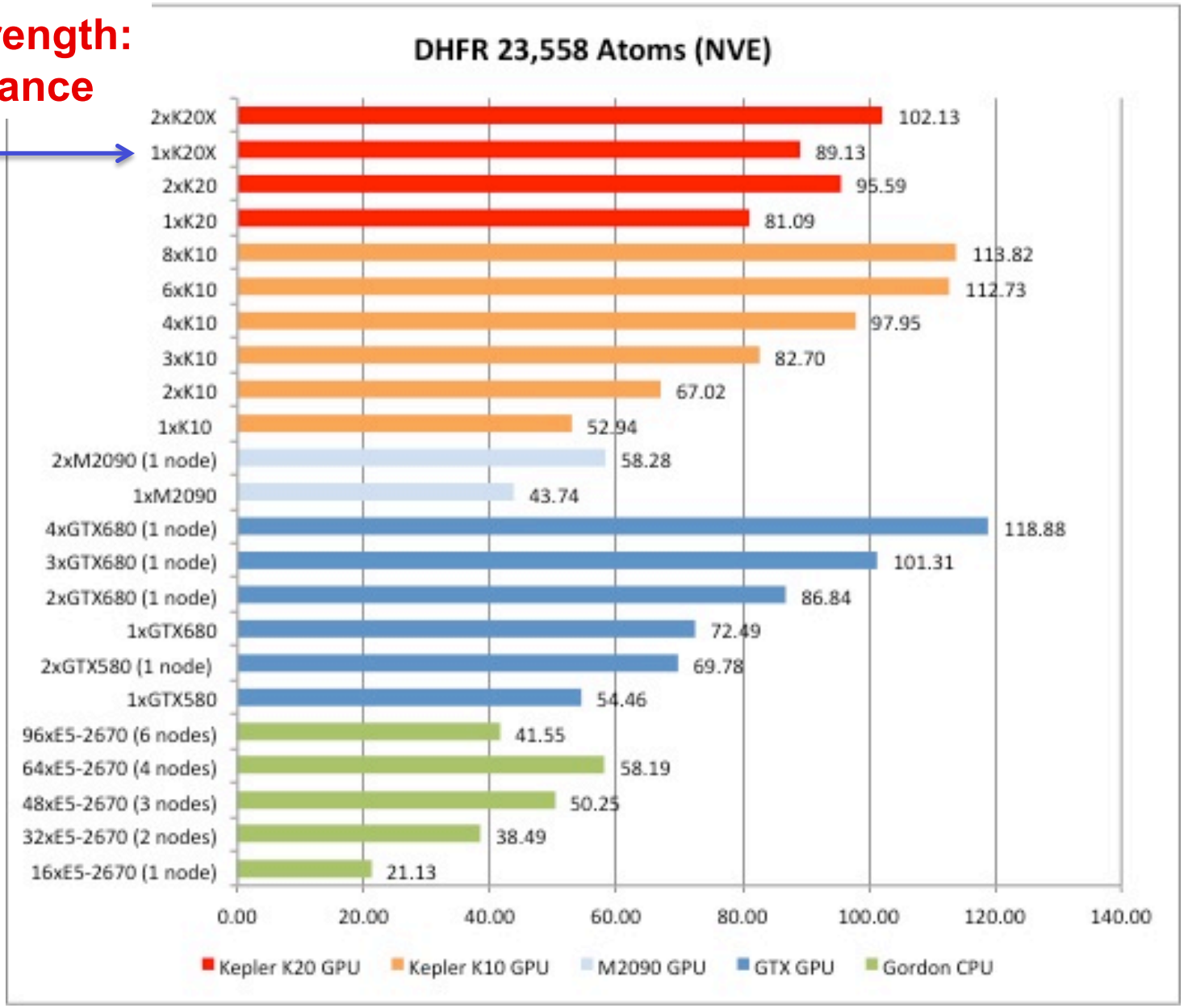
10.0: **AmberTools (open source), OpenMP**
separate configure for AmberTools, sander, pmemd

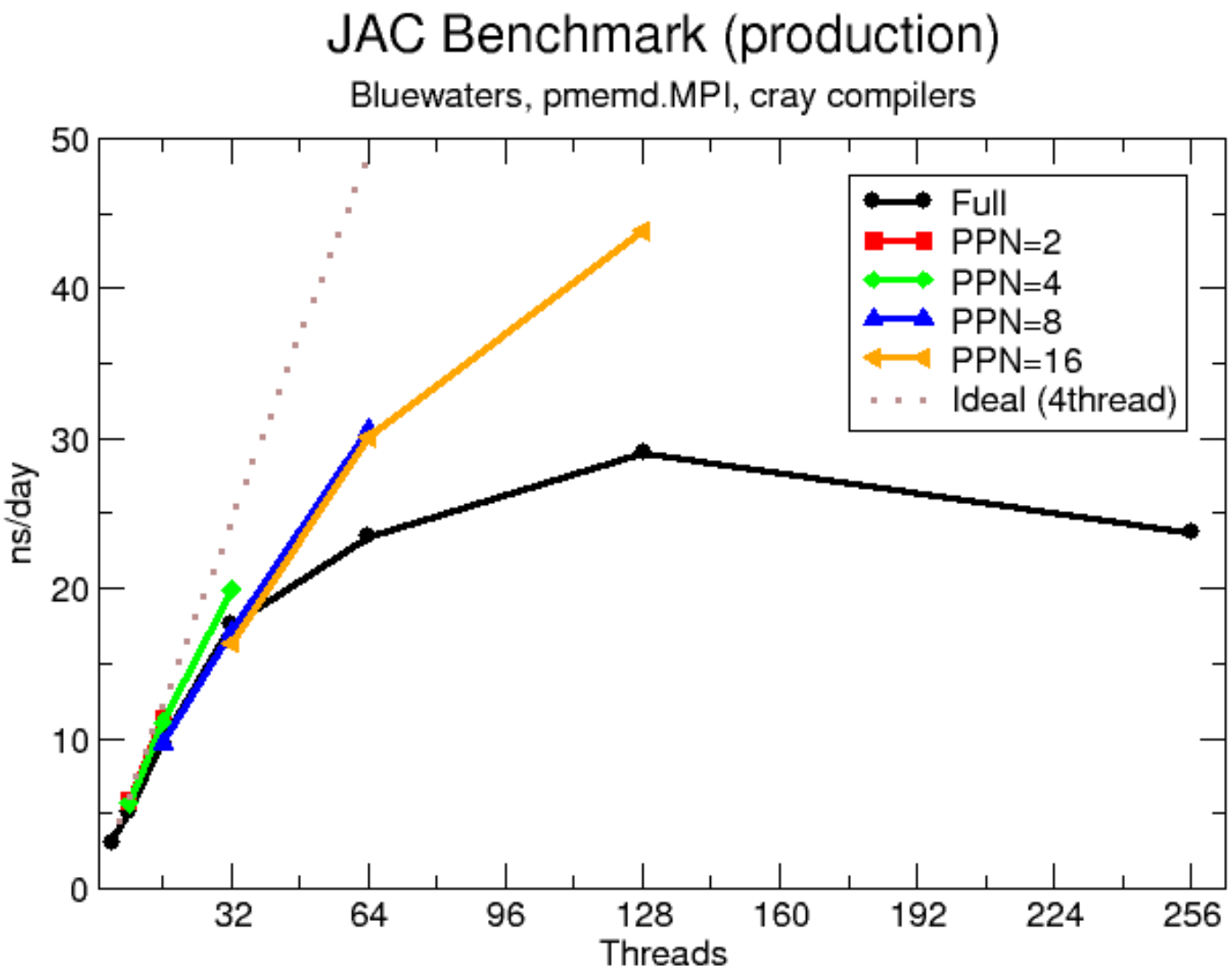
11:0: **git tree, full F90, makedepend**

12.0: **Unified “configure” script, automatic bug patching**

(JAC DHFR Production Benchmark)

Key AMBER strength: GPU performance





1 K20X = 89.1 ns / day (81.4 on Cray **`xk`**, ↓ 9%)
2 K20X = 102.1 ns / day

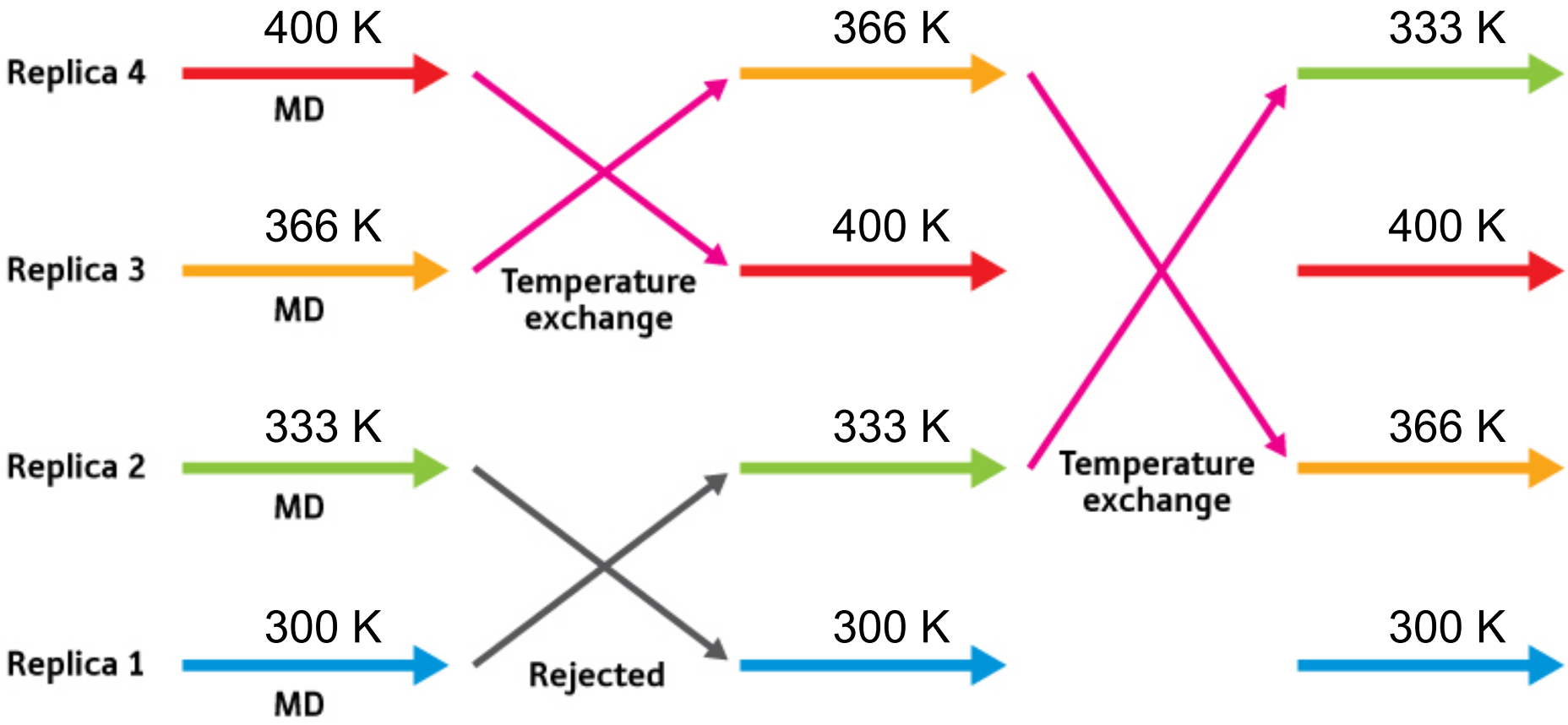
NEIS-P2 SOW

- Design / implement multi-dimensional REMD in PMEMD on CPU and GPU
- Implement accelerated MD (aMD) on CPU and GPU
- Integrate aMD into multi-D REMD
- Design new REMD trajectory format (support multi-D)
- Extend analysis codes (**`cpptraj`**) to understand multi-D REMD data
- Optimize on Blue Waters
- Code up NetCDF checkpoint “restart” formatted files

“our work is never done…”
(devil is in the details)

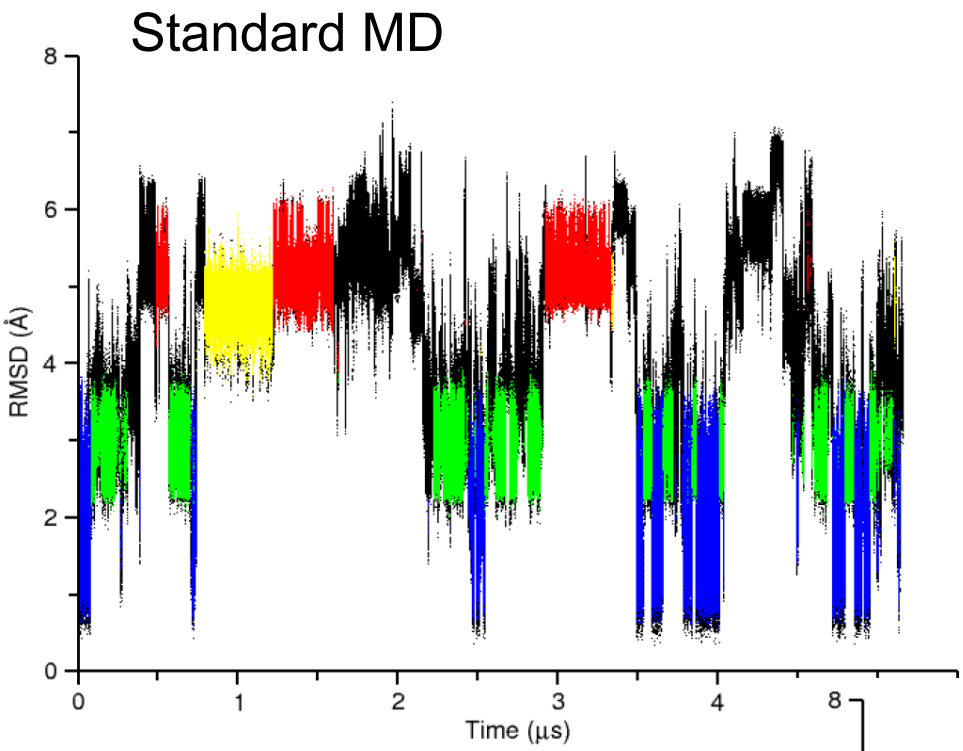
[Main AMBER GIT tree constantly changing,
not all options work everywhere,
tuning required, e.g. this is research…]

REMD = replica exchange molecular dynamics



(replica trajectories span all temperatures; to understand the properties at a particular temperature, we need to sort the replica trajectories; this is automated in **`cpptraj`**)

www.rikenresearch.riken.jp

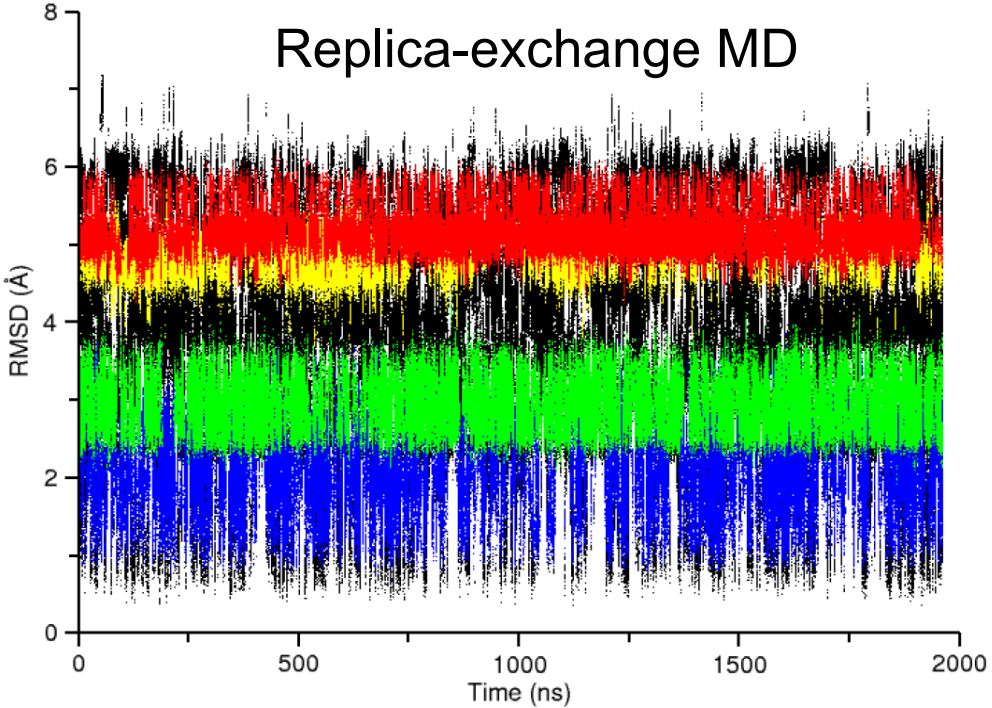


r(GACC) tetranucleotide

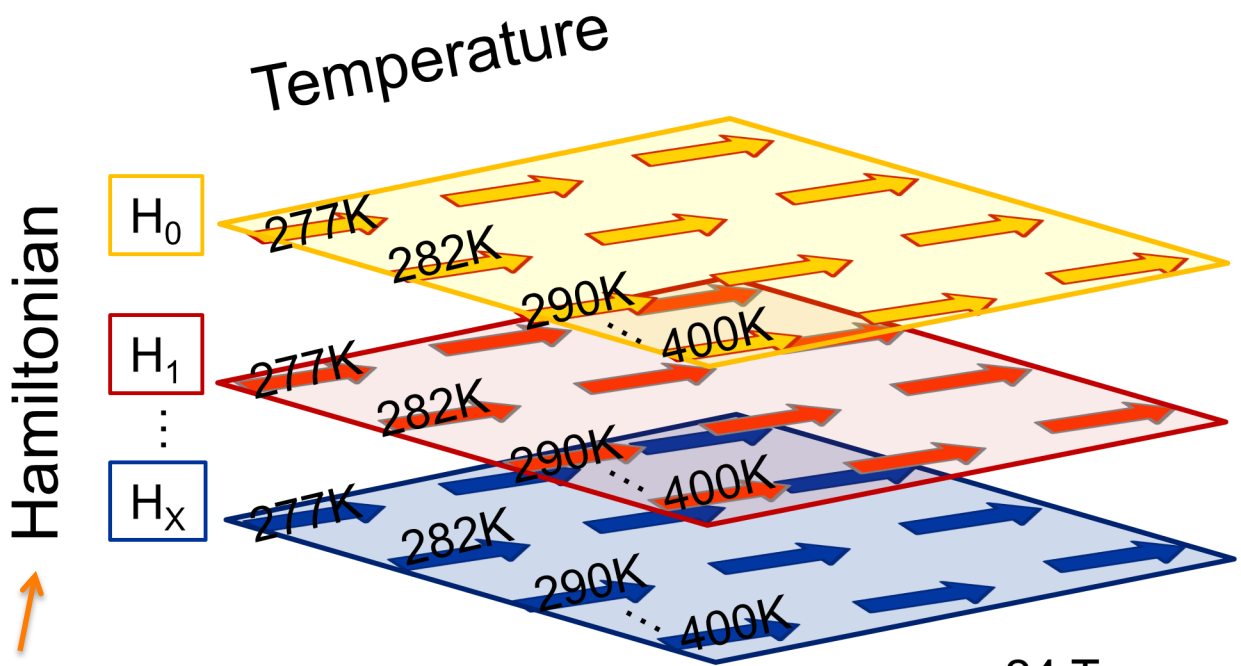
[Turner / Yildirim]

< explicit solvent >

…a system where we can get complete sampling



multi-D REMD



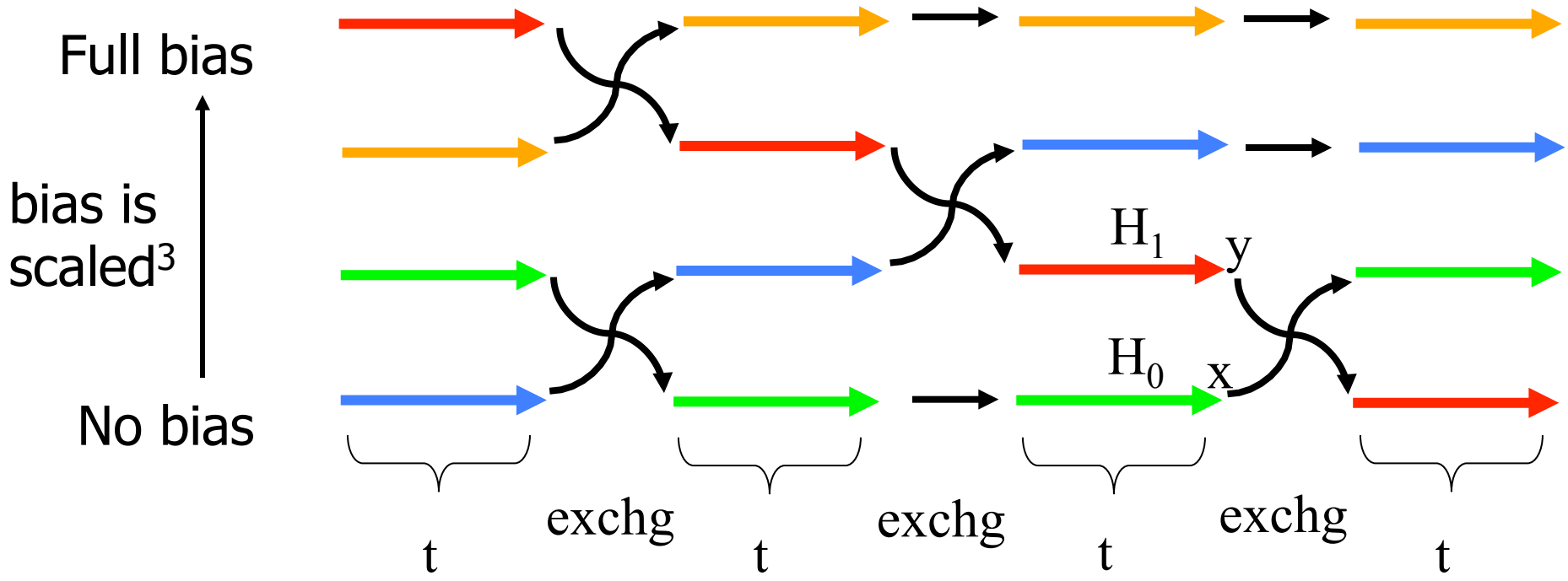
Change in "energy representation"

- **pH**
- **restraints, umbrella potentials, …**
- **force field / parameter sets**
- **biasing potentials (aMD)**

24 Temperatures
x 8 Hamiltonians
=192 replicas

Fukunishi, H., Wanatabe, O., and Takada, S., J. Chem. Phys. 2002.
Sugita, Y., Kitao, A., and Y. Okamoto, J. Chem. Phys. 2000.

Hamiltonian Replica Exchange Molecular Dynamics [1,2] (HREMD)



$$\frac{w\left[H_0(x)H_1(y)\rightarrow H_1(x)H_0(y)\right]}{w\left[H_1(x)H_0(y)\rightarrow H_0(x)H_1(y)\right]}=e^{\beta\left[\left(H_0(x)-H_0(y)\right)-\left(H_1(x)-H_1(y)\right)\right]}$$

1.Fukunishi, H., Wanatabe, O., and Takada, S., J. Chem. Phys. 2002.

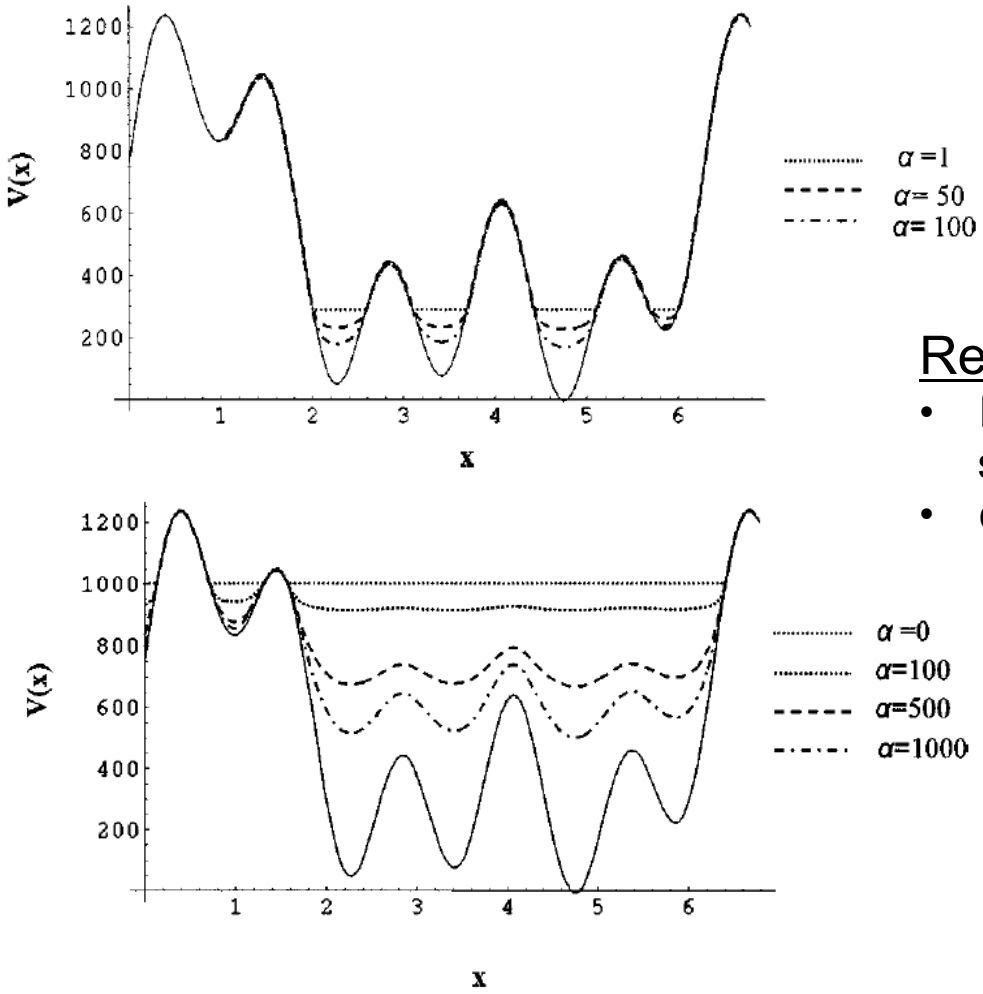
2.Sugita, Y., Kitao, A., and Y. Okamoto, J. Chem. Phys. 2000.

3.Kannan, S., and Zacharias, M., Proteins. 2006

aMD implementation

Hamelberg, Mongan, McCammon, J. Chem. Phys., 2004.

(sander only, port to PMEMD and GPU code)



Low Ethresh, changes effected with smaller alpha values

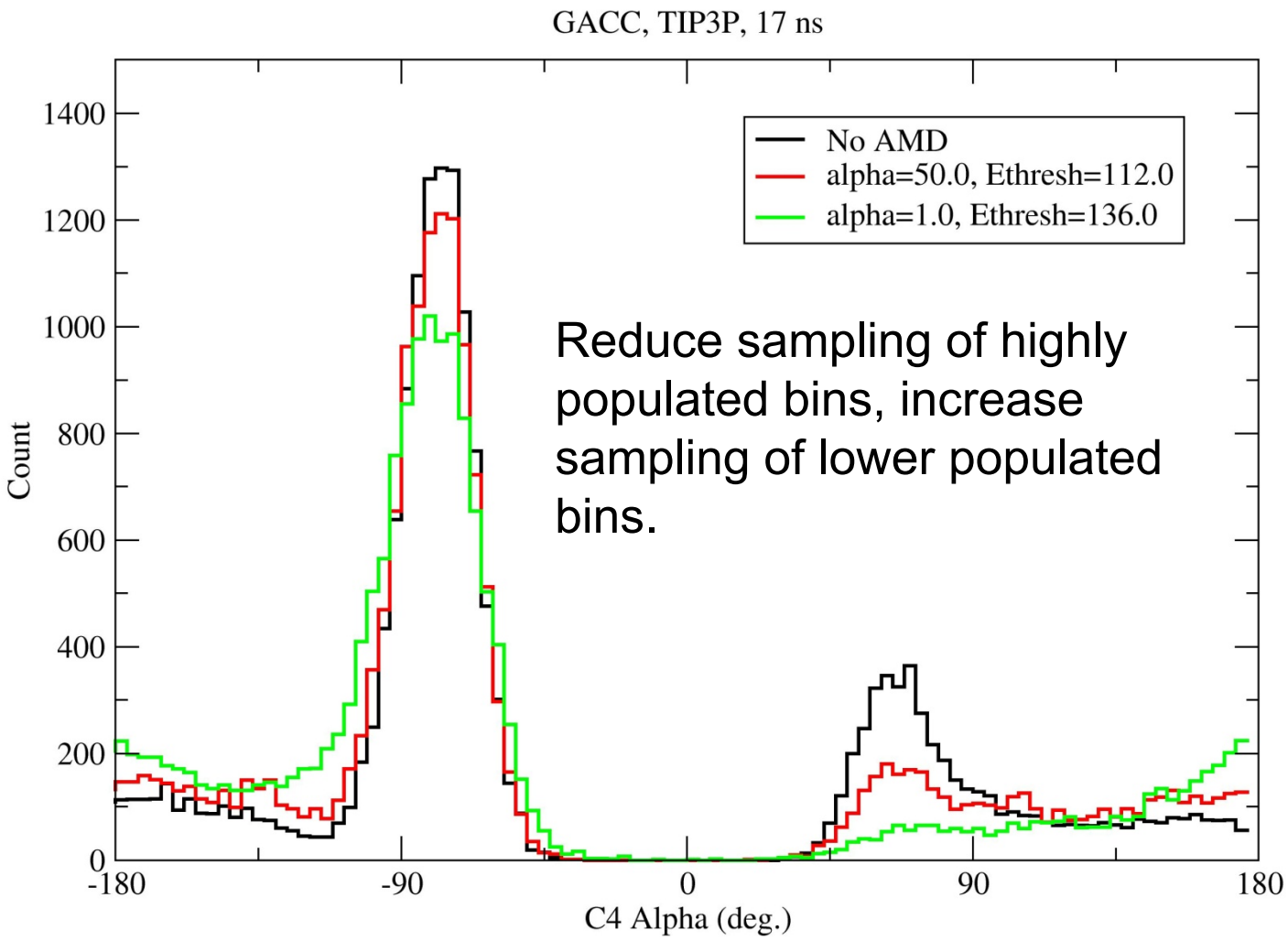
<u>Recommendation:</u>

- E > Vmin, magnitude depends on how much sampling is desired.
- α = E-Vmin echoes shape of potential wells.

high Ethresh, changes effected with larger alpha values

(low values of α (=0) landscape is isoenergetic, random walk)

AMD, Boost Dihedral Energy



Reduce sampling of highly populated bins, increase sampling of lower populated bins.

aMD + H-REMD problem

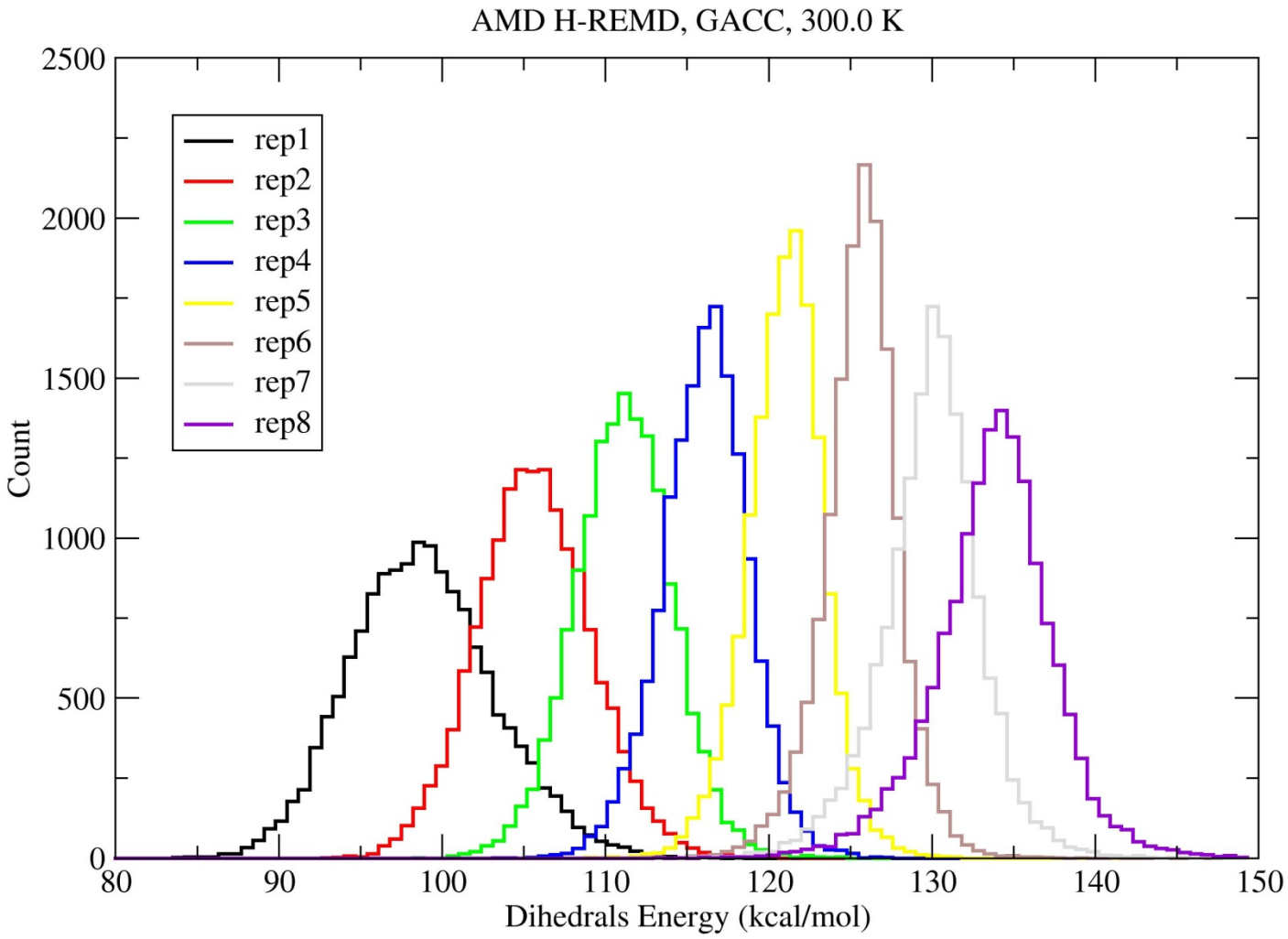
The boosted potential is <u>not</u> added to total potential energy. The Hamiltonian exchange is performed using UN-boosted potential, which is the same for all reps – therefore delta delta E is the same for each coordinate set.

```
# Replica Exchange log file
# numexchg is        100
# REMD filenames:
#   remlog= rem.log
#   remtype= rem.type
# Rep#, Neibr#, Temp0, PotE(x_1), PotE(x_2), left_fe, right_fe, Success, Success rate (i,i+1)
# exchange       1
    1     4    281.30 -25603.25 -25603.25      0.00      0.00    T       0.00
    2     3    281.30 -25603.25 -25603.25      0.00      0.00    T       2.00
    3     2    281.30 -25603.25 -25603.25      0.00      0.00    T       0.00
    4     1    281.30 -25603.25 -25603.25      0.00      0.00    T       2.00
# exchange       2
    1     2    281.30 -25467.08 -24129.32      0.00      0.00    T       1.00
    2     1    281.30 -24129.32 -25467.08      0.00      0.00    T       1.00
    3     4    281.30 -22796.29 -21381.93      0.00      0.00    T       1.00
    4     3    281.30 -21381.93 -22796.29      0.00      0.00    T       1.00
# exchange       3
    1     4    281.30 -25226.72 -21410.60      0.00      0.00    T       0.67
    2     3    281.30 -24115.73 -22747.15      0.00     -0.00    T       1.33
    3     2    281.30 -22747.15 -24115.73     -0.00      0.00    T       0.67
    4     1    281.30 -21410.60 -25226.72      0.00     -0.00    T       1.33
# exchange       4
    1     2    281.30 -24818.04 -24137.06      0.00      0.00    T       1.00
    2     1    281.30 -24137.06 -24818.04      0.00     -0.00    T       1.00
    3     4    281.30 -22817.18 -21416.11     -0.00      0.00    T       1.00
    4     3    281.30 -21416.11 -22817.18     -0.00     -0.00    T       1.00
```

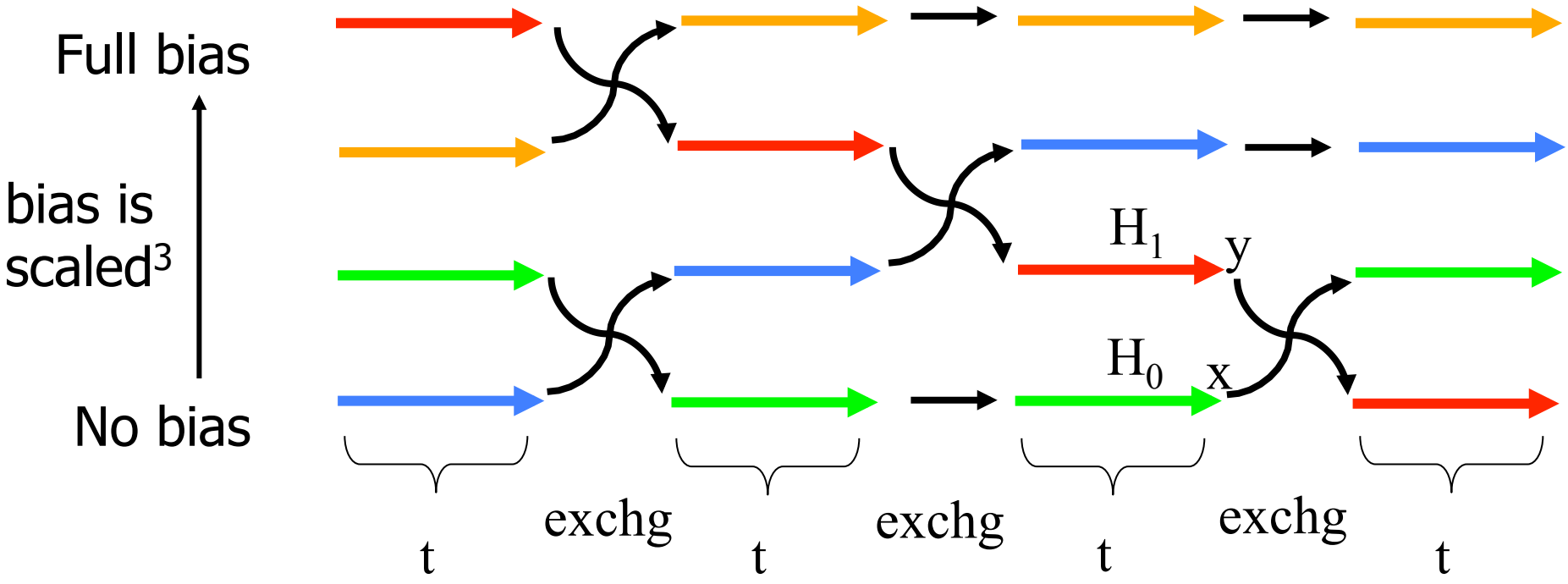
All exchanges attempted are successful. they are not sampling Boltzmann weighted ensemble because Metropolis criteria E doesn't include boosted potential energy.

AMD H-REMD on Bluewaters

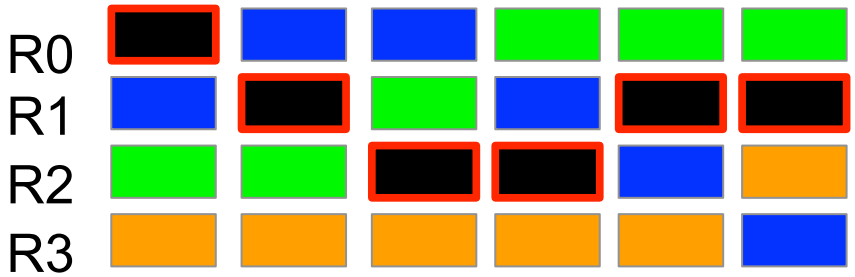
problem #2: How to get good overlap?



problem #3: How to “unbias” the aMD biasing?

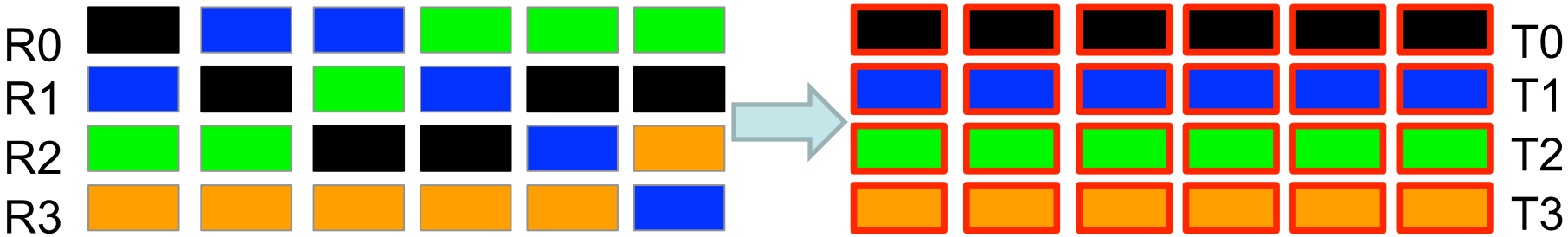


Ensemble Trajectory Processing



- Each REMD trajectory contains frames that may be at different temperatures.
- Previous method: Read in all frames at frame X, pick target frame, process.
- Even though all frames read, only one used!

Ensemble Trajectory Processing



- New 'ensemble' command allows reading and processing of entire ensemble.
- All frames read in are used; sorting performed if necessary. Works in multiple dimensions.
- Actions are run on every member of ensemble after sort; output is directed to a single file.
- Multiple input ensembles can be directed to one sorted ensemble of output trajectories.

use tiered resources to facilitate data analysis

streamed to memory: small size, lifetime ~hours, trivial analysis (very fast timescales) "steering"

/scratch

|| disk, lifetime ~weeks, detailed analysis (dump at rate equivalent to disk speed)

flash: moderate size, lifetime ~days, less trivial analysis (fast timescales)

spinning disk

~months

BW issues:

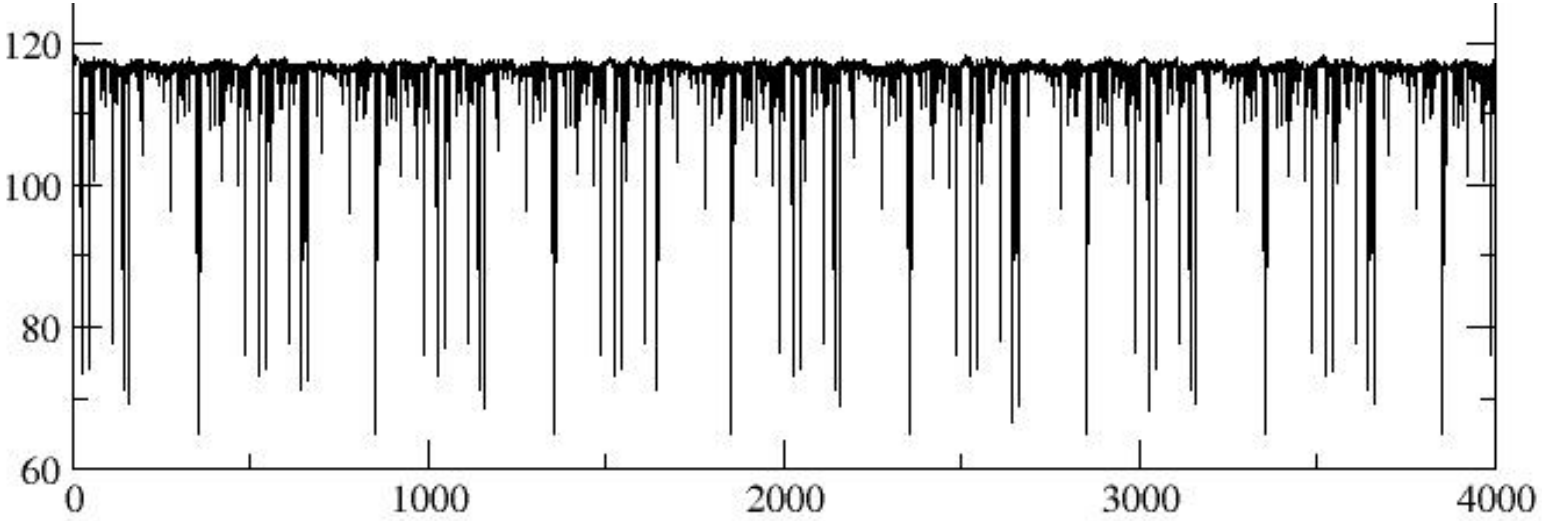
- GPU performance equals ~K20 instead of K20X (🡻 ~9%)
- GPU's + multi-D REMD too fast = too much data!!!
- Compiling status:

	CPU	GPU	
gnu	✖	✔	
pgi	✔	✖	Cuda headers, need nvcc
cray	✖	✖	

FFT #'s wrong – over aggressive optimizations

BW issues:

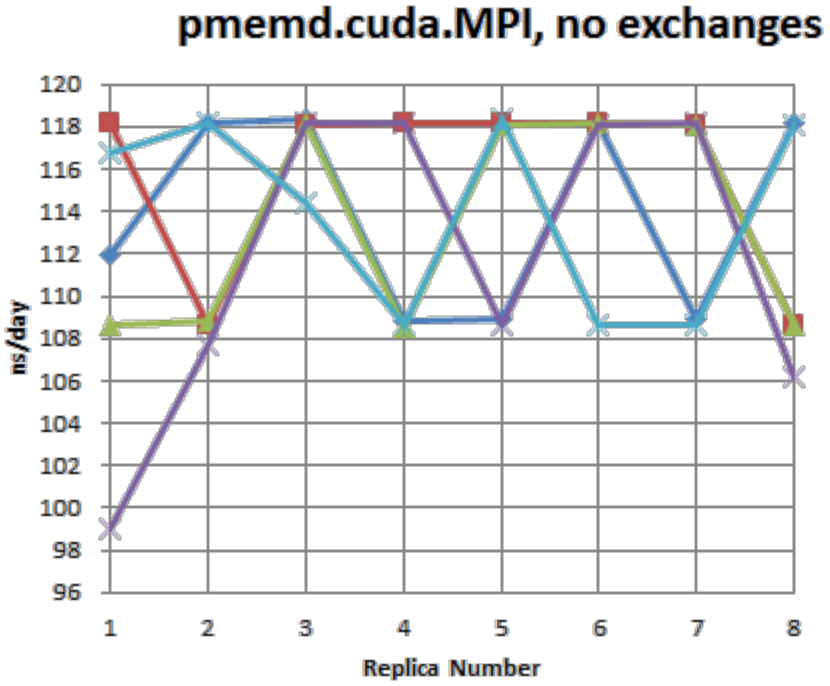
- Variable runtime performance (likely I/O related + dependent on load)



==> 0-30ns/rep.000.out <==
| Master Total wall time: 33647 seconds **9.35 hours**
==> 61-90ns/rep.000.out <==
| Master Total wall time: 26096 seconds **7.25 hours**
==> 91-120ns/rep.000.out <==
| Master Total wall time: 25581 seconds **7.11 hours**
==> 121-150ns/rep.000.out <==
| Master Total wall time: 25875 seconds **7.19 hours**
==> 151-180ns/rep.000.out <==
| Master Total wall time: 26120 seconds **7.26 hours**
==> 181-210ns/rep.000.out <==
| Master Total wall time: 25075 seconds **6.97 hours**
==> 211-240ns/rep.000.out <==
| Master Total wall time: 27697 seconds **7.69 hours**
==> 241-270ns/rep.000.out <==
| Master Total wall time: 23535 seconds **6.54 hours**

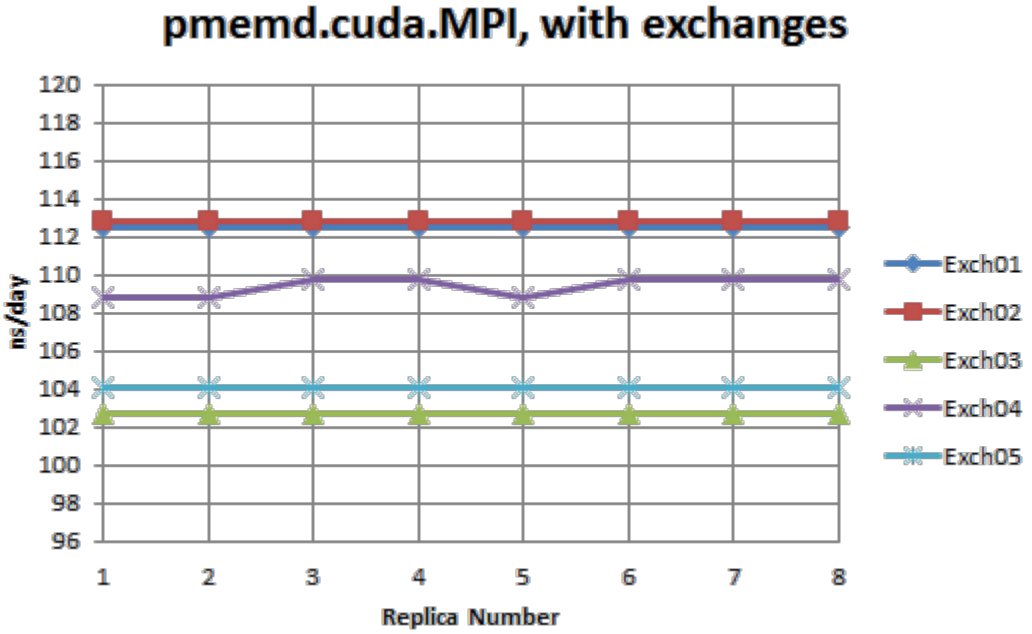
==> 271-300ns/rep.000.out <==
| Master Total wall time: 22907 seconds **6.36 hours**
==> 301-330ns/rep.000.out <==
| Master Total wall time: 24276 seconds **6.74 hours**
==> 31-60ns/rep.000.out <==
| Master Total wall time: 22429 seconds **6.23 hours**
==> 331-360ns/rep.000.out <==
| Master Total wall time: 24356 seconds **6.77 hours**
==> 361-390ns/rep.000.out <==
| Master Total wall time: 25408 seconds **7.06 hours**
==> 391-420ns/rep.000.out <==
| Master Total wall time: 24728 seconds **6.87 hours**
==> 421-450ns/rep.000.out <==
| Master Total wall time: 25972 seconds **7.21 hours**

[higher ns/day (y-axis) means better performance]



Shows periodic / random slowdown. Since no exchanges, this is due to file I/O and not due to exchanging information necessary for H-REMD)

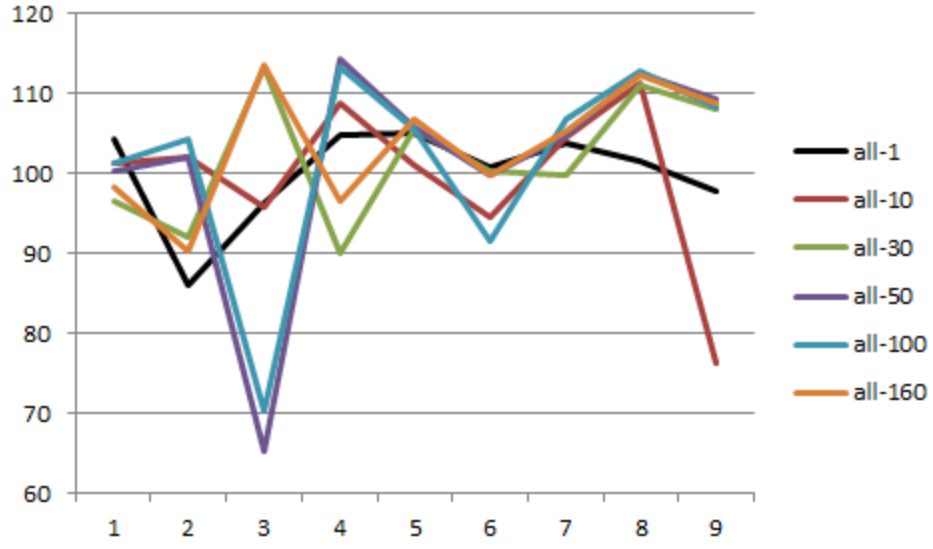
When exchanging information necessary for H-REMD you have to wait for slowest MD engine

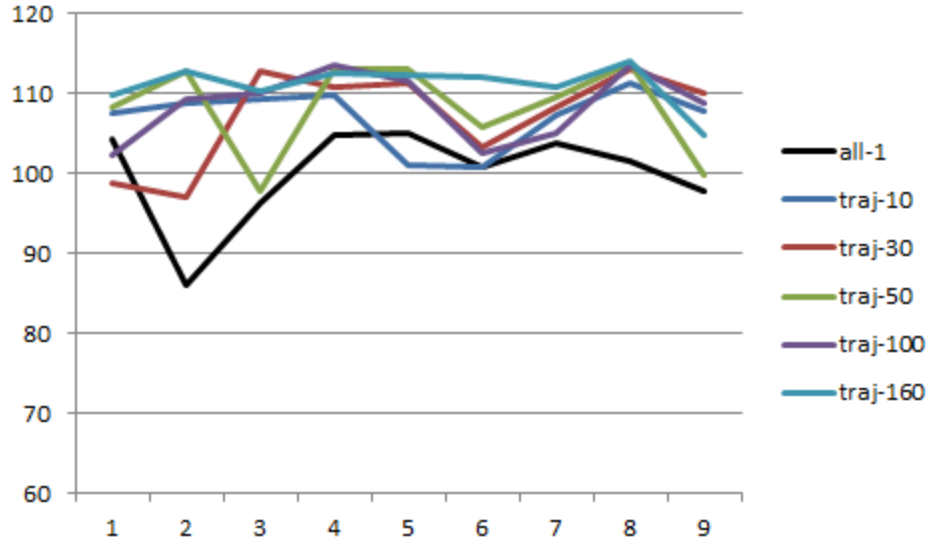


BW issues:

- Variable runtime performance (likely I/O related + dependent on load)
- I/O performance: How to improve?
  - Use NetCDF checkpoint files instead of ASCII (**`ntx=2`**) ✔✔✔
  - Lustre striping on big files improves performance

Performance across replicas as a function of Lustre striping





BW issues:

- Variable runtime performance (likely I/O related + dependent on load)
- I/O performance
  - Use NetCDF checkpoint files instead of ASCII (**`ntx=2`**)
  - Lustre striping
  - Q: Anyone experimented with RDMA on XK nodes?

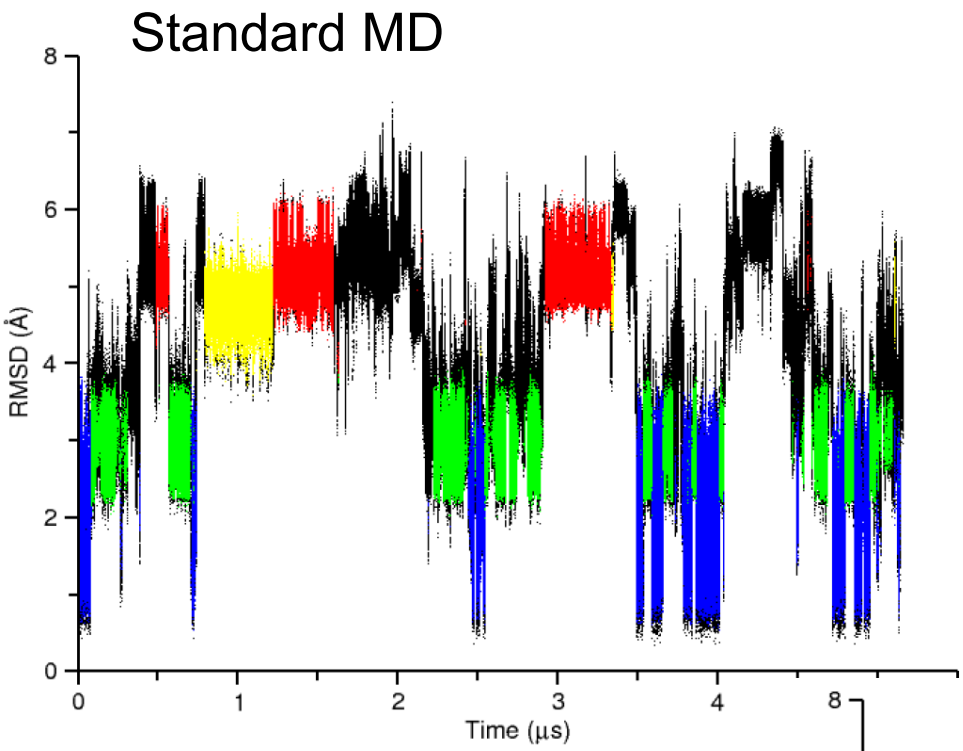
Other plans:

- async H-REMD / REMD exchange
- async file I/O
  - virtual via buffering and round-robin write
  - true via buffering in memory on I/O nodes
  - RDMA – put GPU back to work with alternate I/O thread
- 2 to 1: merge checkpoint info into binary trajectory

Other plans:

- Problem: the code path for ONE GPU with MPI already loses ~20% performance

- Solution: multiple MPI code paths

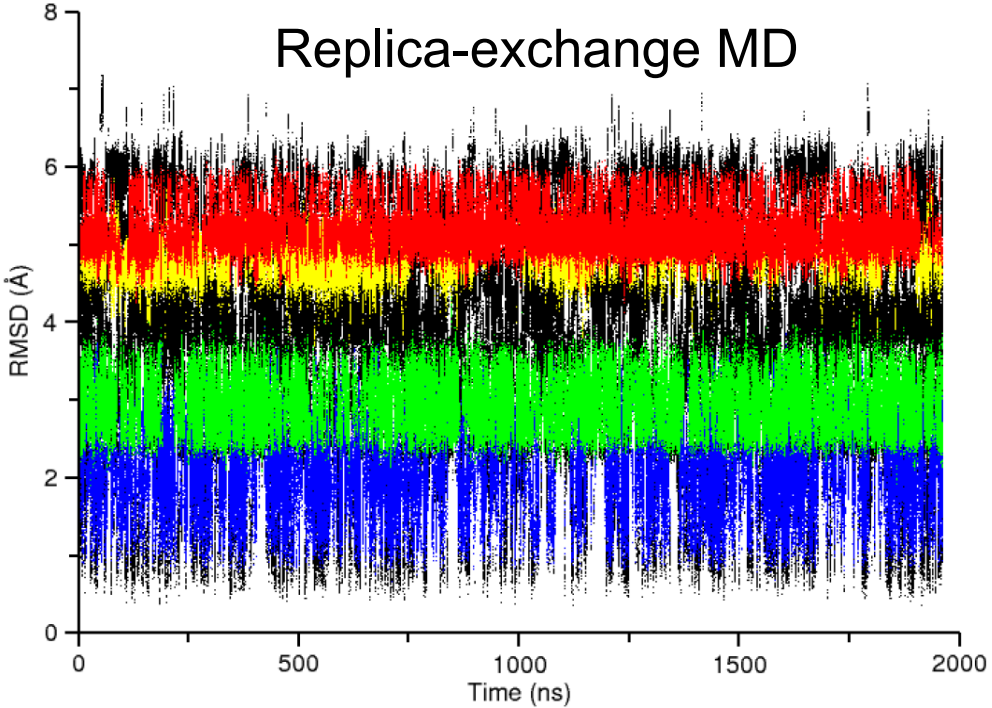


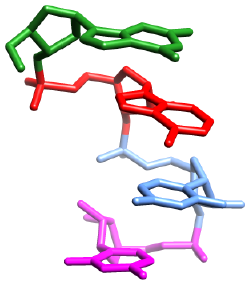
r(GACC) tetranucleotide

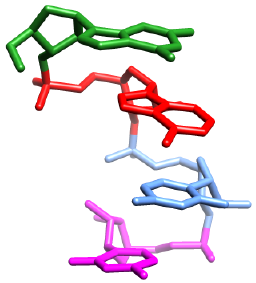
[Turner / Yildirim]

< explicit solvent >

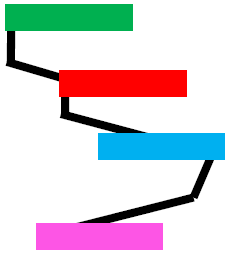
…a system where we can get complete sampling

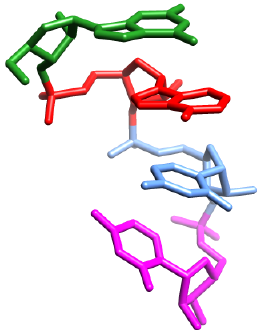


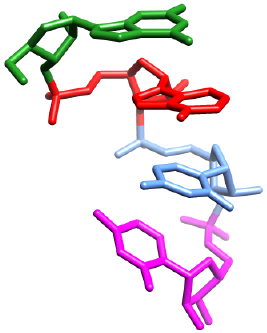




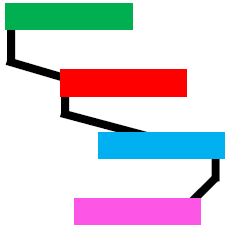
NMR Minor
(Green)

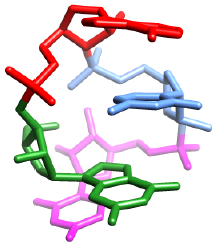
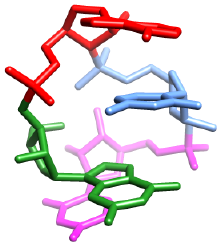




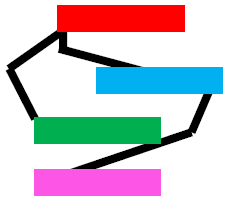


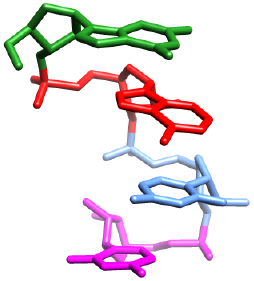
NMR Major
(Blue)



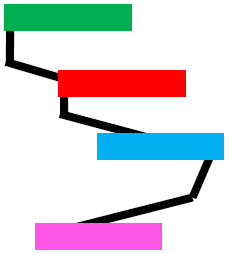



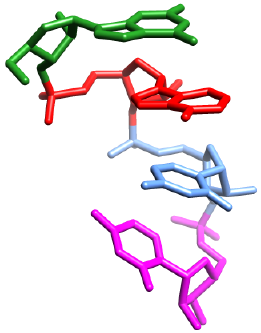
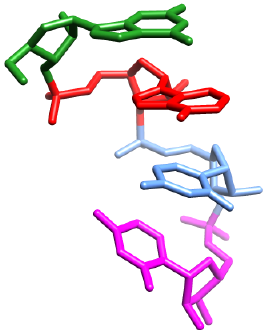
Intercalated
(Red)



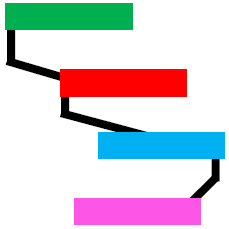



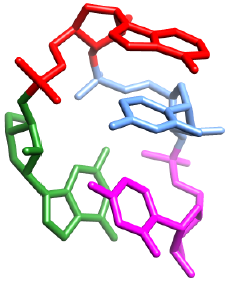
NMR Minor
(Green)



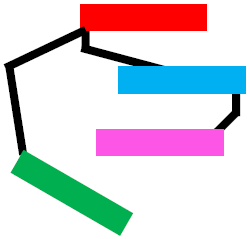



NMR Major
(Blue)

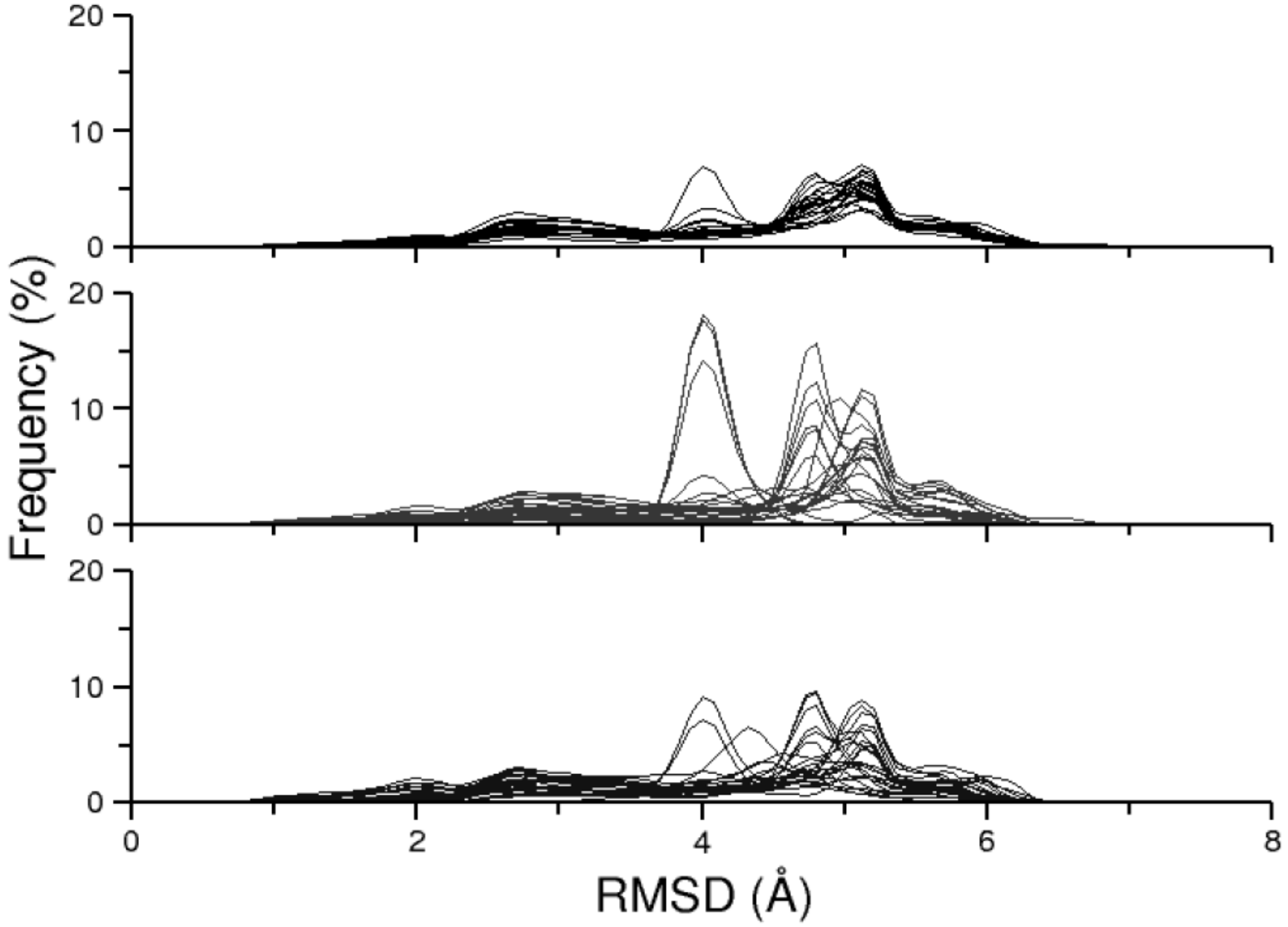




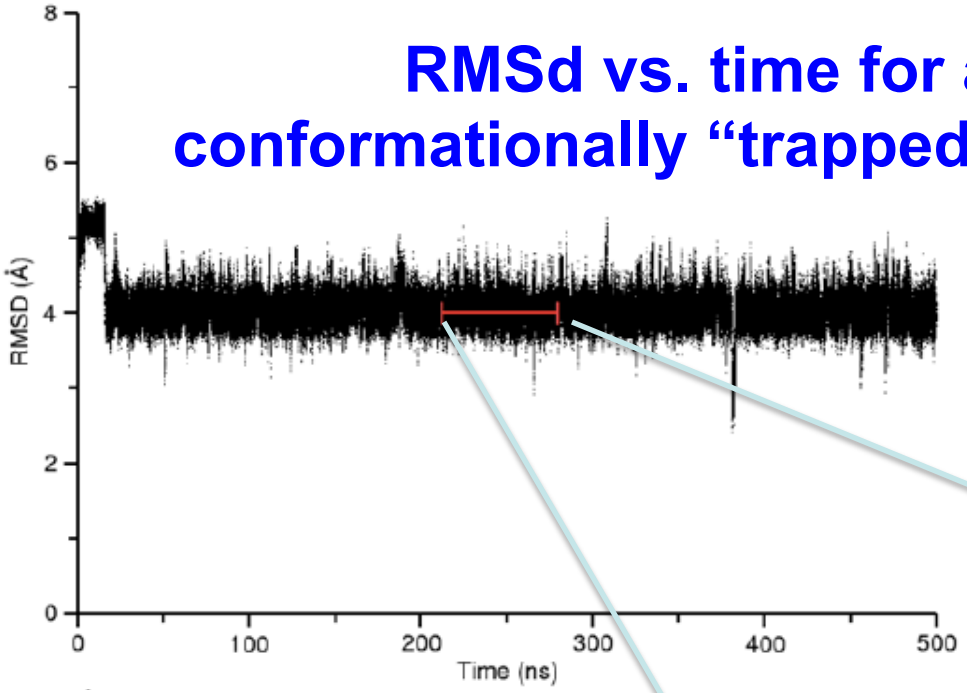

Inverted
(Yellow)



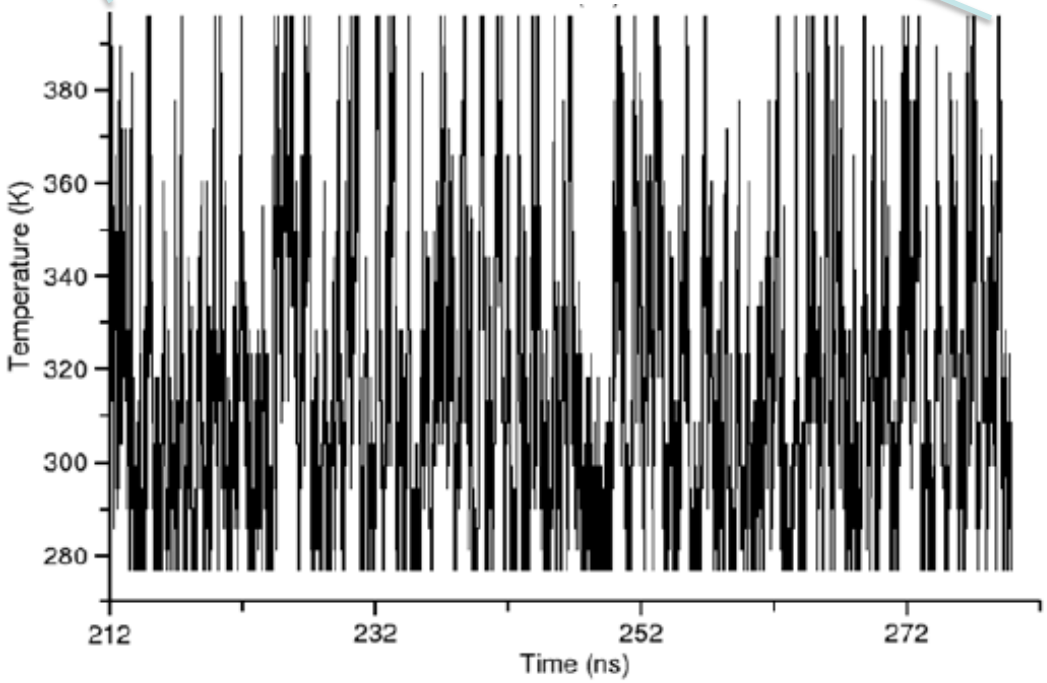
RMSd profiles per replica (they should be the same)



RMSd vs. time for a conformationally "trapped" replica



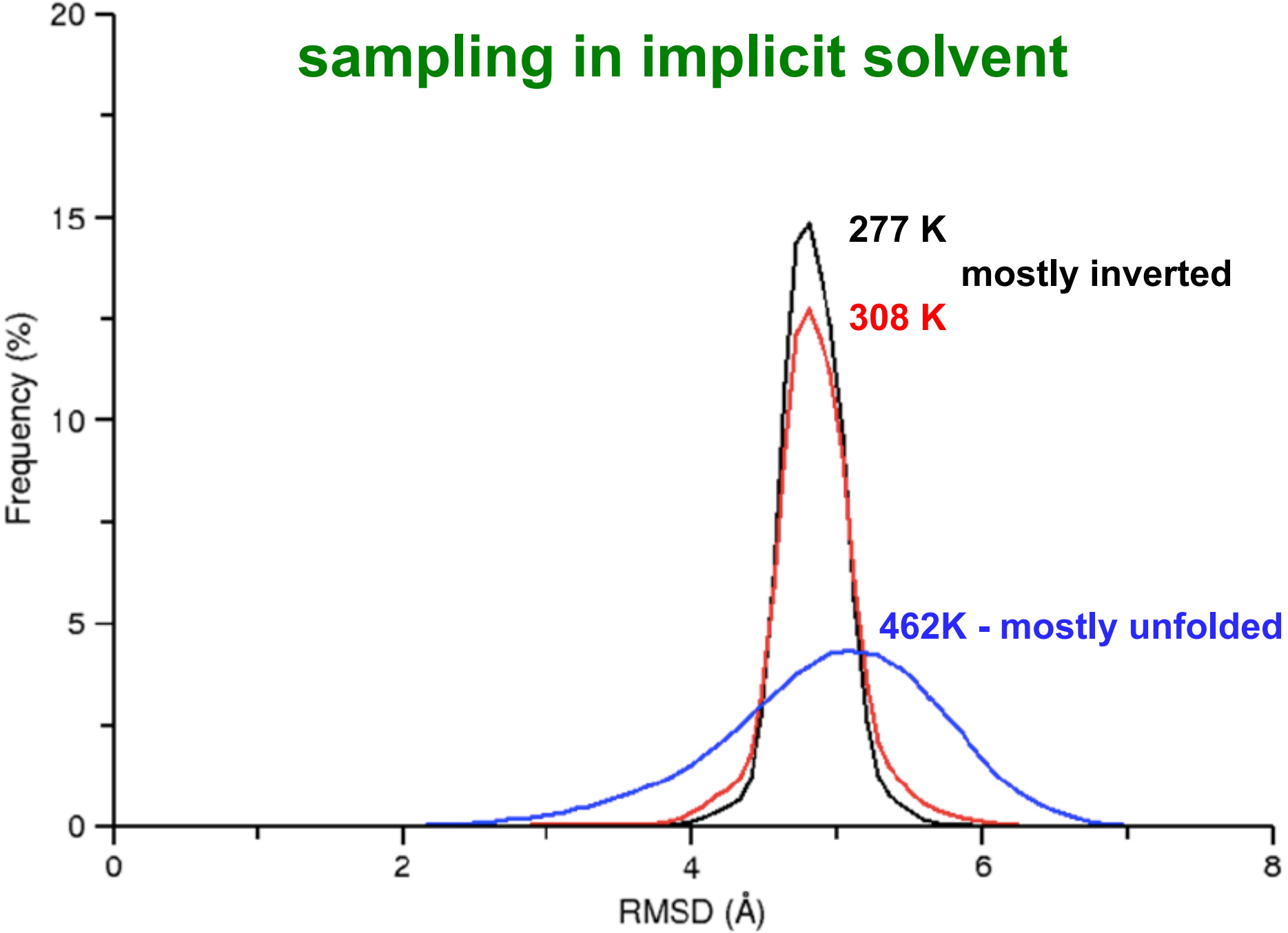
temperature distribution



	rGACC Conformational Frequency (%)			
Simulation ID	Intercalated	NMR Minor	NMR Major	Inverted
RNA-NPT[1]	16.0 (0.3)	12.9 (0.7)	9.2 (0.4)	8.4 (0.1)
RNA-398[2]	6.2 (0.4)	3.5 (0.3)	3.1 (0.1)	7.1 (0.5)
RNA-REMD-GB	-- --	-- --	-- --	92.9 (0.7)
RNA-REMD-1	24.5 (0.9)	15.9 (0.7)	11.8 (0.6)	7.6 (0.0)
RNA-REMD-2	24.2 (1.2)	10.5 (1.0)	8.8 (0.5)	9.9 (0.1)
RNA-REMD-3	18.8 (0.9)	16.3 (1.0)	13.1 (0.5)	7.3 (0.1)
RNA-rREMD-S	29.4 (0.1)	28.3 (1.1)	12.0 (0.2)	-- --
RNA-rREMD-1	18.7 (0.3)	15.5 (0.7)	13.1 (0.4)	11.3 (0.1)
RNA-rREMD-2	18.5 (0.1)	15.3 (1.0)	13.6 (0.1)	10.9 (0.0)
RNA-rREMD-3	18.7 (0.5)	14.6 (0.7)	14.0 (0.4)	10.2 (0.1)

rREMD = reservoir REMD

sampling in implicit solvent



Potential NOEs of inverted conformation projected onto major conformation