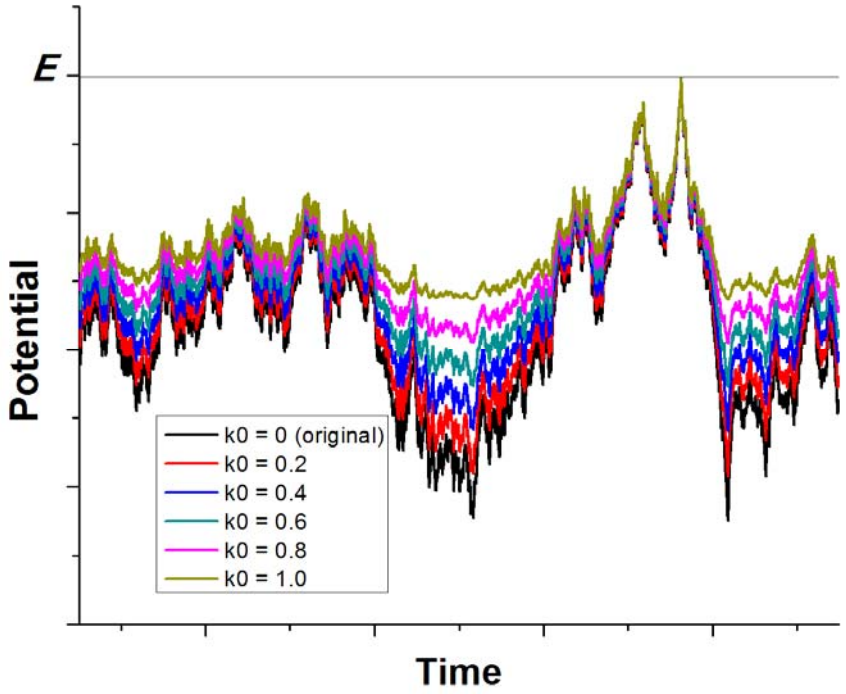
Gaussian Accelerated Molecular Dynamics (GaMD) Tutorial

Yinglong Miao & Ross Walker

Introduction 1
1. Install GaMD supported Amber 2
2. Run GaMD simulations on Alanine Dipeptide 4
3. Simulation Analysis 7

Introduction

Gaussian Accelerated Molecular Dynamics (GaMD) is a biomolecular enhanced sampling method that works by adding a harmonic boost potential to smoothen the system potential energy surface.

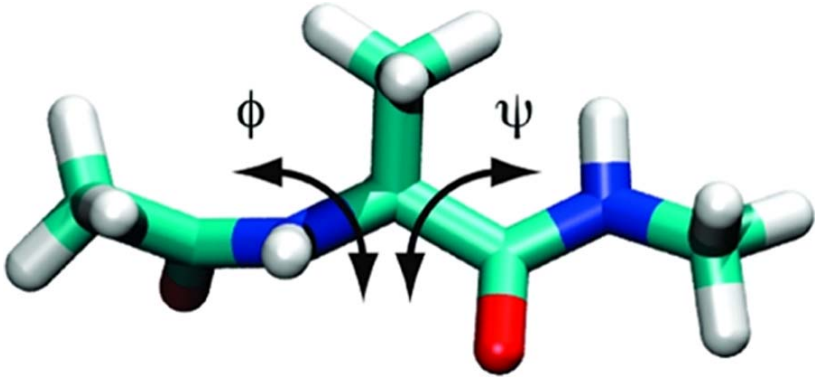


By constructing a boost potential that follows Gaussian distribution, accurate reweighting of the GaMD simulations is achieved using cumulant expansion to the second order. GaMD has been demonstrated on three biomolecular model systems: alanine dipeptide, chignolin folding and ligand binding to the T4-lysozyme. Without the need to set predefined reaction coordinates, GaMD enables unconstrained enhanced sampling of these biomolecules. Furthermore, the free energy profiles obtained from reweighting of the GaMD simulations allow us to identify distinct low energy states of the biomolecules and characterize the protein folding and ligand binding pathways quantitatively.

Reference: Miao, Y.; Feher, V. A.; McCammon, J. A., Gaussian Accelerated Molecular Dynamics: Unconstrained Enhanced Sampling and Free Energy Calculation. *J. Chem. Theory Comput.* **2015,** *11*, 3584-3595.

In this tutorial, we will use alanine dipeptide as a model system to learn the following:

- How to run GaMD simulations with different dihedral-, total- and dual-boost potentials?
- How to analyze simulation trajectories?
- How to reweight GaMD simulations for free energy calculations?



Download **amber-gamd-tutorial.tgz** that includes the simulation input files and example outputs used for data analysis in this tutorial.

1. Install GaMD supported Amber

Note that GaMD will be fully supported in the official release of AMBER16 (**http://ambermd.org**). Prior to that, a patch is made available to provide temporary support of GaMD in AMBER14 (***amber14_GaMD_patch_confidential.txt***). Following are instructions for installing the GaMD patched AMBER14.

1) Obtain "AMBER 14 + AmberTools 15" (http://ambermd.org) and amber-gamd-tutorial.tgz

```
# export AMBERHOME to your amber home directory
export AMBERHOME=~/amber
export LD_LIBRARY_PATH=$AMBERHOME/lib:$LD_LIBRARY_PATH
export PATH=$AMBERHOME/bin:$PATH

# Uncompress the tutorial file
tar -xvzf amber-gamd-tutorial.tgz
```

2) Apply GaMD patch

```
cd amber-gamd-tutorial
cp –v amber14_GaMD_patch_confidential.txt $AMBERHOME
cd $AMBERHOME
./update_amber --update
patch -p0 < amber14_GaMD_patch_confidential.txt
```

3) Compile Amber code

 # pmemd:
cd $AMBERHOME
./configure gnu
cd $AMBERHOME/src/pmemd/src
make install

 # pmemd.MPI:
cd $AMBERHOME
./configure -mpi gnu
cd $AMBERHOME/src/pmemd/src
make parallel

 # pmemd.cuda:
cd $AMBERHOME
./configure -cuda gnu
cd $AMBERHOME/src/pmemd/src
make cuda

 # pmemd.cuda.MPI:
cd $AMBERHOME
./configure -cuda -mpi gnu
cd $AMBERHOME/src/pmemd/src
make cuda_parallel

We will mainly use the **pmemd.cuda** version to run GaMD simulations in this tutorial.

DISCLAIMER

GaMD patch for Amber 14 + AmberTools 15
Authors: Yinglong Miao & Ross C. Walker
Date: 29th July 2015
Programs: pmemd, pmemd.MPI, pmemd.cuda and pmemd.cuda.mpi
Description: This patch adds temporary GaMD support to AMBER 14.
It is built against AMBER 14 + AmberTools 15 + Amber 14 updates to 12.
This patch is provided for convenience prior to AMBER 16 release which
will feature GaMD support. It is made available by request only
and subject to your agreement not to redistribute this patch or to provide access to
a GaMD patched copy of AMBER without the written permission of
Ross C. Walker or Yinglong Miao.

By applying this patch you accept these terms and certify that you have
been granted permission to use this with Amber 14 + AmberTools 15.

2. Run GaMD simulations on Alanine Dipeptide

In this section, we will learn how to run GaMD simulations on alanine dipeptide with different dihedral-, total- and dual-boost potentials. To get started, let's set the following environmental variable:
export GaMDHOME=amber-gamd-tutorial

1) Run total-boost GaMD simulation

- **cd $GaMDHOME/test-dia/gamd-tot**
- **Input file: md.in**

```
&cntrl
  imin = 0, irest = 0, ntx = 1,
  nstlim = 1000, dt = 0.002,
  ntc = 2, ntf = 2, tol = 0.000001,
  iwrap = 1, ntb = 1, cut = 8.0,
  ntt = 3, temp0 = 300.0, tempi = 300.0,
  ntpr = 50, ntwx = 50, ntwr = 500,
  ntxo = 1, ioutfm = 1, ig = -1, ntwprt = 22,
  igamd = 1, iE = 1, irest_gamd = 0,
  ntcmd = 200, nteb = 200, ntave = 100,
  sigma0P = 6.0,
&end
```

- **Run the simulation using pmemd.cuda:**
  pmemd.cuda -O -i md.in -o md-1.out -p dip.top -c dip.crd -r md-1.rst -x md-1.nc

2) Run dihedral-boost GaMD simulation

- **cd $GaMDHOME/test-dia/gamd-dih**
- **Input file: md.in**

```
&cntrl
  imin = 0, irest = 0, ntx = 1,
  nstlim = 1000, dt = 0.002,
  ntc = 2, ntf = 2, tol = 0.000001,
  iwrap = 1, ntb = 1, cut = 8.0,
  ntt = 3, temp0 = 300.0, tempi = 300.0,
  ntpr = 50, ntwx = 50, ntwr = 500,
  ntxo = 1, ioutfm = 1, ig = -1, ntwprt = 22,
  igamd = 2, iE = 1, irest_gamd = 0,
  ntcmd = 200, nteb = 200, ntave = 100,
  sigma0D = 6.0,
&end
```

- **Run the simulation using pmemd.cuda:**
  pmemd.cuda -O -i md.in -o md-1.out -p dip.top -c dip.crd -r md-1.rst -x md-1.nc

3) Run dual-boost GaMD simulation

- **cd $GaMDHOME/test-dia/gamd-dual**
- **Input file: md.in**

  &cntrl
  imin = 0, irest = 0, ntx = 1,
  nstlim = 1000, dt = 0.002,
  ntc = 2, ntf = 2, tol = 0.000001,
  iwrap = 1, ntb = 1, cut = 8.0,
  ntt = 3, temp0 = 300.0, tempi = 300.0,
  ntpr = 50, ntwx = 50, ntwr = 500,
  ntxo = 1, ioutfm = 1, ig = -1, ntwprt = 22,
  igamd = 3, iE = 1, irest_gamd = 0,
  ntcmd = 200, nteb = 200, ntave = 100,
  sigma0P = 6.0, sigma0D = 6.0,
  &end
- **Run the simulation using pmemd.cuda:**

  pmemd.cuda -O -i md.in -o md-1.out -p dip.top -c dip.crd -r md-1.rst -x md-1.nc

4) Run Restart GaMD Simulation

- **cd $GaMDHOME/test-dia/gamd-dual**
- **Input file: md-restart.in**

  &cntrl
  imin = 0, irest = 0, ntx = 1,
  nstlim = 1000, dt = 0.002,
  ntc = 2, ntf = 2, tol = 0.000001,
  iwrap = 1, ntb = 1, cut = 8.0,
  ntt = 3, temp0 = 300.0, tempi = 300.0,
  ntpr = 50, ntwx = 50, ntwr = 500,
  ntxo = 1, ioutfm = 1, ig = -1, ntwprt = 22,
  igamd = 3, iE = 1, irest_gamd = 1,
  ntcmd = 0, nteb = 0, ntave = 100,
  sigma0P = 6.0, sigma0D = 6.0,
  &end
- **Run the simulation using pmemd.cuda:**

  pmemd.cuda -O -i md-restart.in -o md-2.out -p dip.top -c md-1.rst -r md-2.rst -x md-2.nc

5) Run long dual-boost GaMD simulation for free energy calculation

- **cd $GaMDHOME/test-dia/gamd-dual-30ns**
- **Input file: md.in**

  &cntrl
  imin = 0, irest = 0, ntx = 1,
  nstlim = 19000000, dt = 0.002,
  ntc = 2, ntf = 2, tol = 0.000001,
  iwrap = 1, ntb = 1, cut = 8.0,
  ntt = 3, temp0 = 300.0, tempi = 300.0,
  ntpr = 50, ntwx = 50, ntwr = 500,

ntxo = 1, ioutfm = 1, ig = -1, ntwprt = 22,
igamd = 3, iE = 1, irest_gamd = 0,
ntcmd = 1000000, nteb = 3000000, ntave = 50000,
sigma0P = 6.0, sigma0D = 6.0,
&end

- **Run the simulation using pmemd.cuda:**
  pmemd.cuda -O -i md.in -o md-1.out -p dip.top -c dip.crd -r md-1.rst -x md-1.nc

3. Simulation Analysis

In this section, we will learn how to analyze the simulation trajectories, particularly calculating dihedral angles in alanine dipeptide, and reweight GaMD simulations to calculate free energies of biomolecules.

1) Calculate dihedral angles in alanine dipeptide

cd $GaMDHOME/test-dia/gamd-dual/results

 # ./calc_dih.sh

```
echo "trajin md-1.nc
dihedral Phi :1@C  :2@N  :2@CA :2@C out Phi
dihedral Psi :2@N  :2@CA :2@C  :3@N out Psi" > dih-dia.ptraj
cpptraj dip_vac.top < dih-dia.ptraj
awk '{print $2}' Phi | tail -n 20 > Phi.dat
awk '{print $2}' Psi | tail -n 20 > Psi.dat
```

2) Reweight GaMD simulations to calculate free energies

According to our previous studies, when the boost potential follows Gaussian distribution as in GaMD, the Cumulant expansion to 2nd order (also referred to as "Gaussian Approximation") can be effectively used to approximate the ensemble-averaged reweighting factor:

$$\left\langle e^{\beta\Delta V}\right\rangle = \exp\left\{\sum_{k=1}^{\infty}\frac{\beta^k}{k!}C_k\right\},$$

where the first two order cumulants are given by:

$$C_1 = \left\langle \Delta V\right\rangle,$$

$$C_2 = \left\langle \Delta V^2\right\rangle - \left\langle \Delta V\right\rangle^2 = \sigma_{\Delta V}^2.$$

The Exponential Average and Maclaurin series expansion reweighting results are normally less accurate, but they are also made available in the following python codes for comparison

To characterize the extent to which ΔV follows Gaussian distribution, its distribution anharmonicity γ is calculated:

$$\gamma = S_{\max} - S_{\Delta V} = \frac{1}{2}\ln\left(2\pi e\sigma_{\Delta V}^2\right) + \int_0^{\infty} p\left(\Delta V\right)\ln\left(p\left(\Delta V\right)\right)d\Delta V,$$

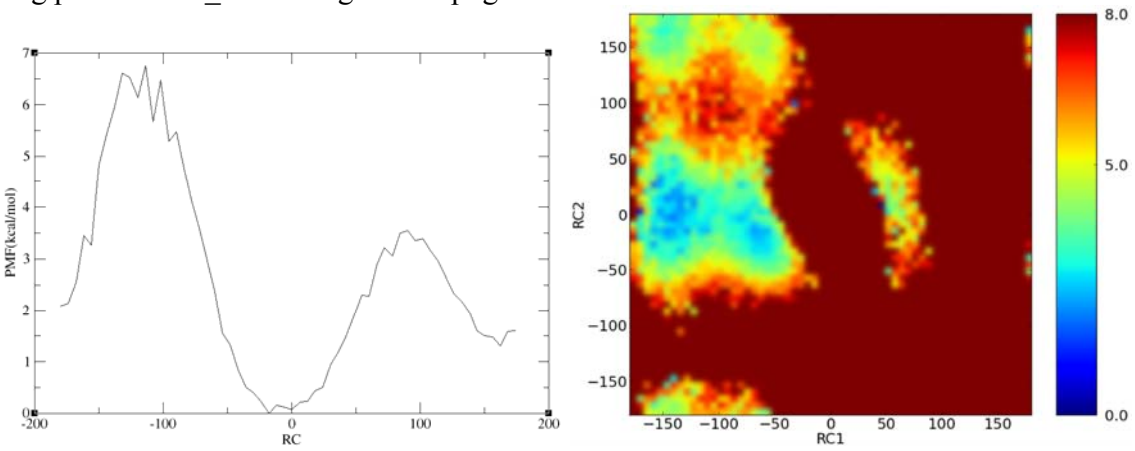
where ΔV is dimensionless as divided by $k_B T$ with k_B and T being the Boltzmann constant and system temperature, respectively, and $S_{\max} = \frac{1}{2}\ln\left(2\pi e\sigma_{\Delta V}^2\right)$ is the maximum entropy of ΔV. When γ is zero, ΔV follows exact Gaussian distribution with sufficient sampling. Reweighting by approximating the exponential average term with cumulant expansion to the second order is able to accurately recover the original free energy landscape. As γ increases, the ΔV distribution becomes less harmonic and the reweighted free energy profile obtained from cumulant expansion to the second order would deviate from the original.

cd $GaMDHOME/test-dia/gamd-dual-30ns/results
 # ./reweight.sh
Prepare input file "**weights.dat**" in the following format:
Column 1: dV in units of kbT; column 2: timestep; column 3: dV in units of kcal/mol
For AMBER14:
 # Ignore the *ntcmd* and *nteb* steps
nlines=300000 # number of data points used for reweighting
tail -n $nlines gamd.log | awk 'NR%1==0' | awk '{print ($8+$7)/(0.001987*300)" "
$2 " " ($8+$7)}' > weights.dat

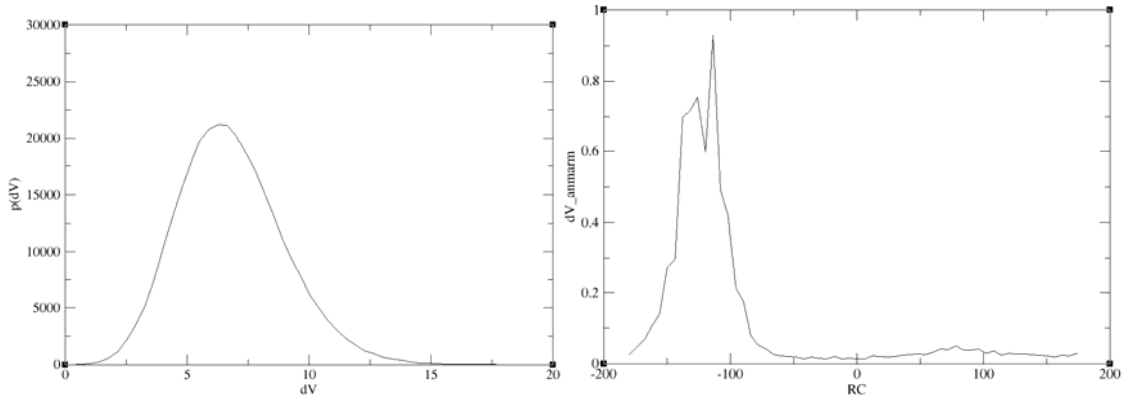
 # 1D data
Reweighting using cumulant expansion
python PyReweighting-1D.py -input Psi.dat -cutoff 10 -Xdim -180 180 -disc 6 -Emax 20 -job amdweight_CE -weight weights.dat | tee -a reweight_variable.log
…
Analyze boost potential distribution and anharmonicity
python PyReweighting-1D.py -input Psi.dat -cutoff 10 -Xdim -180 180 -disc 6 -Emax 20 -job amd_dV -weight weights.dat | tee -a reweight_variable.log

 # 2D data
Reweighting using cumulant expansion
python PyReweighting-2D.py -cutoff 10 -input Phi_Psi -Xdim -180 180 -discX 6 -Ydim -180 180 -discY 6 -Emax 20 -job amdweight_CE -weight weights.dat | tee -a reweight_variable.log
…
Analyze boost potential distribution and anharmonicity
python PyReweighting-2D.py -cutoff 10 -input Phi_Psi -Xdim -180 180 -discX 6 -Ydim -180 180 -discY 6 -Emax 20 -job amd_dV -weight weights.dat | tee -a reweight_variable.log

Visualize output files using xmgrace/gnuplot (linux), excel/originPro (windows) and other similar programs. Examine free energy profiles:
cd $GaMDHOME/test-dia/gamd-dual-30ns/results
xmgrace pmf-Psi-reweight-CE2.xvg &
eog pmf-2D-Phi_Psi-reweight-CE2.png &



Examine boost potential distribution & anharmonicity:
xmgrace dV-hist-Psi.dat.xvg &
xmgrace dV-anharm-Psi.dat.xvg &



Further Questions

If you find any bug or problem during running the GaMD simulations, please feel free to contact "Yinglong Miao <yimiao@ucsd.edu>" and "Ross C Walker <ross@rosswalker.co.uk>". You may also find the latest updates and simulation tips of GaMD at https://gamd.ucsd.edu.