Gaussian Accelerated Molecular Dynamics (GaMD)

1. Implementations of GaMD, LiGaMD and Pep-GaMD algorithms in Amber

GaMD: GaMD has been implemented in *pmemd,* both the serial and parallel versions on CPU (pmemd and pmemd.MPI) and GPU (pmemd.cuda and pmemd.cuda.MPI) by Yinglong Miao. Note that GaMD is not available in Sander. Similar to aMD, GaMD provides options to boost only the total potential energy (GaMD_Tot, *igamd*=1), only the dihedral potential energy (GaMD_Dih, *igamd*=2), both the total and dihedral potential energies (GaMD_Dual, *igamd*=3), the non-bonded potential energy (GaMD_NB, *igamd*=4) and both the non-bonded potential and dihedral energies (GaMD_Dual_NB, *igamd*=5). The dual-boost simulation generally provides higher acceleration than the other single-boost simulations for enhanced sampling. The simulation parameters comprise of settings for calculating the threshold energy values and the effective harmonic force constants of the boost potentials.

LiGaMD: LiGaMD has been implemented by Yinglong Miao in only the serial GPU version of *pmemd* (pmemd.cuda). It provides options to boost only non-bonded potential energy of the bound ligand (LiGaMD, *igamd*=10) and in addition the total system potential energy other than the non-bonded potential energy of bound ligand (LiGaMD_Dual, *igamd*=11). LiGaMD_Dual generally provides higher acceleration than LiGaMD for enhanced sampling. The simulation parameters comprise of settings for calculating the threshold energy values and the effective harmonic force constants of the boost potentials.

Pep-GaMD: Pep-GaMD has been implemented by Jinan Wang in only the serial GPU version of *pmemd* (pmemd.cuda). It provides options to boost only the peptide potential energy (Pep-GaMD, *igamd*=14) and in addition the total system potential energy other than the peptide potential energy (Pep-GaMD_Dual, *igamd*=15). Pep-GaMD_Dual generally provides higher acceleration than Pep-GaMD for enhanced sampling. The simulation parameters comprise of settings for calculating the threshold energy values and the effective harmonic force constants of the boost potentials.

PPI-GaMD: PPI-GaMD has been implemented by Jinan Wang in only the serial GPU version of *pmemd* (pmemd.cuda). It provides options to boost only the protein interaction energy (PPI-GaMD, *igamd*=16) and in addition the total system potential energy other than the protein interaction energy (PPI-GaMD_Dual, *igamd*=17). PPI-GaMD_Dual generally provides higher acceleration than PPI-GaMD for enhanced sampling. The simulation parameters comprise of settings for calculating the threshold energy values and the effective harmonic force constants of the boost potentials.

All the information generated by GaMD simulations, necessary for reweighing is stored at each step into a vector which is flushed to a log file (*gamd.log* by default) every time the coordinates are written to disk, i.e. every *ntwx* steps. The name of the log file can be set to a user defined name by using the command line option -gamdlog when running Amber. Additional parameters are specified by the following variables:

igamd Flag to apply boost potential

= 0 (default) no boost is applied

= 1 boost on the total potential energy only

= 2 boost on the dihedral energy only

= 3 dual boost on both dihedral and total potential energy

= 4 boost on the non-bonded potential energy only

= 5 dual boost on both dihedral and non-bonded potential energy

= 10 boost on non-bonded potential energy of selected region (defined by timask1 and scmask1) as for a ligand (LiGaMD)

= 11 dual boost on both non-bonded potential energy of the bound ligand and remaining potential energy of the rest of the system (LiGaMD_Dual)

= 14 boost on the total potential energy of selected region (defined by timask1 and scmask1) as for a peptide (Pep-GaMD)

= 15 dual boost on both the peptide potential energy and the total system potential energy other than the peptide potential energy (Pep-GaMD_Dual)

= 16 boost on the interaction between protein partners (The first protein is defined by timask1 and scmask1 and the second one defined by bgpro2atm (first atom number of the protein) and edpro2atm (the end atom number of the protein)) for protein-protein interaction GaMD (PPI-GaMD)

= 17 dual boost on both the protein-protein interactions and the remaining potential

iE Flag to set the threshold energy E for applying all boost potentials

= 1 (default) set the threshold energy to the lower bound $E = V_{max}$

= 2 set the threshold energy to the upper bound $E = V_{min} + (V_{max} - V_{min})/k_0$

iEP Flag to overwrite ***iE*** and set the threshold energy E for applying the first boost potential in dual-boost schemes

= 1 (default) set the threshold energy to the lower bound $E = V_{max}$

= 2 set the threshold energy to the upper bound $E = V_{min} + (V_{max} - V_{min})/k_0$

iED Flag to overwrite ***iE*** and set the threshold energy E for applying the second boost potential in dual-boost schemes

= 1 (default) set the threshold energy to the lower bound $E = V_{max}$

= 2 set the threshold energy to the upper bound $E = V_{min} + (V_{max} - V_{min})/k_0$

ntcmdprep The number of preparation conventional molecular dynamics steps. This is used for system equilibration and the potential energies are not collected for calculating their statistics. The default is 200,000 for a simulation with 2 *fs* timestep.

ntcmd The number of initial conventional molecular dynamics simulation steps. Potential energies are collected between ***ntcmdprep*** and ***ntcmd*** to calculate their maximum, minimum, average and standard deviation (V_{max}, V_{min}, V_{avg}, σ_V). The default is 1,000,000 for a simulation with 2 *fs* timestep.

ntebprep The number of preparation biasing molecular dynamics simulation steps. This is used for system

equilibration after adding the boost potential and the potential statistics (V_{max}, V_{min}, V_{avg}, σ_V) are not updated during these steps. The default is 200,000 for a simulation with 2 *fs* timestep.

nteb The number of biasing molecular dynamics simulation steps. Potential statistics (V_{max}, V_{min}, V_{avg}, σ_V) are updated between the ***ntebprep*** and ***nteb*** steps and used to calculate the GaMD acceleration parameters, particularly E and k_0. The default is 1,000,000 for a simulation with 2 *fs* timestep. A greater value may be needed to ensure that the potential statistics and GaMD acceleration parameters level off before running production simulation between the ***nteb*** and ***nstlim*** (total simulation length) steps. Moreover, ***nteb*** can be set to ***nstlim***, by which the potential statistics and GaMD acceleration parameters are updated adaptively throughout the simulation. This in some cases provides more appropriate acceleration.

ntave The number of simulation steps used to calculate the average and standard deviation of potential energies. This variable has already been used in Amber. The default is set to 50,000 for GaMD simulations. It is recommended to be updated as about 4 times of the total number of atoms in the system. Note that ***ntcmd*** and ***nteb*** need to be multiples of ***ntave***.

irest_gamd Flag to restart GaMD simulation

= 0 (default) new simulation. A file "gamd-restart.dat" that stores the maximum, minimum, average and standard deviation of the potential energies needed to calculate the boost potentials (depending on the ***igamd*** flag) will be saved automatically after GaMD equilibration stage.

= 1 restart simulation (***ntcmd*** and ***nteb*** are set to 0 in this case). The "gamd-restart.dat" file will be read for restart.

sigma0P The upper limit of the standard deviation of the first potential boost that allows for accurate reweighting. The default is 6.0 (unit: kcal/mol).

sigma0D The upper limit of the standard deviation of the second potential boost that allows for accurate reweighting in dual-boost simulations (e.g., ***igamd*** = 2, 3, 5, 11 and 15). The default is 6.0 (unit: kcal/mol).

timask1 Specifies atoms of the first (bound) ligand or peptide in ambmask format when ***igamd*** = 10, 11, 14 or 15. The default is an empty string.

scmask1 Specifies atoms of the first (bound) ligand that will be described using soft core in ambmask format in LiGaMD when ***igamd*** = 10 or 11. In Pep-GaMD with ***igamd*** = 14 or 15, this flag was only used to specify atoms of peptide in ambmask format, but the peptide atoms will not be described using soft core. The default is an empty string.

nlig The total number of ligand molecules in the system. The default is 0.

ibblig The flag to boost the bound ligand selectively with ***nlig*** > 1

= 0 (default) no selective boost

= 1 boost the bound ligand selectively out of ***nlig*** ligand molecules in the system based on the shortest distance to the protein target site

= 2 boost the bound ligand selectively out of ***nlig*** ligand molecules in the system based on the smallest mean-square displacement (MSD)

atom_p Serial number of a protein atom (starting from 1 for the first protein atom) used to calculate the ligand

distance. It is used only when ***ibblig*** = 1. The default is 0.

atom_l Serial number of a ligand atom (starting from 1 for the first ligand atom) used to calculate the ligand distance to the protein. It is used only when ***ibblig*** = 1 or 2. The default is 0.

ntmsd Number of timesteps corresponding to the lagging time used to calculate the ligand MSD. It is used only when ***ibblig*** = 2. The default is 50,000.

nftau Number of saved simulation frames used to calculate the ligand MSD. MSD is calculated for every time window of *ntwin* = ***ntmsd*** + ***nftau*ntwx*** steps, for which simulation frames are saved every ***ntwx*** steps. It is used only when ***ibblig*** = 2. The default is 10.

dblig **(Optional)** The cutoff distance between atoms ***atom_p*** and ***atom_l*** used to identify the ligand that is bound in the protein when ***ibblig*** = 1 or the cutoff MSD of atom ***atom_l*** used to identify the ligand that is bound in the protein when ***ibblig*** = 2. If ***dblig*** (default 1.0d99 Å) is not set in the input file, the first boost potential will be selectively applied to the ligand with the smallest distance to the protein (***ibblig*** = 1) or the smallest MSD (***ibblig*** = 2) out of ***nlig*** ligand molecules in the system.

bgpro2atm Start atomic number of the second protein.

edpro2atm The final atomic number of the second protein.

The GaMD algorithm is summarized as the following:

```
GaMD {
  If (irest_gamd == 0) then
    For i = 1, …, ntcmd // run initial conventional molecular dynamics
      If (i >= ntcmdprep) Update Vmax, Vmin
      If (i >= ntcmdprep && i%ntave ==0) Update Vavg, sigmaV
    End
    Save Vmax,Vmin,Vavg,sigmaV to “gamd_restart.dat” file
    Calc_E_k0(iE,sigma0,Vmax,Vmin,Vavg,sigmaV)

    For i = ntcmd+1, …, ntcmd+nteb // Run biasing molecular dynamics simulation steps
      deltaV = 0.5*k0*(E-V)**2/(Vmax-Vmin)
      V = V + deltaV
      If (i >= ntcmd+ntebprep) Update Vmax, Vmin
      If (i >= ntcmd+ntebprep && i%ntave ==0) Update Vavg, sigmaV
      Calc_E_k0(iE,sigma0,Vmax,Vmin,Vavg,sigmaV)
    End
    Save Vmax,Vmin,Vavg,sigmaV to “gamd_restart.dat” file
  else if (irest_gamd == 1) then
    Read Vmax,Vmin,Vavg,sigmaV from “gamd_restart.dat” file
```

```
    End if

    For i = ntcmd+nteb+1, …, nstlim // run production simulation
        deltaV = 0.5*k0*(E-V)**2/(Vmax-Vmin)
        V = V + deltaV
    End
}

Subroutine Calc_E_k0(iE,sigma0,Vmax,Vmin,Vavg,sigmaV) {
if iE = 1 :
    E = Vmax
    k0' = (sigma0/sigmaV) * (Vmax-Vmin)/(Vmax-Vavg)
    k0 = min(1.0, k0')
else if iE = 2 :
    k0" = (1-sigma0/sigmaV) * (Vmax-Vmin)/(Vavg-Vmin)
    if 0 < k0" <= 1 :
            k0 = k0"
            E = Vmin + (Vmax-Vmin)/k0
    else
            E = Vmax
            k0' = (sigma0/sigmaV) * (Vmax-Vmin)/(Vmax-Vavg)
            k0 = min(1.0, k0')
    end
end
}
```

The LiGaMD algorithm is summarized as the following:

LiGaMD {

```
  If (irest_gamd == 0) then
    For i = 1, …, ntcmd // run initial conventional molecular dynamics
      If (i >= ntcmdprep) Update Vmax, Vmin
      If (i >= ntcmdprep && i%ntave ==0) Update Vavg, sigmaV
    End
    Save Vmax,Vmin,Vavg,sigmaV to "gamd_restart.dat" file
    Calc_E_k0(iE,sigma0,Vmax,Vmin,Vavg,sigmaV)
```

```
  For i = ntcmd+1, …, ntcmd+nteb // Run biasing molecular dynamics simulation steps
      deltaV = 0.5*k0*(E-V)**2/(Vmax-Vmin)
      V = V + deltaV
      If (i >= ntcmd+ntebprep) Update Vmax, Vmin
      If (i >= ntcmd+ntebprep && i%ntave ==0) Update Vavg, sigmaV
      Calc_E_k0(iE,sigma0,Vmax,Vmin,Vavg,sigmaV)
  End
  Save Vmax,Vmin,Vavg,sigmaV to "gamd_restart.dat" file
else if (irest_gamd == 1) then
  Read Vmax,Vmin,Vavg, sigmaV from "gamd_restart.dat" file
End if

For i = ntcmd+nteb+1, …, nstlim // run production simulation
  deltaV = 0.5*k0*(E-V)**2/(Vmax-Vmin)
  V = V + deltaV
End

ntwin = ntmsd+nftau*ntwx
lig0=1 // ID of the bound ligand

If (ibblig == 1 && i%ntwx ==0) then // identify the bound ligand according to the distance to protein
    For ilig = 1, …, nlig
      dlig = distance(atom_p, atom_l)
      If (dmin <= dlig) blig_min=ilig; dmin=dlig
    End
    If (dmin <= dblig) blig=blig_min
else if (ibblig == 2 && i%ntwin ==0) then // identify the bound ligand according to MSD
    For ilig = 1, …, nlig
      dlig = msd(atom_l, ntmsd, nftau)
      If (dmin <= dlig) blig_min=ilig; dmin=dlig
    End
    If (dmin <= dblig) blig=blig_min
End if
If (blig != lig0) Swap atomic coordinates, forces and velocities of ligand blig with lig0 for selective higher boost
}
```

The Pep-GaMD algorithm is summarized as the following:

Pep-GaMD {

```
If (irest_gamd == 0) then
  For i = 1, …, ntcmd // run initial conventional molecular dynamics
    If (i >= ntcmdprep) Update Vmax, Vmin
    If (i >= ntcmdprep && i%ntave ==0) Update Vavg, sigmaV
  End
  Save Vmax,Vmin,Vavg,sigmaV to “gamd_restart.dat” file
  Calc_E_k0(iE,sigma0,Vmax,Vmin,Vavg,sigmaV)

  For i = ntcmd+1, …, ntcmd+nteb // Run biasing molecular dynamics simulation steps
      deltaV = 0.5*k0*(E-V)**2/(Vmax-Vmin)
      V = V + deltaV
      If (i >= ntcmd+ntebprep) Update Vmax, Vmin
      If (i >= ntcmd+ntebprep && i%ntave ==0) Update Vavg, sigmaV
      Calc_E_k0(iE,sigma0,Vmax,Vmin,Vavg,sigmaV)
  End
  Save Vmax,Vmin,Vavg,sigmaV to “gamd_restart.dat” file
else if (irest_gamd == 1) then
  Read Vmax,Vmin,Vavg, sigmaV from “gamd_restart.dat” file
End if


For i = ntcmd+nteb+1, …, nstlim // run production simulation
  deltaV = 0.5*k0*(E-V)**2/(Vmax-Vmin)
  V = V + deltaV
End
```

}

The PPI-GaMD algorithm is summarized as the following:

PPI-GaMD {

```
If (irest_gamd == 0) then
  For i = 1, …, ntcmd // run initial conventional molecular dynamics
    If (i >= ntcmdprep) Update Vmax and Vmin of interaction potential energy
    If (i >= ntcmdprep && i%ntave ==0) Update Vavg and sigmaV of interaction potential energy
  End
  Save Vmax,Vmin,Vavg,sigmaV of interaction potential energy to “gamd_restart.dat” file
  Calc_E_k0(iE,sigma0,Vmax,Vmin,Vavg,sigmaV)
```

```
  For i = ntcmd+1, …, ntcmd+nteb // Run biasing molecular dynamics simulation steps
      deltaV = 0.5*k0*(E-V)**2/(Vmax-Vmin)
      V = V + deltaV
      If (i >= ntcmd+ntebprep) Update Vmax and Vmin of interaction potential energy
      If (i >= ntcmd+ntebprep && i%ntave ==0) Update Vavg and sigmaV of interaction potential energy
      Calc_E_k0(iE,sigma0,Vmax,Vmin,Vavg,sigmaV)
  End
  Save Vmax,Vmin,Vavg and sigmaV of of interaction potential energy to "gamd_restart.dat" file
else if (irest_gamd == 1) then
  Read Vmax,Vmin,Vavg and sigmaV of interaction potential energy from "gamd_restart.dat" file
End if


For i = ntcmd+nteb+1, …, nstlim // run production simulation
  deltaV = 0.5*k0*(E-V)**2/(Vmax-Vmin)
  V = V + deltaV
End
}
```

2. Sample input parameters for GaMD simulation algorithms

Example input parameters used in GaMD_Dual simulations include the following in addition to those used in conventional MD:

```
igamd = 3, iE = 1, irest_gamd = 0,
ntcmd = 1000000, nteb = 1000000, ntave = 50000,
ntcmdprep = 200000, ntebprep = 200000,
sigma0P = 6.0, sigma0D = 6.0,
```

Example input parameters used in LiGaMD_Dual simulations include the following in addition to those used in conventional MD:

```
igamd = 11, irest_gamd = 0,
ntcmd = 700000, nteb = 27300000, ntave = 140000,
ntcmdprep = 280000, ntebprep = 280000,
sigma0P = 4.0, sigma0D = 6.0, iEP = 2, iED=1,

icfe = 1, ifsc = 1, gti_cpu_output = 0, gti_add_sc = 1,
timask1 = ':225', scmask1 = ':225',
timask2 = '', scmask2 = '',
```

ibblig = 1, nlig = 10, atom_p = 2472, atom_l = 4,

OR

igamd = 11, irest_gamd = 0,
ntcmd = 700000, nteb = 27300000, ntave = 140000,
ntcmdprep = 280000, ntebprep = 280000,
sigma0P = 4.0, sigma0D = 6.0, iEP = 2, iED=1,

icfe = 1, ifsc = 1, gti_cpu_output = 0, gti_add_sc = 1,
timask1 = ':225', scmask1 = ':225',
timask2 = '', scmask2 = '',

ibblig = 2, nlig = 10, atom_l = 4,
ntmsd = 50000, nftau = 10,

Example input parameters used in Pep-GaMD_Dual simulations include the following in addition to those used in conventional MD:

icfe = 1, ifsc = 1,gti_cpu_output = 0,gti_add_sc = 1,
timask1 = ':1-3', scmask1 = ':1-3',
timask2 = '', scmask2 = '',

igamd = 15, iE = 1, iEP = 1, iED = 1, irest_gamd = 0,
ntcmd = 1000000, nteb = 1000000, ntave = 50000,
ntcmdprep = 200000, ntebprep = 200000,
sigma0P = 6.0, sigma0D = 6.0,

Example input parameters used in PPI-GaMD_Dual simulations include the following in addition to those used in conventional MD:

icfe = 1, ifsc = 1, gti_cpu_output = 0,gti_add_sc = 1,
timask1 = ':1-110', scmask1 = ':1-110',
timask2 = '', scmask2 = '',
bgpro2atm=1, edpro2atm=1453,
igamd = 17, iEP = 2, iED = 1, irest_gamd = 0,
ntcmd = 1000000, nteb = 1000000, ntave = 50000,
ntcmdprep = 200000, ntebprep = 200000,
sigma0P = 6.0, sigma0D = 6.0,

3. Further information

Test cases for running GaMD have been included into the distribution of Amber. The latest updates, examples and simulation tips can be found on the GaMD website. A tutorial based on a study we performed on alanine dipeptide, demonstrating the usage of GaMD on unconstrained enhanced sampling and free energy calculation of biomolecules is also available on the GaMD website.

Energetic reweighting: A toolkit of python scripts "PyReweighting" has been developed to facilitate reweighting analysis of aMD and GaMD simulations. PyReweighting implements a list of commonly used reweighting methods, including (1) exponential average that reweights trajectory frames by the Boltzmann factor of the boost potential and then calculates the ensemble average for each bin, (2) Maclaurin series expansion that approximates the exponential Boltzmann factor, and (3) cumulant expansion that expresses the reweighting factor as summation of boost potential cumulants. Notably, MacLaurin series expansion is equivalent to cumulant expansion on the first order. Cumulant expansion to the 2nd order ("Gaussian approximation") normally provides the most accurate reweighting results. The PyReweighting scripts and tutorial can be downloaded at: https://github.com/MiaoLab20/pyreweighting.

Kinetic reweighting: Reweighting of biomolecular kinetics from GaMD simulations can be obtained by applying Kramers rate theory. The curvatures and energy barriers of the reweighted and modified free energy profiles, as well as the apparent diffusion coefficients, are calculated and used in Kramers' rate equation to determine accelerations of biomolecular kinetics and recover the original biomolecular kinetic rate constants from the GaMD simulations. In addition to "PyReweighting" that facilitates calculations of free energy profiles, a Smoluchowski equation solver coded in C++ ("smol_solver" shared by Prof. Donald Hamelberg) can be used to calculate kinetic rates across PMF free energy barriers as needed to estimate the apparent diffusion coefficients. The source code and test examples, along with compiling and usage instructions included in a README file can be downloaded at: https://github.com/MiaoLab20/smol_solver.