Gaussian Accelerated Molecular Dynamics (GaMD)

1. Gaussian Accelerated Molecular Dynamics (GaMD)

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. The boost potential follows Gaussian distribution, which allows for accurate reweighting using cumulant expansion to the second order. In a previous study[1], GaMD has been demonstrated on simulations of 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 help us to identify distinct low energy states of the biomolecules and characterize the protein folding and ligand binding pathways quantitatively.

Consider a system with *N* atoms at positions $r \equiv \{\vec{r}_1, \cdots, \vec{r}_N\}$. When the system potential $V(r)$ is lower than a reference energy *E*, the modified potential $V^*(r)$ of the system is calculated as:

$$V^*(r) = V(r) + \Delta V(r),$$

$$\Delta V(r) = \begin{cases} \frac{1}{2}k\big(E - V(r)\big)^2, & V(r) < E \\ 0, & V(r) \geq E \end{cases} \quad (1)$$

where *k* is the harmonic force constant. The two adjustable parameters *E* and *k* are automatically determined based on three enhanced sampling principles. The reference energy needs to be set in the following range:

$$V_{max} \leq E \leq V_{min} + \frac{1}{k}, \quad (2)$$

where V_{max} and V_{min} are the system minimum and maximum potential energies. To ensure that **Eqn. (2)** is valid, *k* has to satisfy: $k \leq \frac{1}{V_{max} - V_{min}}$ Let us define $k \equiv k_0 \cdot \frac{1}{V_{max} - V_{min}}$, then $0 < k_0 \leq 1$. The standard deviation of ΔV needs to be small enough (i.e., narrow distribution) to ensure proper energetic reweighting: $\sigma_{\Delta V} = k\big(E - V_{avg}\big)\sigma_V \leq \sigma_0$ where V_{avg} and σ_V are the average and standard deviation of the system potential energies, $\sigma_{\Delta V}$ is the standard deviation of ΔV with σ_0 as a user-specified upper limit (e.g., $10k_BT$) for proper reweighting. When *E* is set to the lower bound $E=V_{max}$, k_0 can be calculated as:

$$k_0 = \min(1.0, k_0') = \min\left(1.0, \frac{\sigma_0}{\sigma_V} \cdot \frac{V_{max} - V_{min}}{V_{max} - V_{avg}}\right). \quad (3)$$

Alternatively, when the threshold energy *E* is set to its upper bound $E = V_{min} + \frac{1}{k}$, k_0 is set to:

$$k_0 = k_0'' \equiv \left(1 - \frac{\sigma_0}{\sigma_V}\right)\frac{V_{max} - V_{min}}{V_{avg} - V_{min}}, \quad (4)$$

if k_0'' is found to be between *0* and *1*. Otherwise, k_0 is calculated using **Eqn. (3)**.

2. Ligand Gaussian Accelerated Molecular Dynamics (LiGaMD)

Based on GaMD, a new algorithm called ligand GaMD or "LiGaMD" has been developed to simulate ligand binding and unbinding[2]. It works by selectively boosting the ligand non-bonded interaction potential energy. Another boost potential could be applied to the remaining potential energy of the entire system in a dual-boost algorithm (LiGaMD_Dual) to facilitate ligand binding. LiGaMD has been demonstrated on host-guest and protein-ligand binding model systems. Repetitive guest binding and unbinding in the β-cyclodextrin host were observed in hundreds-of-nanosecond LiGaMD simulations. The calculated binding free energies of guest molecules with sufficient sampling agreed excellently with experimental data (< 1.0 kcal/mol error). In comparison with previous microsecond-timescale conventional molecular dynamics simulations, accelerations of ligand kinetic rate constants in LiGaMD simulations were properly estimated using Kramers' rate theory. Furthermore, LiGaMD allowed us to capture repetitive dissociation and binding of the benzamidine inhibitor in trypsin within 1 μs simulations. The calculated ligand binding free energy and kinetic rate constants compared well with the experimental data. Therefore, LiGaMD provides a promising approach for characterizing ligand binding thermodynamics and kinetics simultaneously.

In LiGaMD, we consider a system of ligand *L* binding to a protein *P* in a biological environment *E*. We decompose the potential energy into the following terms:

$$\begin{aligned} V(r) = {} & V_{P,b}(r_P) + V_{L,b}(r_L) + V_{E,b}(r_E) \\ & + V_{PP,nb}(r_P) + V_{LL,nb}(r_L) + V_{EE,nb}(r_E) \\ & + V_{PL,nb}(r_{PL}) + V_{PE,nb}(r_{PE}) + V_{LE,nb}(r_{LE}). \end{aligned} \quad (1)$$

where $V_{P,b}$, $V_{L,b}$ and $V_{E,b}$ are the bonded potential energies in protein *P*, ligand *L* and environment *E*, respectively. $V_{PP,nb}$, $V_{LL,nb}$ and $V_{EE,nb}$ are the self non-bonded potential energies in protein *P*, ligand *L* and environment *E*, respectively. $V_{PL,nb}$, $V_{PE,nb}$ and $V_{LE,nb}$ are the non-bonded interaction energies between *P*-*L*, *P*-*E* and *L*-*E*, respectively. According to classical molecular mechanics force fields, the non-bonded potential energies are usually calculated as:

$$V_{nb} = V_{elec} + V_{vdW}. \quad (2)$$

Where V_{elec} and V_{vdW} are the system electrostatic and van der Waals potential energies. Presumably, ligand binding mainly involves the non-bonded interaction energies of the ligand, $V_{L,nb}(r) = V_{LL,nb}(r_L) + V_{PL,nb}(r_{PL}) + V_{LE,nb}(r_{LE})$. Therefore, we add a boost potential selectively to the ligand non-bonded potential energy according to the GaMD algorithm:

$$\Delta V_{L,nb}(r) = \begin{cases} \frac{1}{2}k_{L,nb}\left(E_{L,nb} - V_{L,nb}(r)\right)^2, & V_{L,nb}(r) < E_{L,nb} \\ 0, & V_{L,nb}(r) \geq E_{L,nb} \end{cases} \quad (3)$$

where $E_{L,nb}$ is the threshold energy for applying boost potential and $k_{L,nb}$ is the harmonic constant.

Next, one can add multiple ligand molecules in the solvent to facilitate ligand binding to proteins in MD simulations. This is based on the fact that the ligand binding rate constant k_{on} is inversely proportional to the ligand concentration. The higher the ligand concentration, the faster the ligand binds, provided that the ligand concentration is still within its solubility limit. In addition to selectively boosting the bound ligand, another boost potential could thus be applied on the unbound ligand molecules, protein and solvent to facilitate both ligand dissociation and rebinding. The second boost potential is calculated using the total system potential energy other than the non-bonded potential energy of the bound ligand as:

$$\Delta V_D(r) = \begin{cases} \frac{1}{2}k_D\left(E_D - V_D(r)\right)^2, & V_D(r) < E_D \\ 0, & V_D(r) \geq E_D \end{cases} \quad (4)$$

Where V_D is the total system potential energy other than the non-bonded potential energy of the bound ligand, E_D is the corresponding threshold energy for applying the second boost potential and k_D is the harmonic constant. This leads to dual-boost LiGaMD (LiGaMD_Dual) with the total boost potential $\Delta V(r) = \Delta V_{L,nb}(r) + \Delta V_D(r)$.

3. Peptide Gaussian Accelerated Molecular Dynamics (Pep-GaMD)

Peptides often undergo large conformational changes during binding to the target proteins, being distinct from small-molecule ligand binding or protein-protein interactions. We have developed another algorithm called peptide GaMD or "Pep-GaMD" that enhances sampling of peptide-protein interactions[3].

In Pep-GaMD, we consider a system of ligand peptide L binding to a target protein P in a biological environment E. We decompose the potential energy into the following terms:

$$\begin{aligned} V(r) = {} & V_{P,b}(r_P) + V_{L,b}(r_L) + V_{E,b}(r_E) \\ & + V_{PP,nb}(r_P) + V_{LL,nb}(r_L) + V_{EE,nb}(r_E) \\ & + V_{PL,nb}(r_{PL}) + V_{PE,nb}(r_{PE}) + V_{LE,nb}(r_{LE}). \end{aligned} \quad (1)$$

where $V_{P,b}$, $V_{L,b}$ and $V_{E,b}$ are the bonded potential energies in protein P, peptide L and environment E, respectively. $V_{PP,nb}$, $V_{LL,nb}$ and $V_{EE,nb}$ are the self non-bonded potential energies in protein P, peptide L and environment E, respectively. $V_{PL,nb}$, $V_{PE,nb}$ and $V_{LE,nb}$ are the non-bonded interaction energies between P-L, P-E and L-E, respectively.

Presumably, peptide binding mainly involves in both the bonded and non-bonded interaction energies of the peptide since peptides often undergo large conformational changes during binding to the target proteins. Thus, the essential peptide potential energy is $V_L(r) = V_{LL,b}(r_L) + V_{LL,nb}(r_L) + V_{PL,nb}(r_{PL}) + V_{LE,nb}(r_{LE})$. In Pep-GaMD, we add boost potential selectively to the essential peptide potential energy according to the GaMD algorithm:

$$\Delta V_L(r) = \begin{cases} \frac{1}{2}k_L\left(E_L - V_L(r)\right)^2, & V_L(r) < E_L \\ 0, & V_L(r) \geq E_L \end{cases} \quad (2)$$

where E_L is the threshold energy for applying boost potential and k_L is the harmonic constant.

In addition to selectively boosting the peptide, another boost potential is applied on the protein and solvent to enhance conformational sampling of the protein and facilitate peptide rebinding. The second boost potential is calculated using the total system potential energy other than the peptide potential energy as:

$$\Delta V_D(r) = \begin{cases} \frac{1}{2}k_D\left(E_D - V_D(r)\right)^2, & V_D(r) < E_D \\ 0, & V_D(r) \geq E_D \end{cases} \quad (3)$$

Where V_D is the total system potential energy other than the peptide potential energy, E_D is the corresponding threshold energy for applying the second boost potential and k_D is the harmonic constant. This leads to dual-boost Pep-GaMD (Pep-GaMD_Dual) with the total boost potential $\Delta V(r) = \Delta V_L(r) + \Delta V_D(r)$.

4. Energetic Reweighting using Cumulant Expansion to the Second Order (Gaussian Approximation)

For energetic reweighting of GaMD simulations to calculate potential of mean force (PMF), the probability distribution along a reaction coordinate is written as $p^*(A)$. Given the boost potential $\Delta V(r)$ of each frame, $p^*(A)$ can be reweighted to recover the canonical ensemble distribution, $p(A)$, as:

$$p(A_j) = p^*(A_j)\frac{\langle e^{\beta\Delta V(r)}\rangle_j}{\sum_{i=1}^{M}\langle p^*(A_i)e^{\beta\Delta V(r)}\rangle_i}, \; j = 1, \dots, M, \quad (1)$$

where M is the number of bins, $\beta = k_BT$ and $\langle e^{\beta\Delta V(r)}\rangle_j$ is the ensemble-averaged Boltzmann factor of $\Delta V(r)$ for simulation frames found in the j^{th} bin. The ensemble-averaged reweighting factor can be approximated using cumulant expansion:

$$\langle e^{\beta\Delta V(r)}\rangle = exp\left\{\sum_{k=1}^{\infty}\frac{\beta^k}{k!}C_k\right\}, \quad (2)$$

where the first two cumulants are given by:

$$\begin{aligned} C_1 &= \langle\Delta V\rangle, \\ C_2 &= \langle\Delta V^2\rangle - \langle\Delta V\rangle^2 = \sigma_v^2. \end{aligned} \quad (3)$$

The boost potential obtained from GaMD simulations usually follows near-Gaussian distribution. Cumulant expansion to the second order thus provides a good approximation for computing the reweighting factor. The reweighted free energy $F(A) = -k_BT\ln p(A)$ is calculated as:

$$F(A) = F^*(A) - \sum_{k=1}^{2}\frac{\beta^k}{k!}C_k + F_c, \quad (4)$$

where $F^*(A) = -k_BT\ln p^*(A)$ is the modified free energy obtained from GaMD simulation and F_c is a constant.

5. Kinetic reweighting of GaMD simulations with Kramers' Rate Theory

Reweighting of biomolecular kinetics from GaMD simulations can be obtained by applying Kramers rate theory[2, 4]. For a particle climbing over potential energy barriers, Kramers showed that the reaction rate depends on temperature and viscosity of the host medium. The reaction rates were derived for both limiting cases of small and large viscosity. In the context of biomolecular simulations in aqueous medium, it is relevant for us to focus on the large viscosity limiting case. Biomolecules move in the high friction ("overdamping") regime and energy barriers are much greater than k_BT (k_B is the Boltzmann's constant and T is temperature). In this case, the reaction rate is calculated as:

$$k_R \cong \frac{2\pi w_m w_b}{\xi} e^{-\Delta F/k_B T}, \quad (1)$$

where w_m and w_b are frequencies of the approximated harmonic oscillators (also referred to as curvatures of free energy surface) near the energy minimum and barrier, respectively, ξ is the apparent friction coefficient and ΔF is the free energy barrier of transition.

Without the loss of generality, we consider a 1D potential of mean force (PMF) free energy profile of a reaction coordinate $F(A)$. Near minimum at A_m, the free energy can be approximated by a harmonic oscillator of frequency w_m, i.e., $F(A) = \frac{1}{2}(2\pi w_m)^2(A - A_m)^2$. Near barrier at A_b, the free energy is approximated as $F(A) = F_b - \frac{1}{2}(2\pi w_b)^2(A - A_b)^2$, where F_b is the free energy at A_b and w_b is the frequency of the approximated harmonic oscillator. Then we can calculate w_m and w_b as:

$$w = \sqrt{\frac{|F"(A)|}{2\pi}}, \quad (2)$$

where $F"(A)$ is the second-order derivative of the PMF profile.

The apparent friction coefficient ξ or diffusion coefficient D with $\xi = k_BT/D$ can be estimated as follows. First, we calculate a survival function $S(t)$ as the probability that the system remains in an energy well longer than time t. In a direct approach[5], we count the events that the system visits the energy well throughout a simulation. We record and measure the time intervals of each visiting event until the system escapes over an energy barrier. Then we have a time series T_i, where i=1, 2, …, N, and N is the total number barrier transitions observed in the simulation. The time series is subsequently ordered such that $\hat{T}_1 \le \hat{T}_2 \le \cdots \le \hat{T}_N$. With that, the survival function is estimated as

$S(\hat{T}_i) \approx 1 - i/N$, which is the probability that the system is trapped in the energy well for time longer than $\hat{T}_i$. Alternatively, we can numerically calculate the time-dependent probability density of reaction coordinate A, $\rho(A,t)$ by solving the Smoluchowski equation along 1D PMF profile of the reaction coordinate:

$$\frac{\partial \rho(A,t)}{\partial t} = D\frac{\partial}{\partial A}\left[e^{-F(A)/k_BT}\frac{\partial}{\partial A}\left(e^{F(A)/k_BT}\rho\right)\right]. \quad (3)$$

Then the survival function is calculated as $S(t) = \int_t^{\infty}\int_{A_{b1}}^{A_{b2}}\rho(A,t)dAdt$, where A_{b1} and A_{b2} are two boundaries of the energy well. The initial condition is often set as the Boltzmann distribution of reaction coordinate A in the energy well, i.e., $\rho(A,0) = e^{-F(A)/k_BT}$.

Second, using the above survival functions, we estimate the effective kinetic rates as the negative of the slopes in linear fitting of the ln[$S(t)$] versus t, i.e., $k = -d\ln[S(t)]/dt$. This is based on the assumption that the survival function exhibits exponential decay as observed in earlier studies. Finally, the apparent diffusion coefficient D is obtained by dividing the kinetic rate calculated directly using the transition time series collected from the simulation by that using the probability density solution of the Smoluchowski equation.

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 in the GaMD simulations. This allows us to recover the original biomolecular kinetic rate constants from the GaMD simulations.

6. 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.

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)

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.

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
}
```

7. 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,
```

8. Further information

Test cases for running GaMD have been included into the distribution of Amber. The latest updates, examples and simulation tips of GaMD can be found at: http://miao.compbio.ku.edu/GaMD. A tutorial based on a study we performed on alanine dipeptide[1], 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: http://miao.compbio.ku.edu/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: http://miao.compbio.ku.edu/GaMD/lib/smol_solver.tgz.

References:

1. Miao, Y., V.A. Feher, and J.A. McCammon, *Gaussian Accelerated Molecular Dynamics: Unconstrained Enhanced Sampling and Free Energy Calculation.* Journal of Chemical Theory and Computation, 2015. **11**(8): p. 3584-3595.
2. Miao, Y., A. Bhattarai, and J. Wang, *Ligand Gaussian accelerated molecular dynamics (LiGaMD): Characterization of ligand binding thermodynamics and kinetics.* bioRxiv, 2020. **https://doi.org/10.1101/2020.04.20.051979**.
3. Wang, J. and Y. Miao, *Peptide Gaussian accelerated molecular dynamics (Pep-GaMD): Enhanced sampling of peptide-protein interactions.* 2020. **In preparation**.
4. Miao, Y., *Acceleration of Biomolecular Kinetics in Gaussian Accelerated Molecular Dynamics.* Journal of Chemical Physics, 2018. **149**(7): p. 072308.
5. Hamelberg, D., T. Shen, and J.A. McCammon, *Relating kinetic rates and local energetic roughness by accelerated molecular-dynamics simulations - art. no. 2411003.* Journal of Chemical Physics, 2005. **122**(24).