## 22.6. Adaptively biased MD, steered MD, and umbrella sampling with REMD

### 22.6.1. Overview

The following describes a suite of modules useful for the calculation of the free energy associated with a reaction coordinate $\sigma(\mathbf{r}_1,\ldots,\mathbf{r}_N)$ (which is defined as a smooth function of the atomic positions $\mathbf{r}_1,\ldots,\mathbf{r}_N$):

$$f(\xi) = -k_B T \ln \big\langle \delta\left[\xi - \sigma(\mathbf{r}_1,\ldots,\mathbf{r}_N)\right] \big\rangle,$$

(the angular brackets denote an ensemble average, $k_B$ is the Boltzmann constant and $T$ is the temperature) that is also frequently referred to as the *potential of mean force*.

Specifically, new frameworks are provided for equilibrium umbrella sampling and steered molecular dynamics that enhance the functionality delivered by earlier implementations (described earlier in this manual), along with a new Adaptively Biased Molecular Dynamics (ABMD) method [432] that belongs to the general category of umbrella sampling methods with a time-dependent potential. Such methods were first introduced by Huber, Torda and van Gunsteren (the Local Elevation Method [433]) in the molecular dynamics (MD) context, and by Wang and Landau in the context of Monte Carlo simulations [434]. More recent approaches include the metadynamics method [435, 436]. All these methods estimate the free energy of a reaction coordinate from an evolving ensemble of realizations, and use that estimate to bias the system dynamics to flatten an effective free energy surface. Collectively, these methods may all be considered to be umbrella sampling methods with an evolving potential. The algorithms discussed here were developed by the group of Prof. Celeste Sagui (sagui@ncsu.edu) and Prof. Christopher Roland (cmroland@ncsu.edu); the current version was implemented by Dr. Volodymyr Babin.

The ABMD method grew out of attempts to speed up and streamline the metadynamics method for free energy calculations with a *controllable* accuracy. It is characterized by a favorable scaling in time, and only a few (two) control parameters. It is formulated in terms of the following equations:

$$m_a \frac{\mathrm{d}^2\mathbf{r}_a}{\mathrm{d}t^2} = \mathbf{F}_a + \frac{\partial}{\partial \mathbf{r}_a} U\left[t|\sigma(\mathbf{r}_1,\ldots,\mathbf{r}_N)\right],$$

$$\frac{\partial U(t|\xi)}{\partial t} = \frac{k_B T}{\tau_F} G\left[\xi - \sigma(\mathbf{r}_1,\ldots,\mathbf{r}_N)\right],$$

where the first equation represents Newton's law that governs ordinary MD (temperature and pressure regulation terms are not shown) augmented with an additional force coming from the time dependent biasing potential $U(t|\xi)$ $[U(t=0|\xi)=0]$, whose time evolution is given by the second equation. $G(\xi)$ is a positive definite and symmetric kernel, which may be thought of as a smoothed Dirac delta function. For large enough $\tau_F$ (the flooding timescale) and small kernel width, the biasing potential $U(t|\xi)$ converges towards $-f(\xi)$ as $t \to \infty$.

Our numerical implementation of the ABMD method involves the use of a bi-weight kernel along with the use of cubic B-splines (or products thereof) to discretize the biasing potential $U(t|\xi)$ w.r.t. $\xi$, and an Euler-like scheme for time integration. ABMD admits two important extensions, which lead to a more uniform flattening of $U(t|\xi)+f(\xi)$ due to an improved sampling of the "evolving" canonical distribution. The first extension is identical in spirit to the *multiple walkers metadynamics* [437, 438]. It amounts to carrying out several different MD simulations biased by the same $U(t|\xi)$, which evolves via:

$$\frac{\partial U(t|\xi)}{\partial t} = \frac{k_B T}{\tau_F} \sum_\alpha G\left[\xi - \sigma(\mathbf{r}_1^\alpha,\ldots,\mathbf{r}_N^\alpha)\right],$$

where $\alpha$ labels different MD trajectories. A second extension is to gather several different MD trajectories, each bearing its own biasing potential and, if desired, its own distinct collective variable, into a generalized ensemble for "replica exchange" with modified "exchange" rules [439–441]. Both extensions are advantageous and lead to a more uniform flattening of $U(t|\xi)+f(\xi)$.

In order to assess and improve the accuracy of the free energies, the ABMD accumulations may need to be followed up with equilibrium umbrella sampling runs, which make use of the biasing potential $U(t|\xi)$ as is. Such a procedure is very much in the spirit of adaptive umbrella sampling. With these runs, one calculates the biased probability density:

$$p^B(\xi) = \left\langle \delta\left[\xi - \sigma(\mathbf{r}_1,\ldots,\mathbf{r}_N)\right]\right\rangle_B.$$

The idea here is that if, as a result of an ABMD run, $f(\xi)+U(t|\xi)=0$ exactly, then the biased probability density $p^B(\xi)$ would be flat (constant). In practice, this is typically not the case, but one can use $p^B(\xi)$ to "correct" the free energy via:

$$f(\xi) = -U(\xi) - k_B T \ln p^B(\xi).$$

With the ABMD procedure, one can obtain accurate free energy curves and equilibrium properties. We note that to obtain ABMD free energies requires a (minor) amount of post-processing by means of the nfe-umbrella-slice utility freely available in AmberTools as described in Subsection 22.6.7. This methodology has been applied to a variety of biomolecular systems, including small peptides [432, 442, 443], sugar puckering [444], polyproline systems [445–447], guest-host systems [448, 449], polyglutamine systems [450, 451], and DNA systems [452]. In addition, SMD simulations (discussed below) have been used to examine transition pathways and mechanisms, to estimate free energy differences [446, 453], and to calculate transition rates [454–456].

While the above represents the basic ABMD implementation, AMBER 16 introduces two additional algorithms – the Well-Tempered (WT) ABMD and a selection mechanism for multiple walker ABMD – both of which enhance the stability and convergence of an ABMD simulation. Future enhancements to the set ABMD simulation tools are to include the Driven ABMD (D-ABMD) [457] and the Swarms-of-trajectories string method (STSM) [458].

**The Well-Tempered ABMD:** An alternative to the follow-up equilibrium simulations for increased ABMD accuracy is provided by the WT-ABMD, which is implemented in the spirit of the WT-metadynamics [459]. In the original ABMD implementation, the history dependent biasing potential is built up at a fixed rate:

$$U(\xi,t) = U^0(\xi) + \int_0^t dt'\,\omega\, G(\xi - \xi^{t'}), \tag{22.14}$$

in which $U(\xi,t)$ is the biasing potential at time $t$, $U^0$ is an arbitrary function that typically represents the initial guess for the biasing potential (in the absence of a guess, this is assumed to be flat) and $\omega = k_B T/\tau_F$ is a constant, unbiased rate. As the simulation proceeds and reaches convergence, then $\left\langle U(\xi, t\to\infty)\right\rangle_a \approx U^s(\xi)+u(t)$, in which $\left\langle . \right\rangle_a$ is the ensemble-average over the adaptive trajectories, the stationary term is $U^s \approx -F(\xi)$, and $u(t)$ is an additive time-dependent constant [459]. Unfortunately, updating the biasing potential at the same rate throughout the simulation may lead to a poorly converged result, since the biasing potential ends up fluctuating around $-F(\xi)$ with an amplitude that depends on $\omega$.

One way to resolve this problem is to update the kernel at a non-uniform rate by means of a "well-tempered" $\omega$:

$$U(\xi,t) = U^0(\xi) + \int_0^t dt'\,\omega(\xi^{t'},t')\, G(\xi - \xi^{t'}), \tag{22.15}$$

in which $\omega(\xi,t)$ is a time-dependent, non-uniform rate chosen to be $\omega_0\, \mathrm{e}^{-\beta' U(\xi,t)}$ ($1/\beta' = k_B T'$ where $T'$ is a pseudo-temperature) that reduces to a constant $\omega_0$ in the $\beta' \to 0$ limit (*i.e.*, resulting in conventional ABMD). With this choice, one can show that $\left\langle U(\xi,t\to\infty)\right\rangle_a \approx U^s(\xi)+u(t)$, ($u(t)$ is an additive constant) in which $U^s(\xi)$ and $F(\xi)$ are related via $U^s(\xi) = -(1+\frac{\beta'}{\beta})^{-1}F(\xi)$ or $F(\xi) = -(1+\frac{T}{T'})U^s(\xi)$. This way of updating the biasing potential leads to a considerably smoother convergence to the desired free energy and more stable ABMD simulations.

**Multiple walker selection algorithm:** The ABMD multiple walker algorithm can be improved by allowing for periodic interactions between the different walkers and "resampling" on-the-fly. The rationale behind this is that not all walkers are equally effective in sampling the configuration space. A situation that is all too common is that different walkers end up being "bunched up" or clustered together in some local metastable region, because of hidden barriers that are oriented along orthogonal degrees of freedom to the reaction coordinate. To improve this situation, one would like to facilitate walkers that are sampling the undersampled regions of phase space, and force the walkers in the oversampled regions to move away and explore regions not yet covered. Such an algorithm has previously been implemented via scripts in the NAMD code for the adaptive biasing force algorithm [460].

A resampling or selection algorithms for interacting multiple walkers requires a continual monitoring of the walkers by means of a periodic evaluation of a fitness function and a resampling of the walkers according to their fitness efficiency[460]. Efficient walkers that are wandering in the undersampled regions are enhanced by being cloned, while inefficient walkers found in the oversampled regions of phase space are correspondingly killed. This procedure is then repeated periodically during the simulation, thereby accelerating convergence to a more uniform distribution of walkers and flattening of the free energy landscape.

Our specific interacting/resampling/selection multiple-walker algorithm is implemented as follows. Each walker $n$ is assigned a weight $w_n$, which is evaluated at the end of each resampling period of time $\tau$. At the $i^{th}$ resampling period, i.e., from time $t_{(i-1)} = (i-1)\tau$ to $t_i = i\tau$, walker $n$ moves through configuration space building up its own trajectory $(r_1^n, \ldots, r_N^n)$. The weights are then tested and updated every fixed time interval of length $\tau$. Specifically, after the $i^{th}$ time interval, weights are estimated by:

$$w_n = K^{-1} \exp\left( \int_{t_{i-1}}^{t_i} S(\xi_n^t) dt \right),$$

where $\xi_n^t$ represents the collective variable evaluated at time t for trajectory $n$, $K = \sum_{n=1}^{N_w} w_n$ is the normalization factor, and

$$S(\xi) = C\nabla^2(\rho(\xi))/\rho(\xi),$$

with $\rho(\xi)$ representing the density of microstates in the collective variable space and $C$ a constant. The quantity $S(\xi)$ will be positive typically if the walker is found in the undersampled regions, which have a convex density function. Similarly, a negative $S(\xi)$ value indicates that the system is in the concave region of the density function, which typically is oversampled. In the context of ABMD implementation, the biasing potential is approximately proportional to the histogram of the collective variable by construction, and represents a good estimate for $\rho$. The implementation is therefore straightforward; the integral above is estimated for each trajectory independently by summing over $S(\xi_n^t)$ at every step from $t = t_{i-1}$ to $t = t_i$, in which $\Delta t$ is the MD *timestep*. At the end of each period the walkers send their unnormalized weight estimates to the "master processor" to normalize them. A stochastic resampling method is then used to clone/kill the replicas based on their weight factors [460]. The number of copies present in the next period for walker $n$ is determined by the integer number:

$$\begin{aligned} W_1 &= \lfloor \eta_1 + N_w w_1 \rfloor, \\ W_n &= \lfloor \eta_n + N_w \sum_{m=1}^{n} w_m \rfloor - \lfloor \eta_n + N_w \sum_{m=1}^{n-1} w_m \rfloor, \quad \text{for } n > 1. \end{aligned}$$

in which $0 < \eta_n < 1$ is drawn from a uniform distribution (using a random number generator). The atomic coordinates and velocities of the walkers with $N_n > 0$ are "sent" to $N_n$ walkers. The resampling algorithm above guarantees $\sum_n W_n = N_w$.

In terms of an ABMD simulation, the selection algorithm is most beneficial during the initial and middle parts of the simulation when there are large variations in the biasing potential. In the latter parts, when the effective free energy is almost flat, the distribution of walkers should be roughly uniform. In that case, the selection mechanism is unnecessary and, if one wishes to continue the simulation, it is best to proceed with the non interacting multiple walker algorithm. It has been found that a convenient stopping mechanism may be based on the entropy of the weights. Defining $H = \sum_n w_n log(w_n)$, the selection mechanism will be stopped if $E_w = H - log(1/N_w)$ goes below -$\varepsilon log(1/N_w)$. Here, $log(1/N_w)$ represents the entropy of uniform weights, and the stopping parameter $\varepsilon$ varies between $0 \leq \varepsilon \leq 1$. When $\varepsilon = 0$, the algorithm never stops, while $\varepsilon = 1$ forces a stop irrespective of the values of the weights.

In addition to $\varepsilon$, there are also two other user-defined variables in the selection algorithm, including the constant C and the interval time $\tau$. While the physical interpretation of $\tau$ is straightforward, C represents a pseudo diffusion constant. One may think of the selection algorithm as an induced diffusion in the reaction coordinate space; The larger the value of C, the faster the system will diffuse along the reaction coordinate space. Therefore C determines the strength or aggressiveness of the resampling algorithm. The most efficient value for C is dependent on the nature of the collective variable and the shape of its density $\rho$. Since the best choice of C for a given problem is somewhat of an art, we refer the interested reader to the ABMD tutorials on the AMBER webpage for insight into

```
&colvar
   cv_type = STRING
   cv_ni = N, cv_nr = M
   cv_i = i1, i2, ..., iN
   cv_r = r1, r2, ..., rM
/
```

Figure 22.4.: *Syntax of reaction coordinate definition:* `cv_type` *is a* `STRING`*,* `cv_i` *is a list of integer numbers and* `cv_r` *is a list of real numbers.*

choosing this variable. Finally, we also note that the multiple walker selection mechanism can be invoked as is or in conjunction with the WT-ABMD for enhanced stability and convergence.

If any of these modules prove to be useful, please consider quoting the following papers: V. Babin, C. Roland and C. Sagui, *"Adaptively biased molecular dynamics for free energy calculations"*, J. Chem. Phys. **128**, 134101 (2008); V. Babin, V. Karpusenka, M. Moradi, C. Roland and C. Sagui, *"Adaptively biased molecular dynamics: an umbrella sampling method with a time-dependent potential"*, Int. J. Quant. Chem. **109**, 3666 (2009).

From Amber16, we implement these modules from SANDER to PMEMD and the modules are GPU compatible. To keep the consistency in format, we do a series of changes and updates to the usage of these modules. One big change is that you must set ***infe*** = 1 in &cntrl to activate these modules. Also, the input format has been changed to namelist style and reaction coordinate variables will be read from separate files. For the details, please read Subsection 22.6.6

**infe** This variable controls the usage of the non-equilibrium free energy method. When *infe=0*, the ABMD and related methods are turned off; when *infe=1*, they are turned on and the blocks `nfe_smd`, `nfe_pmd`, `nfe_abmd` and `nfe_bbmd` will be recognized. The default value is 0. Note that use of these algorithms may require a (minor) amount of post-processing by means of the nfe-umbrella-slice utility freely available in AmberTools described in Subsection 22.6.7.

### 22.6.2. Reaction Coordinates

A reaction coordinate is defined in the `colvar` namelist in a separate file. (see Fig. 22.4). This section must contain a `cv_type` keyword along with a value of type `STRING` and a list of integers `cv_i` (the number of integers is defined by `cv_ni`). For some types of reaction coordinates the `colvar` section must also contain a list of real numbers, `cv_r`, whose length is defined by `cv_nr`.

The following reaction coordinates (specfied by `cv_type`) are currently implemented:

**DISTANCE:** distance (in Å) between two atoms whose indexes are read from the list `cv_i`.

**COM_DISTANCE:** distance between the center of mass of two atom groups. The cv_i list is interpreted as a list of indexes of participating atoms. Zeros separate the groups, the last zero is optional. eg: cv_i = a1,...,aN,0,b1,...bM,0.

**DF_COM_DISTANCE:** difference of distances between the center of mass of first two atom groups and second two atom groups. The cv_i list is interpreted as a list of indexes of participating atoms. Zeros separate the groups, the last zero is optional. eg: cv_i = a1,...,aN,0,b1,...bM,0,c1,...cL,0,d1,...,dK,0, DF_COM_DISTANCE is COM_DISTANCE(a1,...,aN,0,b1,...bM) - COM_DISTANCE(c1,...cL,0,d1,...,dK).

**LCOD:** linear combination of distances (in Å) between pairs of atoms listed in `cv_i` with the coefficients read from `cv_r` list. For example, `i = 1, 2, 3, 4` and `r = 1.0, -1.0` define the difference between 1-2 and 3-4 distances, *i.e.* LCOD = r1*distance(1, 2) + r2*distance(3,4).

**ANGLE:** angle (in radians) between the lines joining atoms with indexes `i1` and `i2` and atoms with indexes `i2` and `i3`.

**COM_ANGLE:** angle (in radians) formed by the center of mass of three atom groups. The cv_i list is interpreted as a list of indexes of participating atoms. Zeros separate the groups, the last zero is optional. eg: cv_i = a1,...,aN,0,b1,...bM,0,c1,...,cK,0

**TORSION:** dihedral angle (in radians) formed by atoms with indexes `i1`, `i2`, `i3` and `i4`.

**COM_TORSION:** dihedral angle (in radians) formed by the center of mass of four atom groups. The cv_i list is interpreted as a list of indexes of participating atoms. Zeros separate the groups, the last zero is optional. eg: cv_i = a1,...,aN,0,b1,...bM,0,c1,...,CK,0,d1,...,dL,0.

**COS_OF_DIHEDRAL:** sum of cosines of dihedral angles formed by atoms with indexes in the list `cv_i`. The number of atoms must be a multiple of four.

**SIN_OF_DIHEDRAL:** sum of sines of dihedral angles formed by atoms with indexes in the list cv_i. The number of atoms must be a multiple of four.

**PAIR_DIHEDRAL:** sum of cosines of a list of angles each formed by summing two neighboring dihedral angles from a list formed by atoms with indices cv_i. The number of atoms must be a multiple of four. For a list of dihedral angles such as $\{\alpha_1, . . . ,\alpha_N\}$, PAIR_DIHEDRAL is $\sum_{i=1}^{N-1} cos(\alpha_i + \alpha_{i+1})$ which ranges between -N+1 and N-1.

**PATTERN_DIHEDRAL:** a particular pattern-recognizing function defined on a list of dihedral angles formed by atoms with indices cv_i. The number of atoms must be a multiple of four. The definition is particularly relevant for the dihedral angles with a binary-like behavior of being either around 0 or 180 (e.g., ω backbone dihedral angle). For a list of dihedral angles such as $\{\alpha_1, . . . ,\alpha_N \}$, PATTERN_DIHEDRAL is $\sum_{i=1}^{N} cos^2(\alpha_i/2)2^{i-1}$ which ranges between 0 and $2^N - 1$.

**R_OF_GYRATION:** radius of gyration (in Å) of atoms with indexes given in the `cv_i` list (mass weighted).

```
&colvar
   cv_type = 'MULTI_RMSD'
   cv_ni = 9, cv_nr = 21,
   cv_i = 1, 2, 3, 4, 0, 3, 4, 5, 0 ! the last zero is optional
   cv_r = 1.0, 1.0, 1.0, ! group #1, atom 1
        2.0, 2.0, 2.0, ! group #1, atom 2
        3.0, 3.0, 3.0, ! group #1, atom 3
        4.0, 4.0, 4.0, ! group #1, atom 4
       23.0, 23.0, 23.0, ! group #2, atom 3
          4.0, 4.0, 4.0, ! group #2, atom 4
          5.0, 5.0, 5.0 ! group #2, atom 5
/
```

Figure 22.5.: *An example of `MULTI_RMSD` variable definition.*

**MULTI_RMSD:** RMS (in Å, mass weighted) of RMSDs of several groups of atoms w.r.t. reference positions provided in the `cv_r` list. The `cv_i` list is interpreted as a list of indexes of participating atoms. Zeros separate the groups. An atom may enter several groups simultaneously. The `cv_r` array is expected to contain the reference positions (without zero sentinels). The implementation uses the method (and the code) introduced in Ref. [461]. An example of variable of this type is presented in Fig. 22.5. Two groups are defined here: one comprises the atoms with indexes 1, 2, 3, 4 (line 3 in Fig. 22.5, numbers prior to the first zero) and another one of atoms with indexes 3, 4, 5. The code will first compute the (mass weighted) RMSD ($R_1$) of atoms belonging to the first group w.r.t. reference coordinates provided in the `cv_r` array (first $12 = 4 \times 3$ real numbers of it; lines 4, 5, 6, 7 in Fig. 22.5). Next, the (mass weighted) RMSD ($R_2$) of atoms of the second group w.r.t. the corresponding reference coordinates (last $9 = 3 \times 3$ elements of the cv_r array in Fig. 22.5) will be computed. Finally, the code will compute the value of the variable as follows:

$$\text{value} = \sqrt{\frac{M_1}{M_1 + M_2} R_1^2 + \frac{M_2}{M_1 + M_2} R_2^2}\,,$$

where $M_1$ and $M_2$ are the total masses of atoms in the corresponding groups.

**N_OF_BONDS:**

$$\text{value} = \sum_p \frac{1 - \left(r_p \middle/ r_0\right)^6}{1 - \left(r_p \middle/ r_0\right)^{12}}\,,$$

where the sum runs over pairs of atoms $p$, $r_p$ denotes distance between the atoms of pair $p$ and $r_0$ is a parameter measured in Å. The cv_r array must contain exactly one element that is interpreted as $r_0$. The cv_i array is expected to contain pairs of indexes of participating atoms. For example, if 1 and 2 are the indexes of Oxygen atoms and 3, 4, 5 are the indexes of Hydrogen atoms and one intents to count all possible O-H bonds, the cv_i list must be (1, 3, 1, 4, 1, 5, 2, 3, 2, 4, 2, 5), that is, it must explicitly list all the pairs to be counted.

**HANDEDNESS:**

$$\text{value} = \sum_a \frac{\mathbf{u}_{a,3} \cdot [\mathbf{u}_{a,1} \times \mathbf{u}_{a,2}]}{|\mathbf{u}_{a,1}|\,|\mathbf{u}_{a,2}|\,|\mathbf{u}_{a,3}|}\,,$$

where

$$\begin{aligned}
\mathbf{u}_{a,1} &= \mathbf{r}_{a+1} - \mathbf{r}_a \\
\mathbf{u}_{a,2} &= \mathbf{r}_{a+3} - \mathbf{r}_{a+2} \\
\mathbf{u}_{a,3} &= (1-w)(\mathbf{r}_{a+2} - \mathbf{r}_{a+1}) + w(\mathbf{r}_{a+3} - \mathbf{r}_a)\,,
\end{aligned}$$

and $\mathbf{r}_a$ denote the positions of participating atoms. The cv_i array is supposed to contain indexes of the atoms and the cv_r array may provide the value of $w$ ($0 \le w \le 1$, the default is zero).

**N_OF_STRUCTURES:**

$$\text{value} = \sum_g \frac{1 - \left(R_g \middle/ R_{0,g}\right)^6}{1 - \left(R_g \middle/ R_{0,g}\right)^{12}}\,,$$

where the sum runs over groups of atoms, $R_g$ denotes the RMSD of the group $g$ w.r.t. some reference coordinates and $R_{0,g}$ are positive parameters measured in Å. The cv_i array is expected to contain indexes of participating atoms with zeros separating different groups. The elements of the cv_r array are interpreted as the reference coordinates of the first group followed by their corresponding $R_0$; then followed by the reference coordinates of the atoms of the second group, followed by the second $R_0$, and so forth. To make the presentation clearer, let us consider the example presented in Fig. 22.6. The atomic groups and reference coordinates are the same as the ones shown in Fig. 22.5. Lines 7 and 11 in Fig. 22.6 contain additional entries that set the values of the threshold distances $R_{0,1}$ and $R_{0,2}$. To compute the variable, the code first computes the mass weighted RMSD values $R_1$ and $R_2$ for both groups –much like in the MULTI_RMSD case– and then combines those in a manner similar to that used in the N_OF_BONDS variable.

$$\text{value} = \frac{1 - \left(R_1 \middle/ R_{0,1}\right)^6}{1 - \left(R_1 \middle/ R_{0,1}\right)^{12}} + \frac{1 - \left(R_2 \middle/ R_{0,2}\right)^6}{1 - \left(R_2 \middle/ R_{0,2}\right)^{12}}\,.$$

In other words, the variable “counts” the number of structures that match (stay close in RMSD sense) with the reference structures.

```
&colvar
   cv_type = 'N_OF_STRUCTURES'
   cv_ni = 9, cv_nr = 23,
   cv_i = 1, 2, 3, 4, 0, 3, 4, 5, 0 ! the last zero is optional
   cv_r = 1.0, 1.0, 1.0, ! group #1, atom 1
        2.0, 2.0, 2.0, ! group #1, atom 2
        3.0, 3.0, 3.0, ! group #1, atom 3
        4.0, 4.0, 4.0, ! group #1, atom 4
        1.0,           ! R0 for group #1
     23.0, 23.0, 23.0, ! group #2, atom 3
        4.0, 4.0, 4.0, ! group #2, atom 4
        5.0, 5.0, 5.0, ! group #2, atom 5
        2.0           ! R0 for group #2
/
```

Figure 22.6.: *An example of* `N_OF_STRUCTURES` *variable.*

### 22.6.3. Steered Molecular Dynamics

The `&smd` namelist, if present in the `MDIN` file, activates the steered `MD` code (the method itself is extensively described in the literature: see for example Ref. [462] and references therein). The prefix NFE appears in several switches to do with steered MD: this stands for “Non-equilibrium Free Energy”.

The following is recognized within the `&smd` namelist:

**output_file** sets the output file name. Default is ’nfe-smd.txt’.

**output_freq** sets the output frequency (in `MD` steps). Default is 50.

**cv_file** sets the collective variable file name. Default is ’nfe-smd-cv’.

There must be at least one reaction coordinate defined (that is, there must be at least one `&colvar` namelist in the cv_file). The steered `MD` code requires that additional entries be present in the `colvar` namelist:

**path** the steering path whose elements must be real numbers. The path must include at least two elements. The upper limit on the number of entries is 20000. The elements define Catmull-Rom spline used for steering.

**npath** sets the number of elements in path. Default is 0.

**path_mode** The way steering paths are constructed. There are two modes available. In `SPLINE` mode (default) the path is approximated by spline that passes through the given points; in `LINES` mode the path is represented by the line segments joining the control points.

**harm** specifies the harmonic constant. If a single number is provided, e.g., `harm = 10.0`, then it is constant throughout the run. If two or more numbers are provided, e.g., `harm = 10.0, 20.0`, then the harmonic constant follows Catmull-Rom spline built upon the provided values.

**nharm** sets the number of elements in harm. Default is 0.

**harm_mode** The way harmonical paths are constructed, similar with path_mode.

An example of `MDIN` file and `CV.IN` file for steered `MD` is shown in Fig. 22.7. The reaction coordinate is defined in `cv.in`. The spring constant is set constant throughout the run and the steering path is configured from 5.0 to 3.0. The values of the reaction coordinate, harmonic constant and the work performed on the system are requested to be dumped to the `smd.txt` file every 50 MD steps.

```
title line
&cntrl
..., infe = 1
/

&smd
   output_file = 'smd.txt'
   output_freq = 50
   cv_file = 'cv.in'
/
```

```
cv_file
&colvar
  cv_type = 'DISTANCE'
  cv_ni = 2
  cv_i = 5, 9
  npath = 2, path = 5.0, 3.0, path_mode = 'LINES',
  nharm = 1, harm = 10.0
/
```

Figure 22.7.: *An example `MDIN` file and `CV.IN` file for steered `MD`. Only the relevant part is shown.*

### 22.6.4. Umbrella sampling

To activate the umbrella sampling code, the `&pmd` namlist must be present in the `MDIN` file. This is for SANDER only: `&pmd` is currently not activated for PMEMD. Users who do want to use this code with PMEMD should write to *sagui@ncsu.edu* or *fpan3@ncsu.edu*. `output_file`, `output_freq` and `cv_file` entries are recognized like in the steered `MD` case presented earlier. The `cv_file` must contain at least one `&colvar` namelist section. For umbrella sampling, the `&colvar` section(s) must contain two additional entries:

**anchor_position** real number that sets the position of the minimum of the umbrella (harmonic) potential. Default is 0.

**anchor_strength** non-negative real number that sets the harmonic constant for the umbrella (harmonic) potential. Default is 0.

An example of an `MDIN` file and `CV.IN` file for an umbrella sampling simulation is shown in Fig. 22.8. The first reaction coordinate here is the angle formed by the lines joining the 5th with 9th and 9th with 15th atoms. It is to be harmonically restrained near 1.0 *rad* (`anchor_position` entry) using the spring of strength 10.0 $kcal/mol/rad^2$ (`anchor_strength` entry). The second reaction coordinate requested in Fig. 22.8 is a dihedral angle (`type = 'TORSION'`) formed by the 1st, 2nd, 3rd and 4th atoms (the `cv_i` array). It is to be restrained near zero with strength 23.8 $kcal/mol/rad^2$. The values of the reaction coordinate(s) are to be dumped every 50 `MD` steps to the `pmd.txt` file.

The `NFE` implementation of umbrella sampling works correctly with the Amber standard replica-exchange MD described earlier in this manual. It assumes, however, that the number and type of reaction coordinate(s) are the same in all replicas. On the other hand, both `anchor_position` and `anchor_strength` may be different for different temperatures. For replica-exchange MD the output files (set by the `output_file` keyword on a per-replica basis) are temperature bound (or `MDIN`-bound, since there is one-to-one temperature-`MDIN` correspondence).

### 22.6.5. Adaptively Biased Molecular Dynamics

The implementation has a very simple and intuitive interface: the code is activated if either an `&abmd` (both SANDER and PMEMD) or an `&bbmd` (SANDER only) namelist is present in the `MDIN` file (the difference between

```
title line
&cntrl
..., infe = 1
/

&pmd
   output_file = 'pmd.txt'
   output_freq = 50
   cv_file = 'cv.in'
/
```

```
cv_file
&colvar ! first
   cv_type = 'ANGLE'
   cv_ni = 3, cv_i = 5, 9, 15
   anchor_position = 1.0
   anchor_strength = 10.0
/
&colvar ! second
   cv_type = 'TORSION'
   cv_ni = 4, cv_i = 1, 2, 3, 4
   anchor_position = 0.0
   anchor_strength = 23.8
/
```

Figure 22.8.: *An example `MDIN`file and `CV.IN`file for umbrella sampling (only relevant part is presented in full).*

those "flavors" is purely technical and will become clear later). Unlike in the `&smd` and `&pmd` cases, the dimensionality of a reaction coordinate (the number of `&colvar` namelists in the `cv_file`) cannot exceed five (though three is already hardly useful due to statistical reasons).

As previously noted, in order to activate the ABMD and related algorithm, the variable ***infe*** in &cntrl must be set to unity (*i.e.* ***infe*** = 1; default value ***infe*** = 0).

In addition to the `cv_file` entry, the following entries are recognized within the `&abmd` (or `&bbmd`) namelist:

**mode** sets the execution mode. There are three modes available: `'ANALYSIS' | 'UMBRELLA' | 'FLOODING'`. In `ANALYSIS` mode the dynamics is not altered. The only effect of this mode is that the value(s) of the reaction coordinate(s) is(are) dumped every `monitor_freq` to `monitor_file`. In `UMBRELLA` mode, biasing potential from the `umbrella_file` is used to bias the simulation ($\tau_F = \infty$, biasing potential does not change). In `FLOODING` mode the adaptive biasing is enabled.

**monitor_file** sets the name of the file to which value(s) of reaction coordinate(s) (along with the magnitude of biasing potential in `FLOODING` mode) are dumped.

**monitor_freq** the frequency of the output to the `monitor_file`.

**timescale** $\tau_F$, the flooding timescale in picoseconds (only required in `FLOODING` mode).

**umbrella_file** biasing potential file name (the file must exist for the `UMBRELLA` mode).

In `FLOODING` mode, the following two entries are optional:

**snapshots_basename** sets the name of the file to which the biasing potential is dumped during the simulation for snapshot.

**snapshots_freq** the frequency of dumping snapshot biasing potential (in MD steps). If snapshots_freq is not specified, the snapshot biasing potential will not be dumped.

and the &colvar namelist for &abmd method must also contain the following entries:

**cv_min** smallest desired value of the reaction coordinate (required, unless the reaction coordinate is limited from below).

**cv_max** largest desired value of the reaction coordinate (required, unless the reaction coordinate is limited from above).

**resolution** the "spatial" resolution for the reaction coordinate.

To access the biasing potential files created in the course of FLOODING simulations, the nfe-umbrella-slice utility is provided (it prints a short description of itself if invoked with --help option).

The multiple-walker selection algorithm can improve the simulation by resampling between different walkers. The well-tempered ABMD can lead to a smoother convergence to the desired free energy. These two algorithm are implemented to SANDER and PMEMD from Amber16 onwards.

The multiple-walker selection algorithm currently works with &abmd only. The algorithm should be used only within the multiple-walker scheme (*i.e.*, when command-line **-rem** flag is set to zero). The following entries are recognized regarding with the selection algorithm (selection algorithm can work with FLOODING and UMBRELLA mode):

**selection_freq** positive integer number that sets the frequency of the resampling algorithm (in MD steps). If selection_freq is not specified, the selection algorithm will not be used.

**selection_constant** positive real number that sets the parameter C. if selection_freq is specified, specifying selection_constant is required (no default value). Parameter C is to determine how strong the selection mechanism is. If C is too large, all the walkers will be replaced with the most dominant one. If C is too small, there will be no killing/duplicating of walkers.

**selection_epsilon** positive real number (typically less than unity) that sets the stopping criterion parameter ε . Parameter ε determines the threshold for stopping the selection algorithm. If selection_epsilon is not specified, there will be no stop to the algorithm. If selection_epsilon is equal or larger than one, the algorithm will be stopped after the first attempt.

The well-tempered flavor can be used within either &abmd or &bbmd namelist. There are two entries relevant to the well-tempered feature:

**wt_temperature** positive real number that sets the pseudo-temperature T'. If this flag is not specified, conventional ABMD will be used (*i.e.*, T' → ∞ or β' → 0). The smaller the T'; the smoother/slower the convergence.

**wt_umbrella_file** the file name of true biasing potential after modification by 1 + ( T /T' ) in which T is the reference temperature of the system (temp0).

An example MDIN file and CV.IN file for the nfe_abmd flavor of ABMD is shown in the Fig. 22.9.

In this example, the reaction coordinate is defined as the distance between the 5th and 9th atoms (more than one reaction coordinates might be requested by mere inclusion of additional &colvar subsections). The mode is set to FLOODING thus enabling the adaptive biasing with flooding timescale $\tau_F = 100 ps$. The region of interest of the reaction coordinate is specified to be between -1 Å and 10 Å and the resolution is set to 0.5 Å. The lower bound (-1 Å) could have been omitted for DISTANCE variable: the default value of zero would be used in such case. The code will try to load the biasing potential from the umbrella.nc file and use it as the value of $U(t|\xi)$ at the beginning of the run. The biasing potential built in the course of simulation will be saved to the same file (umbrella.nc) every time the RESTRT file is written. The selection algorithm is used with the frequency of selection defined as 10000 MD steps and selection constant defined as 0.001. The well-tempered algorithm is also used, with the pseudo-temperature defined as 10000 K in and the true biasing potential will be dumped as wt_umbrella.nc file.

```
title line
&cntrl
..., infe = 1
/

&abmd
   mode = 'FLOODING'

   monitor_file = 'abmd.txt'
   monitor_freq = 33
   cv_file = 'cv.in'

   umbrella_file = 'umbrella.nc'

   timescale = 100.0 ! in ps

   selection_freq = 10000
   selection_constant = 0.001

   wt_temperature = 10000.0
   wt_umbrella_file = 'wt_umbrella.nc'
/
```

```
cv_file
&colvar
  cv_type = 'DISTANCE'
  cv_ni = 2, cv_i = 5, 9
  cv_min = -1.0, cv_max = 10.0 ! min is not needed for DISTANCE
  resolution = 0.5 ! required for mode = FLOODING
/
```

Figure 22.9.: *An example `MDIN` file and `CV.IN` file for `ABMD` (only the relevant part is presented in full).*

The `nfe-umbrella-slice` utility can then be used to access its content. An `MDIN` file for the follow up biased run at equilibrium would look much like the one shown in the Fig. 22.9, but with `mode` changed from `FLOODING` to `UMBRELLA`.

For SANDER only, the `&abmd` code works correctly with replica-exchange (that is, for `-rem` flag set to 1). In such case the monitor and umbrella files are temperature-bound (unlike, e.g., `MDOUT` and `MDCRD` files that require post processing). If number of `sander` groups exceeds one (the flag `-ng` is greater than one) and `-rem` flag is set to zero, the code runs *multiple walkers* `ABMD`. In both cases the number and type(s) of variable(s) must be the same across all replicas.

Finally, the `&bbmd` flavor allows one to run replica-exchange (`AB`)`MD` with different reaction coordinates and different modes (`ANALYSIS`, `UMBRELLA` or `FLOODING`) in different replicas (along with different temperatures, if desired). To this end, the `-rem` flag must be set to zero and the `&bbmd` sections must be present in all `MDIN` files. The `MDIN` file for the replica of rank zero (first line in the group file) is expected to contain additional information as compared to `%abmd` case (an example of such `MDIN` file for replica zero is shown in Fig. 22.10). The `MDIN` files for all other replicas except zero do not need any additional information, and therefore take the same form as in the `&abmd` flavor (except that the namelist is changed from `&abmd` to `&bbmd`, thus activating a slightly different code path). Each `MDIN` file may define its own reaction coordinates, have different `mode` and temperature if desired.

Within the first replica `&bbmd` namelist the following additional entries are recognized:

**exchange_freq** number of `MD` steps between the exchange attempts.

**exchange_log_file** the name of the file to which exchange statistics is to be reported.

```
title line
&cntrl
..., infe = 1
/

&bbmd

   ! 0th replica only

   exchange_freq = 100 ! try for exchange every 100 steps

   exchange_log_file = 'bbmd.log'
   exchange_log_freq = 25

   mt19937_seed = 123455 ! random generator seed
   mt19937_file = 'mt19937.nc' ! file to store/load the PRG

   ! not specific for 0th replica

   mode = 'ANALYSIS'

   monitorfile = 'bbmd.01.txt' ! it is wise to have different
                               ! names in different replicas
   monitor_freq = 123
   cv_file = 'cv.in'
/
```

```
cv_file
&colvar
  cv_type = 'DISTANCE'
  cv_ni = 2, cv_i = 5, 9
/
```

Figure 22.10.: *An example* `MDIN` *file and* `CV.IN` *file for* `&bbmd` *flavor of* `ABMD` *(only the relevant part is presented in full)(for SANDER only).*

**exchange_log_freq** frequency of `exchange_log_file` updates.

**mt19937_seed** seed for the random generator (Mersenne twister [463]).

**mt19937_file**the name of the file to which the state of the Mersenne twister is dumped periodically (for restarts).

The `MDOUT`, `MDCRD`, `RESTRT`, `umbrella_file` and `monitor_file` files are `MDIN`-bound in course of the `bbmd`-enabled run. An example that uses this kind of replica exchange is presented in Ref.444.

### 22.6.6. Implementation in PMEMD

From Amber16 and onwards, the above features have been implemented in PMEMD. Becaused of GPU-enhanced calculations in PMEMD, users can now utilize GPUs to improve the speed of free energy sampling a lot with *pmemd.cuda* and *pmem.cuda.MPI*. This is very important when we want to study system in explicit solvent. With *pmemd* and *pmemd.MPI* the option to use CPUs is also kept, and the whole method has been tested sucessfully on both implicit and explicit enviroment. If you have questions with regards to the PMEMD implementation of ABMD and related algorithms, please contact Feng Pan <fpan3@ncsu.edu> and Prof. M. Moradi <moradi@uark.edu>.

Not all functionality has been implemented from SANDER to PMEMD, and some functionality has been modified. Here are all the changes:

- Naming was changed, all the input format is changed to namelist style. The reaction coordinate (collective variable) information is read from a separate file so it can be reusable by different runs.
- A new variable ***infe*** is added within the namelist &cntrl, to control the usage of the NFE method in a friendly way. To disable the NFE method, set ***infe*** to 0; to enable it, set ***infe*** to 1.
- Temperature replica-exchange method in `&abmd` block is not available(that is `-rem` must be set to 0).
- `&pmd`+ `&bbmd` (for umbrella sampling and T-REMD or H-REMD) is not currently available with this release. PMEMD code (to be used "as is") may be obtained by writing to sagui@ncsu.edu or fpan3@ncsu.edu. Code will be put on Amber website via an upcoming tutorial.
- Two new entries have been added to the blocks `&abmd` and `&bbmd` to give snapshots of the biasing potential during the simulation:

  **snapshots_basename** = STRING : sets the snapshots file name.

  **snapshots_freq** = INTEGER : sets the snapshot frequency. (in MD steps)
- It is not sensible to run a one-replica simulation with multiple GPUs
- Two new algorithms have been added to both SANDER and PMEMD: (1) a selection algorithm for multiple-walker ABMD; (2) the well-tempered ABMD (WT-ABMD).
- Several new reaction coordinates are added, which include
  - `type = SIN_OF_DIHEDRAL`
  - `type = PAIR_DIHEDRAL`
  - `type = PATTERN_DIHEDRAL`
  - `type = DF_COM_DISTANCE`

### 22.6.7. Post-processing of biasing potential

When you get the biasing potential (`*.nc` file), you can always use the `nfe-umbrella-slice` utility to access its content and get a friendly-written ASCII file from which one can obtain the free energy map. The output is the free energy value, which is the opposite of the biasing potential ( $f = -U$ ($units\ kcal/mol$)). The `nfe-umbrella-slice` utility has been included in AmberTools.

**Usage:** `nfe-umbrella-slice [options] bias_potential.nc`

**Options:**

**`-h,--help`** Print out a usage summary

**`-p,--pretend`** Only print out the basic properties of source without biasing potential data (off by default)

**`-g,--gradient`** Print out the gradients (off by default)

**`-t,--translate`** Translate the numerical value of biasing potential by a real number (0 by default)

**`-d,--dimensions`** Set the way of slice in different dimensions. The format is "D1:D2:...:Dn", n is the number of dimensions. Each D can only be set with one number or three numbers, seperated by comma. If only one number is set, the variable will be fixed at that value. If three numbers are set, the first two define the boundary of slice and the last one defines the number of points.

**Example:**

- `nfe-umbrella-slice -d “-5.0,5.0,50” 1d-bias.nc > FE.dat`

This processes the 1-dimensional biasing potential file `1d-bias.nc` and prints out the results to `FE.dat`. The variable will be taken from -5.0 to 5.0 using 50 points.

- `nfe-umbrella-slice -g -t 50.0 -d “1.0:-2.0,2.0,20” 2d-bias.nc > FE.dat`

This processes the 2-dimensional biasing potential file `2d-bias.nc` and prints out the results to `FE.dat`. All the free energy will be incremented by a constant 50.0. The gradients in both dimensions will be printed out. For the first dimension, the variable will be fixed at 1.0; for the second dimension, the variable will be taken from -2.0 to 2.0 using 20 points.

- `nfe-umbrella-slice wt_umbrella.nc > wt_FE.dat`

This processes the biasing potential after WT-ABMD and prints out the results to `wt_FE.dat`. The default dimensional information is obtained and used by the program from the biasing potential file.

[423] Okur, A.; Roe, D.; Cui, G.; Hornak, V.; Simmerling, C. Improving convergence of replica-exchange simulations through coupling to a high-temperature structure reservoir. *J. Chem. Theory comput.*, **2007**, *3*, 557–568.

[424] Roitberg, A.; Okur, A.; Simmerling, C. Coupling of replica exchange simulations to a non-Boltzmann structure reservoir. *J. Phys. Chem. B*, **2007**, *111*, 2415–2418.

[425] Sabri Dashti, D.; Roitberg, A. E. Calculating the pKa Shift of Titratable Group at Position 66 of Staphylococcal Nuclease Mutant with the Replica Exchange Free Energy Perturbation method (REFEP). *In preparation*, **2012**.

[426] Sabri Dashti, D.; Roitberg, A. E. Optimization of Umbrella Sampling Replica Exchange Molecular Dynamics by Replica Positioning. *J. Chem. Theory Comput.*, **2013**, *9*, 4692–4699.

[427] Fajer, M.; Hamelberg, D.; McCammon, J. A. Replica-Exchange Accelerated Molecular Dynamics (REX-AMD) Applied to Thermodynamic Integration. *J. Chem. Theory Comput.*, **2008**, *4*, 1565–1569.

[428] Arrar, M.; de Oliveira, C. A. F.; Fajer, M.; Sinko, W.; McCammon, J. A. w-REXAMD: A Hamiltonian Replica Exchange Approach to Improve Free Energy Calculations for Systems with Kinetically Trapped Conformations. *J. Chem. Theory Comput.*, **2013**, *9*, 18–23.

[429] Shirts, M. R.; Chodera, J. D. Statistically optimal analysis of samples from multiple equilibrium states. *J. Chem. Phys.*, **2008**, *129*, 124105–124105–10.

[430] Pohorille, A.; Jarzynski, C.; Chipot, C. Good practices in Free-Energy calculations. *J. Phys. Chem. B*, **2010**, *114*, 10235–10253.

[431] Swails, J. M. *Free Energy Simulations of Complex Biological Systems at Constant pH.* PhD thesis, University of Florida, 2013.

[432] Babin, V.; Roland, C.; Sagui, C. Adaptively biased molecular dynamics for free energy calculations. *J. Chem. Phys.*, **2008**, *128*, 134101.

[433] Huber, T.; Torda, A. E.; van Gunsteren, W. F. Local elevation: a method for improving the searching properties of molecular dynamics simulation. *J. Comput. Aided. Mol. Des.*, **1994**, *8*, 695–708.

[434] Wang, F.; Landau, D. P. Efficient, multiple-range random walk algorithm to calculate the density of states. *Phys. Rev. Lett.*, **2001**, *86*, 2050–2053.

[435] Laio, A.; Parrinello, M. Escaping free-energy minima. *Proc. Natl. Acad. Sci.*, **2002**, *99*, 12562–12566.

[436] Iannuzzi, M.; Laio, A.; Parrinello, M. Efficient exploration of reactive potential energy surfaces using car-parrinello molecular dynamics. *Phys. Rev. Lett.*, **2003**, *90*, 238302–1.

[437] Lelièvre, T.; Rousset, M.; Stoltz, G. Computation of free energy profiles with parallel adaptive dynamics. *J. Chem. Phys.*, **2007**, *126*, 134111.

[438] Raiteri, P.; Laio, A.; Gervasio, F. L.; Micheletti, C.; Parrinello, M. Efficient reconstruction of complex free energy landscapes by multiple walkers metadynamics. *J. Phys. Chem.*, **2006**, *110*, 3533–3539.

[439] Sugita, Y.; Kitao, A.; Okamoto, Y. Multidimensional replica-exchange method for free-energy calculations. *J. Chem. Phys.*, **2000**, *113*, 6042–6051.

[440] Bussi, G.; Gervasio, F. L.; Laio, A.; Parrinello, M. Free-energy landscape for $\beta$ hairpin folding from combined parallel tempering and metadynamics. *J. Am. Chem. Soc.*, **2006**, *128*, 13435–13441.

[441] Piana, S.; Laio, A. A bias-exchange approach to protein folding. *J. Phys. Chem. B*, **2007**, *111*, 4553–4559.

[442] Babin, V.; Roland, C.; Darden, T. A.; Sagui, C. The free energy landscape of small peptides as obtained from metadynamics with umbrella sampling corrections. *J. Chem. Phys.*, **2006**, *125*, 2049096.

[443] Babin, V.; Karpusenka, V.; Moradi, M.; Roland, C.; Sagui, C. Adaptively biased molecular dynamics: an umbrella sampling method with a time dependent potential. *Inter. J. Quantum Chem.*, **2009**, *109*, 3666–3678.

[444] Babin, V.; Sagui, C. Conformational free energies of methyl-alpha-l-iduronic and methyl-beta-d-glucuronic acids in water. *J. Chem. Phys.*, **2010**, *132*, 104108.

[445] Moradi, M.; Babin, V.; Roland, C.; Darden, T.; Sagui, C. Conformations and free energy landscapes of polyproline peptides. *Proc. Natl. Aca. Sci. USA*, **2009**, *106*, 20746.

[446] Moradi, M.; Babin, V.; Roland, C.; Sagui, C. A classical molecular dynamics investigation of the free energy and structure of short polyproline conformers. *J. Chem. Phys.*, **2010**, *133*, 125104.

[447] Moradi, M.; Babin, V.; Sagui, C.; Roland, C. in *Proline: Biosynthesis, Regulation and Health Benefits*, B. Nedjimi, Ed., pp 67–110. Nova Publishers, 2013.

[448] Moradi, M.; Babin, V.; Sagui, C.; Roland, C. A statistical analysis of the PPII propensity of amino acid guests in proline-rich peptides. *Biophysical J.*, **2011**, *100*, 1083 – 1093.

[449] Moradi, M.; Babin, V.; Sagui, C.; Roland, C. PPII propensity of multiple-guest amino acids in a proline-rich environment. *J. Phys. Chem. B.*, **2011**, *115*, 8645–8656.

[450] Babin, V.; Roland, C.; Sagui, C. The alpha-sheet: A missing-in-action secondary structure? *Proteins –Structure Function and Bioinformatics*, **2011**, *79*, 937–946.

[451] Moradi, M.; Babin, V.; Roland, C.; Sagui, C. Are long-range structural correlations behind the aggregation phenomena of polyglutamine diseases? *PLoS Comput. Biol.*, **2012**, *8*, e1002501.

[452] Moradi, M.; Babin, V.; Roland, C.; Sagui, C. Reaction path ensemble of the B-Z-DNA transition: a comprehensive atomistic study. *Nucleic Acids Research*, **2013**, *41*, 33–43.

[453] Moradi, M.; Lee, J.-G.; Babin, V.; Roland, C.; Sagui, C. Free energy and structure of polyproline peptides: an ab initio and classical molecular dynamics investigation. *Int. J. Quantum Chem.*, **2010**, *110*, 2865–2879.

[454] Moradi, M.; Sagui, C.; Roland, C. Calculating relative transition rates with driven nonequilibrium simulations. *Chem. Phys. Lett.*, **2011**, *518*, 109.

[455] Moradi, M.; Sagui, C.; Roland, C. Investigating rare events with nonequilibrium work measurements: I. Nonequilibrium transition paths. *J. Chem. Phys.*, **2014**, *140*, 034114.

[456] Moradi, M.; Sagui, C.; Roland, C. Investigating rare events with nonequilibrium work measurements: II. Transition and reaction rates. *J. Chem. Phys.*, **2014**, *140*, 034115.

[457] Moradi, M.; Tajkorshid, E. Driven Metadynamics: Reconstructing equilibrium free energies from driven adaptive-bias simulations. *J. Phys. Chem. Lett.*, **2013**, *4*, 1882.

[458] Pan, A. C.; Sezer, D.; Roux, B. Finding transition pathways using the string method with swarms of trajectories. *J. Phys. Chem. B*, **2008**, *112*, 3432–3440.

[459] A. Barducci and G. Bussi and M. Parrinello. Well-tempered metadynamics: a smoothly converging and tunable free energy method. *Phys. Rev. Lett.*, **2008**, *100*, 020603.

[460] K. Minoukadeh and Ch. Chipot and T. Lelievre. Potential of Mean Force Calculations: A multiple-walker adaptive biasing force technique. *J. Chem. Theor. and Comput.*, **2010**, *6*, 1008.

[461] Coutsias, E. A.; Seok, C.; Dill, K. A. Using quaternions to calculate RMSD. *J. Comput. Chem.*, **2004**, *25*, 1849–1857.

[462] Park, S.; Khalili-Araghi, F.; Tajkhorshid, E.; Schulten, K. Free energy calculation from steered molecular dynamics simulations using Jarzynski's equality. *J. Chem. Phys.*, **2003**, *119*, 3559–3566.