WHAM convergence module for a two-dimensional reaction coordinate system

Cameron Kopp

February 2023

1 Introduction

1.1 A time consuming consideration

I gave a considerable amount of thought on whether or not a two-dimensional probability distribution produced by a biased MD simulation along two reaction coordinates could be considered separable, i.e. $\rho(x,y) = \rho(x) \cdot \rho(y)$. I was well aware, and am even more convinced of it now, that there is no reason why this should be the case. However, the reason why I was so tempted was that it would have drastically reduced the time of convergence, as it would have implied minimizing the equations with a greatly reduced number of parameters (if an equal number of simulations were conducted along each reaction coordinate, then the number of parameters would be reduced by a power of $1/2$). Nonetheless, seeing as most of the literature does not use separation of variables and that the purpose here is to improve upon the commonly used iterative convergence algorithm, we still expect to observe significantly reduced run times associated with the convergence of the WHAM equations.

2 WHAM equations for two reaction coordinates

2.1 Biasing potentials along two reaction coordinates

It will be useful to compare the two dimensional case back to the one dimensional case for a reference, seeing as the extra indices in the subsequent equations have a tendency of becoming burdensome. Equations which are unnumbered typically designate the one-dimensional reference point.

Assuming the biasing potentials for each reaction coordinate are the same (in the context of multi-ion pulling simulations this should hold, though there are systems where this is not the case), the definition of the harmonic biasing potentials applied to each simulation does not change drastically with the addition of a second reaction coordinate. We define $w_i\left(x,y\right)$ as the two dimensional biasing potential for simulation $i = 1,...,S$ where S is the total number of simulations. In the two dimensional case, we use $\mu_{1,i}$ and $\mu_{2,i}$ to correspond to the different centers of the biasing potentials along the two reaction coordinates for the same simulation i.

The one dimensional form of the equation is given by

$$w_i\left(x\right) = \frac{k}{2}\left(x-\mu_i\right)^2, \quad i = 1,...,S \quad \Rightarrow$$

and we make the following alterations for the two dimensional case

$$w_i\left(x,y\right) = \frac{k}{2}\left[\left(x-\mu_{1,i}\right)^2 + \left(y-\mu_{2,i}\right)^2\right], \quad i = 1,...,S \tag{1}$$

Defining the pairs μ_1 and μ_2 for the set of simulations $\{i\}$ becomes slightly more convoluted than in the one-dimensional case. Due to the fact that we ultimately want to denote our set of simulations using a single index, i, were going to have to arbitrarily choose which pairs are assigned to each simulation. If the two reaction coordinates use the same number of simulations along their axes then we have a situation where each center coordinate of the biasing potential for both coordinate axes will occur $\sqrt{S}$ times (S is the total number of simulations), each with its own $\{i\}$ index as it has to be paired with the entire range of biasing potential locations for the other reaction coordinate. Luckily, the WHAM equations that we are going to minimize are not dependent on the order that we designate our simulations, just so long as we represent them in a one-dimensional array.

Due to the fact that we are only using one set of indices to designate the simulations where the biasing potentials are propagated along two independent reaction coordinates, the order in which we designate these simulations is going to be seemingly arbitrary. Luckily for us, though, this ambiguity will not influence the effectiveness of our convergence algorithm for solving the coupled WHAM equations. For the set $\{(\mu_1,\mu_2)_i\}$ with $i = 1,...,S$ we could leave μ_1 the same for $i = 1,...,\sqrt{S}$, while we increment μ_2 through it reaction coordinate. Then we can increment μ_1 for the first time at $i = \sqrt{S}+1$, and then leave it the same as we repeat the process again for μ_2 through its reaction in simulations $i = \sqrt{S}+1,...,2\sqrt{S}$. Regardless of the simulation number that we choose to assign to the pairs $\{(\mu_1,\mu_2)\}$, we can write an expression for how the centers of the biasing potentials are propagated along their reaction coordinates.

The coordinate μ defining the center of the biasing along one of two reaction coordinates will be incremented uniformly so as to sample the entire reaction coordinate. Lets say we are looking at μ_1 corresponding to the x reaction coordinate. If we run a total of S simulation sampling the entire two reaction coordinate landscape, then the number of times we have to increment μ_1 to represent the entire coordinate is $\sqrt{S}$. The first location of the biasing potential should corresponds to the midpoint of the first bin along the x coordinate axis. This give

$$\mu_1 = x_{min} + \frac{\Delta_x}{2} \tag{2}$$

which corresponds to the midway point in the first bin along the x reaction coordinate. We have defined Δ_x as the bin size. Lets denote the set of incremented μ_1 values using some other index than i so as not to confuse it with our set of simulations, say $\{\mu_{1n}\}$, where $n = 1, ..., \sqrt{S}$.

Next we define the increment along the reaction coordinate for the next value of μ_{1n}, denoted $\Delta_\mu = \frac{x_{max} - x_{min}}{\sqrt{S}}$. We can see that if the set of $\sqrt{S}$ biasing center are equally spaced, our expression for for the set of single reaction biasing coordinates is given by:

$$\mu_{1,n} = x_{min} + \frac{\Delta_x}{2} + (n-1)\,\Delta_\mu. \tag{3}$$

The set of simulations covering both reaction coordinates is created by combining each possible value of μ_1 with each possible value of μ_2. Again, the process by which we choose the pair that corresponds to a specific simulation i is arbitrary ad will not effect our methods.

2.2 Set up for WHAM algorithm with two reaction coordinates

For each sample corresponding to a simulation $\{i\}$ with a biasing potential given by EQUATION (1), the recorded reaction coordinate pair, (x, y), is binned into a histogram, with $\{n_{i,(k,l)}\}$ defining the number of samples counted in bin indexed (k, l) during simulation i. The index $\{k\}$ represents the bin index along reaction coordinate x, which the total number of bins along a reaction coordinate determined by the bin width. Likewise, index $\{l\}$ represents the bin index along reaction coordinate y. For our purposes, we define the sets of bin indexes as $\{k\} = 1, ..., K$ and $\{l\} = 1, ..., L$, where K and L are the total number of bins for each reaction coordinate. The total number of samples for a given simulation, then, is given by $N_i = \sum_{k,l} n_{i,(k,l)}$, where the summation is over the entire range of bin indices for a specified simulation i.

Our underlying goal of generating an unbiased free energy map requires an estimation of an unbiased probability distribution from the histogram data generated by the set of $\{i\}$ simulations. We will denote the binned unbiased probability distribution function as $\rho^0_{(k,l)}$, note that there is no simulation index $\{i\}$ as the unbiased distribution is the approximate combination with the biasing potentials removed.

The unbiased free energy map, $G(x_k, y_l)$ is related to the to the unbiased probability distribution as follows

$$G(x_k, y_l) = -k_B T \log \left(\frac{\rho^0_{(k,l)}}{\Delta_l \Delta_k} \right) \tag{4}$$

where $k_B T$ is the Boltzmann concept time the temperature, and $\Delta_l \Delta_k$ is the product of the widths of the bins corresponding to each reaction coordinate.

2.3 Coupled WHAM equations for a two-dimensional reaction coordinate histogram

A more detailed derivation of what follows can be found in the paper, Convergence and Error Estimation in Free Energy Calculations Using the Weighted Histogram Analysis Method by Hummer and Zhu. For our purposes here, we will simply explain the function of the relevant equations and what the variables and their indices mean.

First, given a binned unbiased probability distribution function, $\rho_{(k,l)}$, we can write an expression for a binned biased probability distribution corresponding to simulation i, $\rho_{i,(k,l)}$.

$$\rho_{i,(k,l)} = f_i c_{i,(k,l)} \rho^0_{(k,l)} \tag{5}$$

where,

$$c_{i,(k,l)} = \exp\left[-w_i (x_k, y_l)\right] \tag{6}$$

is the biasing potential for each sim at each bin, and

$$f_i = \frac{1}{\sum_{k,l} c_{i,(k,l)} \rho^0_{(k,l)}} \tag{7}$$

is a normalization constant calculated for each simulation, ensuring that $\sum_{k,l} \rho_{i,(k,l)} = 1$.

From here, a brief statistical derivation which can be found in the paper cited above by Hummer and Zhu, provides a negative log-likelihood function whose minimization is the method used here in solving the coupled WHAM equations. The function in a two-dimensional reaction coordinate system is given by:

$$A\left(\{\rho^0_{(k,l)}\}\right) = -\sum_i N_i \log(f_i) - \sum_{k,l} M_{l,k} \log\left(\rho^0_{(l,k)}\right). \tag{8}$$

We have introduced a new parameter here, $M_{l,k}$, to denote the total number of samples falling in to the bin indexed l, k over all simulations

$$M_{(k,l)} = \sum_i n_{i,(k,l)}, \tag{9}$$

where we have defined $n_{i,(k,l)}$ in section 2.2.

In order to reduce the dimensional of our minimization problem, we can rewrite EQUATION (6) as a function of the set of normalization constants $\{f_i\}$ defined above in EQUATION (5). Taking the derivative of equation (6) with respect to each bin's unbiased probability distribution and setting the derivative equal to 0 (just the process of minimizing a function), we a arrive at an expression for each bin's unbiased probability distribution in terms of the set of normalization constants:

$$\rho^0_{(k,l)} = \frac{M_{(k,l)}}{\sum_i N_i f_i c_{i,(k,l)}}. \tag{10}$$

Substituting EQUATION (8) into EQUATION (6), we arrive at the function which this paper is concerned with minimizing:

$$A\left(\{f_i\}\right) = -\sum_i N_i \log\left(f_i\right) - \sum_{k,l} M_{l,k} \log\left(\frac{M_{(k,l)}}{\sum_i N_i f_i c_{i,(k,l)}}\right). \tag{11}$$

Our convergence algorithm is going to require the the derivative of this function with respect to each parameter of the set $\{f_i\}$, so we might as well state this derivative here as well:

$$\frac{\partial A\left(\{f_i\}\right)}{\partial f_i} = N_i \left(\sum_{k,l} \frac{M_{(k,l)} c_{i,(k,l)}}{\sum_{i'} N_{i'} f_{i'} c_{i',(k,l)}} - \frac{1}{f_i}\right). \tag{12}$$

Note that the summation in the denominator is over all of the simulations, rather than just the simulation, i, of the parameter that the derivative is being taken with respect to, f_i. This is why I have denoted the simulation indices within this sum as i'.

3 BFGS with Armijo Line Search

3.1 Introduction to BFGS

Here we are trying to minimize the equations (11) and (12) by iteratively getting closer to the set $\{f_i\}$ which defines the unbiased probability distribution through equation (5). It is necessary here to think of the set of normalization factors as a vector, such that equation (11) becomes the expression for $A\left(\vec{f}\right)$, where the indices of $\vec{f}$ are given by $\{f_i\}$ with $i = 1, ..., S$. We also need to define a new set of indices to represent the iterations throughout the BFGS algorithm. Lets use the index $m = 1, 2, ...$ to denote this iterative process, with $\vec{f}^0$ representing our initial estimate, and $\vec{f}^m$ to represent the vector after an arbitrary number of iterations. We can use the same indices to keep track of the iterations of the inverse Hessian matrix, another feature of the BFGS algorithm. H^0 is simply a $[S \times S]$ identity matrix, however the subsequent H^ms are given by a mess of an expression that we will go over later. We also need to define the vector based on result of equation (12) which will serve as the gradient of the set of normalization constants through each

BFGS iteration:

$$\vec{g}^{m}\left(\vec{f}^{m}\right)=\left[g_{0}\left(\vec{f}\right),g_{1}\left(\vec{f}\right),...,g_{S}\left(\vec{f}\right)\right]^{T},\quad g_{i}\left(\vec{f}^{m}\right)=\frac{\partial A}{\partial f_{i}},\quad i=0,1,...,S. \tag{13}$$

The following steps outline the BFGS algorithm for a given H^0, $\vec{f}^{0}$: for $m = 0, 1, ...$ iterations

(1)compute the quasi-Newtonian search direction using H^m,$\vec{f}^{m}$, and $\vec{g}^{m}$

(2)determine the step size α^m and calculate $\vec{f}^{m+1}$

(3)compute H^{m+1}

3.2 Determine Quasi Newtonian Search Direction

3.3 Back Tracking Armijo Line Search

3.4 Updating the Inverse Hessian Matrix

4 Results