WHAM Convergence Module

Cameron Kopp

December 2022

1 Data Generation

1.1 Method: SampleConstructor.xp1nverseSinSq(x)

The goal here is generate random data from the probability distribution function (FIGURE 2 (a))

$$\rho(x) = \frac{1}{\left(1 + (1-x)^2\right)} \frac{\sin^2(2x) + 2}{3} \tag{1}$$

for a set of biased simulations $\{i\}\,(i = 0, 1, 2, ..., S)$, where

$$\rho_i(x) = \sqrt{\frac{k}{2\pi}} * e^{-\frac{k}{2}(x-\mu_i)^2} * \rho(x) \tag{2}$$

and

$$\mu_i = x_{min} + i\left(\frac{x_{max} - x_{min}}{S}\right) \tag{3}$$

with C as a normalization constant. If we define the biasing potential for a simulation i, which will be centered around μ_i (equation (3) above), as

$$V_i'(x) = \sqrt{\frac{k}{2\pi}} * e^{-\frac{k}{2}(x-\mu_i)^2} \tag{4}$$

then we see that:

$$\rho_i(x) = C\rho(x) * V_i'(x) \tag{5}$$

Figure 1 illustrates the relation between equations (1), (4) and (5) for different values of μ_i

2 BFGS Algorithm with Armijo Line Search

Here we are trying to minimize a function $A\left(\vec{f}\right)$ by altering the values of the indices $\{f_i\}\,(i = 0, ..., S)$, of the vector $\vec{f}$. In the context of the wham equations, we are minimizing equation(19) of the Hummer and

Zhu paper by altering the vector of the logarithms of the normalization constants for a simulation i. We will use the index $l = 0, 1, 2, ...$ to label the iteration number of the BFGS algorithm, such that $\vec{f}^0$ will represent the input estimate, $l = 0$, and $\vec{f}^l$ will represent an arbitrary estimate at iteration l. We will use $A\left(\vec{f}^l\right)$ to represent equation (6) evaluated for the set of inverse partition functions $\vec{f}^l$, with the other variable (the $\vec{f}^J$ or $\vec{f}^K$ that we are not minimizing over) held constant. H^l defines an $(S \times S)$ matrix for iteration l of the BFGS algorithm, which is the inverse Hessian matrix and will be used to determine the search direction for iteration l where $H^{l=0} = I^S$, the $S \times S$ identity matrix. The following vector also has to be defined:

$$\vec{g}^l\left(\vec{f}^l\right) = \left[g_0\left(\vec{f}\right), g_1\left(\vec{f}\right), ..., g_S\left(\vec{f}\right)\right]^T, \quad g_i\left(\vec{f}^l\right) = \frac{\partial A}{\partial f_i}, \quad i = 0, 1, ..., S \tag{6}$$

where $\frac{\partial A}{\partial f}$ is defined above in equation (7) or (8). The following steps outline the BFGS algorithm for a given H^0, $\vec{f}^0$: for $l = 0, 1, ...$

(1)compute the quasi-Newtonian search direction

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

(3)compute H^{l+1}

2.1 Search Direction (Step 1)

The fist step in the BFGS algorithm is computing the search direction $\vec{p}$, which sets the direction that we will look in for the new $\vec{f}^{l+1}$. The BFGS calculates the search direction with the equation

$$\vec{p} = -H^l \vec{g}^l. \tag{7}$$

Intuitively, it should make sense that $\vec{p}$ is proportional to the gradient $\vec{g}^l$. If we were using Newton's method, our search direction would be $\vec{p} = \vec{g}\left(\frac{\partial^2 A}{\partial f^2}\right)^{-1}$. Rather than computing the second derivative with each step, the BFGS algorithm approximates it with H^l.

2.2 Armijo Backtracking Line Search and $\vec{f}^{l+1}$ (Step 2)

The Armijo backtracking line search algorithm requires the inputs α^0, and $\tau, \beta \in (0, 1)$, for which I've been using $\alpha^0 = 2$, $\tau = 0.5$, and $\beta = 0.1$. The goal is to compute a $\Delta\vec{f}$ such that $\vec{f}^{l+1} = \vec{f}^l + \Delta\vec{f}$, with $\Delta\vec{f} = \alpha^m \vec{p}$. This method imposes the following condition

$$A\left(\vec{f}^l + \alpha^m \vec{p}\right) \leq A\left(\vec{f}^l\right) + \alpha^m \beta \left[\vec{g}^l\right]^T \vec{p} \tag{8}$$

to require that we achieve a reduction in $A\left(\vec{f}^l\right)$ during this step that is at least a fraction β of the reduction expected in Newton's Method. Given α_0, τ, β, the algorithm is:

Until $A\left(\vec{f}^l + \alpha^m \vec{p}\right) \leq A\left(\vec{f}^l\right) + \alpha^m \beta \left[\vec{g}^l\right]^T \vec{p}$:

(1) set $\alpha^{m+1} = \tau\alpha^m$

(2) increment m by 1

Once α^m is found, $\vec{f}^{l+1}$ is calculated by

$$\vec{f}^{l+1} = \vec{f}^{l} + \alpha^m \vec{p} \tag{9}$$

2.3 H^{l+1} calculation (Step 3)

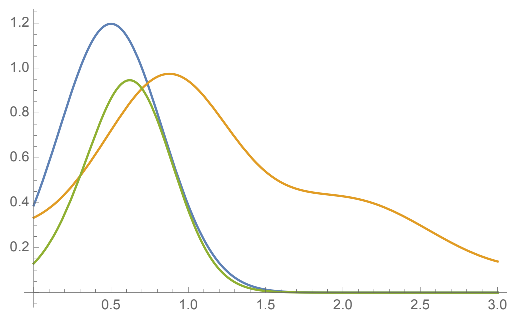
We have to define the following vectors:

$$\vec{s} = \vec{f}^{l+1} - \vec{f}^{l}, \quad \vec{y} = \vec{g}^{l+1} - \vec{g}^{l}. \tag{10}$$

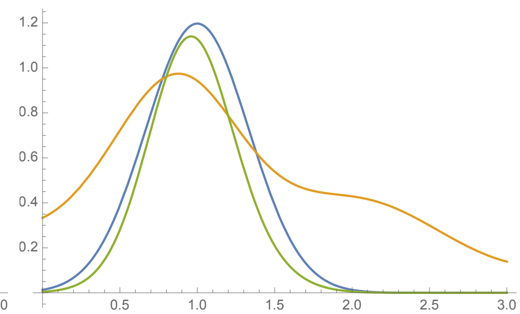
We can then calculate the updated inverse Hessian for the BFGS directly:

$$H^{l+1} = \left(I^n - \frac{\vec{y}\,\vec{s}^T}{\vec{y}^T\vec{s}} \right) H^l \left(I^n - \frac{\vec{y}\,\vec{s}^T}{\vec{y}^T\vec{s}} \right) + \frac{\vec{s}\,\vec{s}^T}{\vec{y}^T\vec{s}} \tag{11}$$

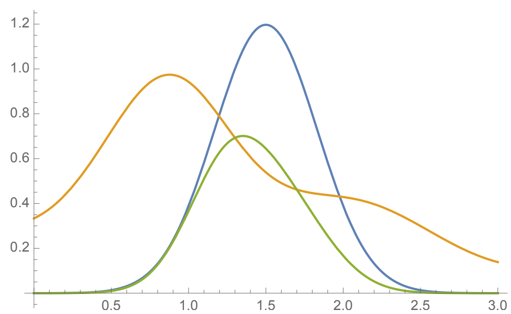
The form of this equation is messy, but it comes from a relatively simple argument which maintains that the inverse Hessian satisfies the secant condition,$\vec{s} = H^{l+1}\vec{y}$, and then defines a quadratic approximation of the function being minimized, in our case $A\left(\vec{f}^{l}\right)$. FIGURE 2 demonstrates the convergence of this algorithm applied to histogram data generated by the method described in section 1.1.



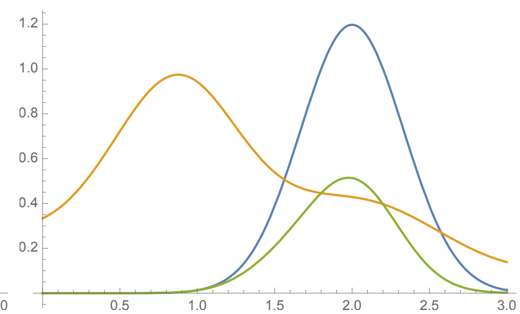
(a) $\mu_i = 0.5$



(b) $\mu_i = 1.0$

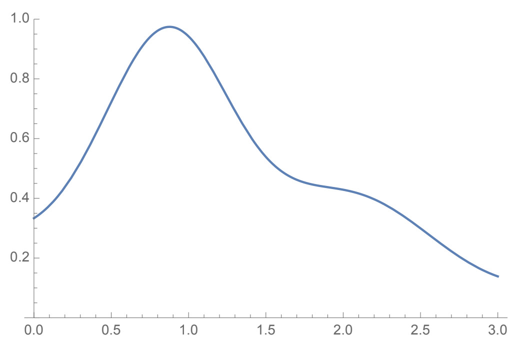


(c) $\mu_i = 1.5$

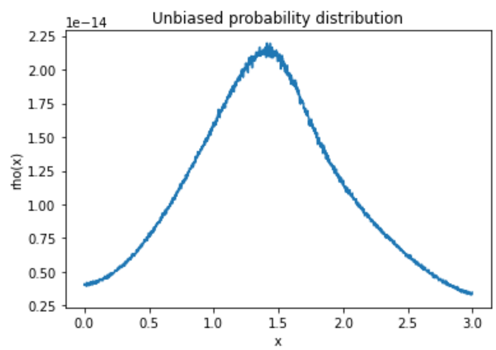


(d) $\mu_i = 2.0$

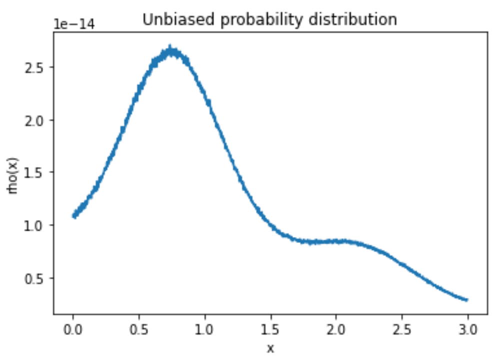
Figure 1: Plots of ORANGE : $\rho(x)$, BLUE : $V_i'(x)$, and GREEN : $\rho_i(x) = \rho(x) * V_i'(x)$ for simulations with incremented values of μ_i (spring constant: $k = 9$)



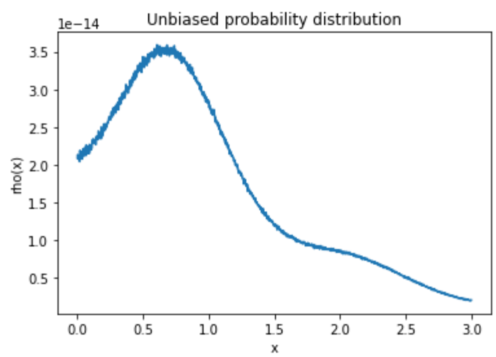
(a) The known unbiased potential $\rho\,(x)$



(b) $l = 1$



(c) $l = 5$



(d) $l = 10$

Figure 2: The unbiased distribution (a), which is the target of the BFGS convergence. The estimated unbiased distribution given after loop iteration l, illustrating the convergence (b-c).