Markov Chain Monte Carlo

Casey Sobecks

May 2022

1 Monte Carlo

The Monte Carlo (MC) technique is one that utilizes random sampling [1]. This method is great when trying to approximate things like integrals. One of the previous homeworks in the semester, I utilized it to find the area of a circle with my random number generator. Numerical recipes also provides some code to integrate other functions. One ideal integral is the gaussian, which regular Monte Carlo can do very successfully. This method, however, is not always very efficient in time or resources, since it requires a new random number be generated every iteration. For this reason, there is a variation on Monte Carlo called Markov Chain Monte Carlo (MCMC). This techniques utilizes random sampling, but rather than sample the entire region in consideration uniformly, the goal is to visit certain points with a probability proportional to some given distribution function [1]. This method is very useful when trying to find things like the minimum of some function. A great example of this is the Rosenbrock function which is defined as

$$f(x,y) = (a-x)^2 + b(y-x^2)^2. \tag{1}$$

Inspecting this equation, we can see that the minimum is at the point (a, a^2). Numerically, however, this is expensive to find via regular Monte Carlo. Using MCMC, the minimum can be found with relative ease because of the restriction on step choices. Given a random starting point, the markov chain will pick points that move downhill towards the minimum. This method is especially useful because it can overcome the flat regions of the function and find the true minimum. The downside of this method is that it will correlate data since the current point in the chain depends on previous points. Therefore, this method is most successful when dealing with functions or spaces that are very hard to sample directly. Another part of this method is that it requires some "burn in" time to create an accurate picture of the function.

2 Updating via Feynman path integral

In quantum mechanics, a value called the transition amplitude is defined as the probability amplitude for a particle to go from some initial position x_i at an initial time t_i to a final position x_f at some final time t_f. This term is defined as

$$\langle x_f(t_f)|x_i(t_i)\rangle\,. \tag{2}$$

By solving for this term, the path of a particle can be found. This value can be found by using a path integral. A path integral looks at all the possible paths that the particle could follow to from (x_i, t_i) to get to (x_f, t_f) [2]. Ultimately, this path integral is proportional to the classical action of the system, or

$$\langle x_f(t_f)|x_i(t_i)\rangle \sim \exp\left(-\frac{iS}{\hbar}\right) \tag{3}$$

where

$$S = \int \mathcal{L}(x,\dot{x})dt. \tag{4}$$

The action can be used as the condition for MCMC updates to find the actual path of a particle. Specifically, the updates will be done via the Metropolis algorithm. For this algorithm, the points that are sampled are not unrelated, but connected in a Markov chain, or a sequence of points that are locally correlated and can be shown to visit every point with some probability proportional to the probability distribution [1]. This method can be realized using a random number generation to determine if an update should be accepted or not. The probability that will be used is the change in the action of the system, or δS.

3 Simple Harmonic Oscillator

The first example considered is the simple harmonic oscillator. Specifically, a one dimensional system with periodic boundary conditions is considered since it aligns with the Metropolis algorithm condition. The action for this system can be written as

$$S = \int d\tau \left(\frac{m\dot{x}^2}{2} + \frac{m\omega^2 x^2}{2}\right). \tag{5}$$

The change in the action between points can then be found as

$$\begin{aligned}\delta S_L = \frac{m}{2}\sum_{n=0}^{N-1}[&(x_{n+1}-x_n^{new})^2 - (x_{n+1}-x_n)^2 \\ &+ \frac{\omega^2}{4}(x_{n+1}+x_n^{new})^2 - \frac{\omega^2}{4}(x_{n+1}+x_n)^2].\end{aligned} \tag{6}$$

```
dS=(pow((site[tau+1]-new_site[tau]),2.0)
   +0.25*omega*omega*pow((site[tau+1]+new_site[tau]),2.0))
   -(pow((site[tau+1]-old_site[tau]),2.0)
   +0.25*omega*omega*pow((site[tau+1]+old_site[tau]),2.0));
dS=(m/2.0)*dS;
```

Figure 1: Simple Harmonic Oscillator Code

```
u=drand48();
if(u<exp(-dS)){
    site[tau]=new_site[tau];
}
else{
    site[tau]=old_site[tau];
}
```

Figure 2: Metropolis Update

This equation shows that after picking some new point x_n^{new}, the difference between this value and the previous point is found. After summing over all terms, the total change in action is found and then can be used. In code, this is implemented and shown in 1. After this value is calculated, the update can be made as shown in Fig. 2. As seen in this figure, the update has a small chance of happening if the change in action is large. If the action is very small or even negative, then there is a greater change that the update will be accepted. Following this method, the path for a particle in a simple harmonic oscillator potential, given ω and m, can be found. Two cases are shown in Figs. 3 and 4. These figures match our expectations. Where the potential is zero, the particle spends most of its time. With this solution, we can expand to one other case.

4 Double Well

Now we will consider the double well given by Grasinger [3]. The equation for the potential is

$$U = ax^4 - bx^2 - fx \tag{7}$$

and the shape of the potential is shown in Fig. 5. When $f = 0$, the potential is symmetric and when $f \neq 0$ it is asymmetric. This is simply a different potential and so the same path integral can be used. The change in action is now change as shown in Fig. 6. After doing this, the code can run the same. The output of this code is shown in Figs. 7 and 8.

As seen above, the path integral update matches the MCMC update that is done in the paper. The update is such that the particle would be stuck in the left well and wouldn't explore the right well. This means that there is a need to change the way that updates are made. The paper goes into this discussing how to use group action [3]. I didn't explore this, but created my code to match the the standard MCMC done in Fig. 5

5 Conclusion

In conclusion, the method of Markov Chain Monte Carlo is an effective way of finding ground state energies of spectra or paths in a given potential. It is more efficient than regular monte carlo for difficult functions and therefore is an important tool for research. While it is not without its own issues, changes can be made to make this method more robust. It is a current area of research that is alive and well.

References

[1] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes 3rd Edition: The Art of Scientific Computing.* Cambridge University Press, 3 edition, 2007.

[2] Jun John Sakurai. *Modern quantum mechanics; rev. ed.* Addison-Wesley, Reading, MA, 1994.

[3] Matthew Grasinger. Group action markov chain monte carlo for accelerated sampling of energy landscapes with discrete symmetries and energy barriers. *arXiv*, 2022.

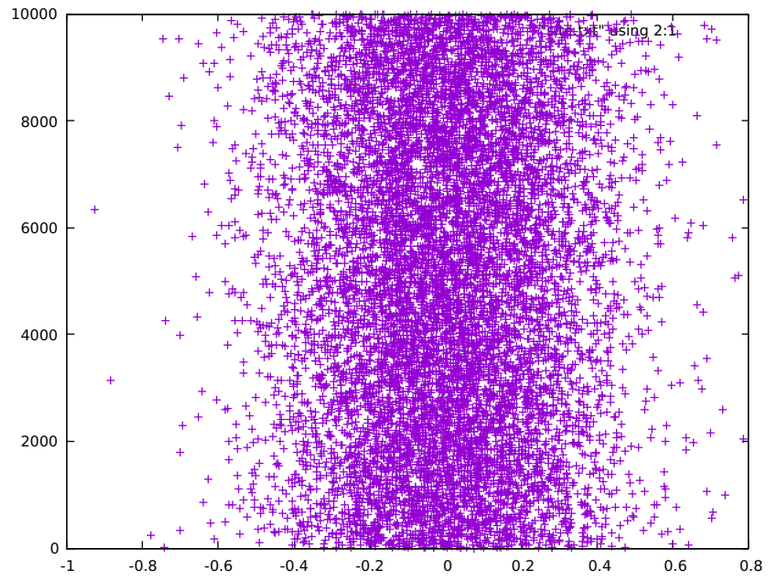

Figure 3: Simple harmonic oscillator $m = 10$, $\omega = 2$

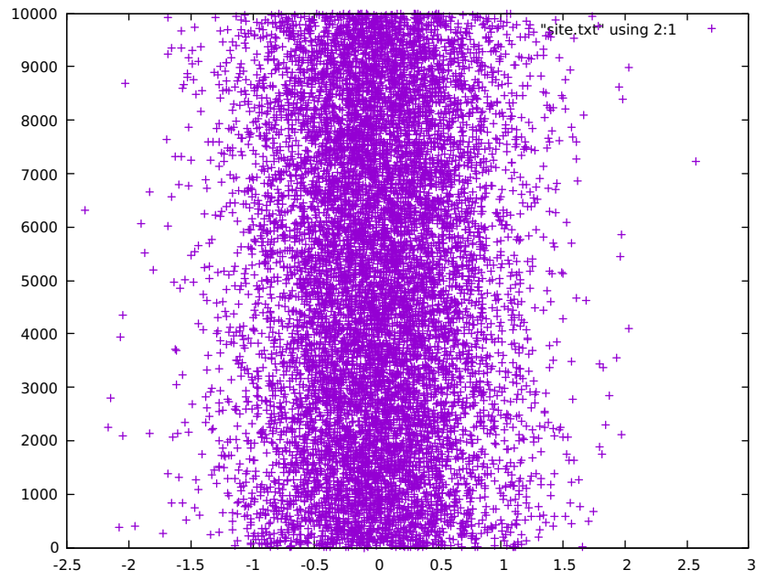

Figure 4: Simple harmonic oscillator $m = 0.5$, $\omega = 100$

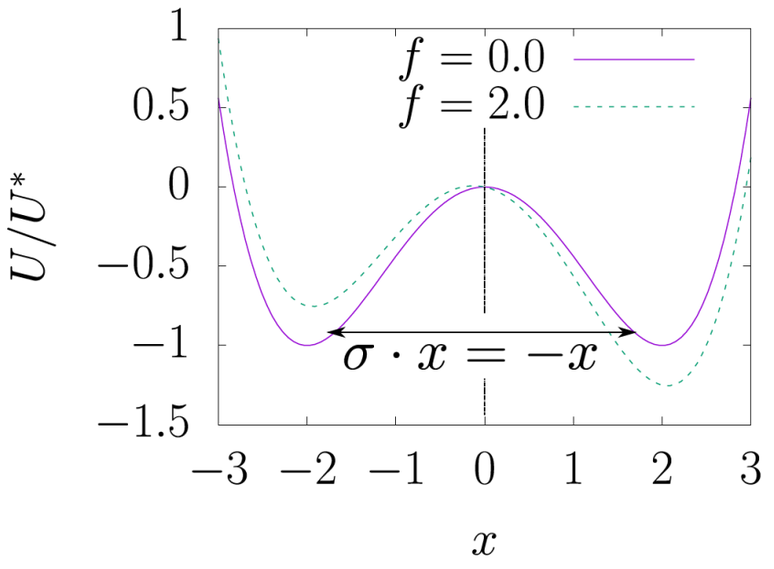

Figure 5: Double Well Potential

```
dS=(1/2)*(-a*(pow((site[tau+1]+new_site[tau]),4.0))
        +b*((pow((site[tau+1]+new_site[tau]),2.0)))
        +f*(site[tau+1]+new_site[tau])
        -a*(pow((site[tau+1]+old_site[tau]),4.0))
        +b*((pow((site[tau+1]+old_site[tau]),2.0)))
        +f*(site[tau+1]+old_site[tau]));
```

Figure 6: Double well change in energy

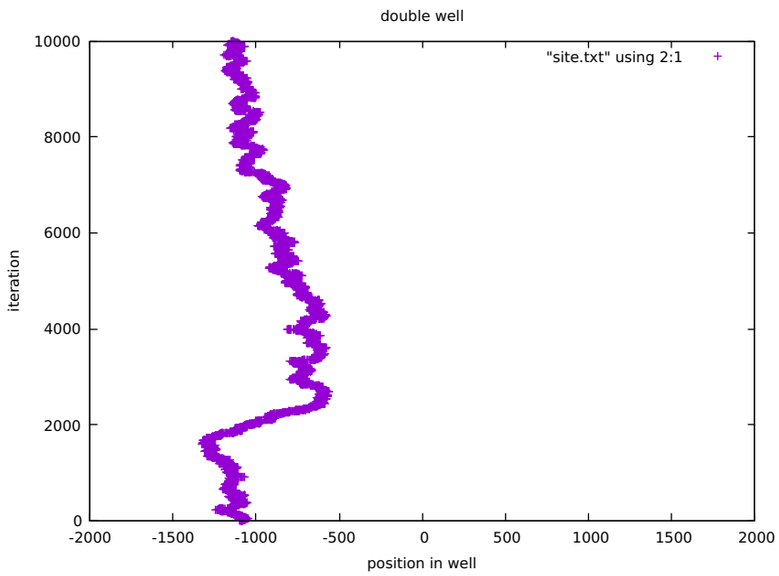

Figure 7: Double well with $f = 0$

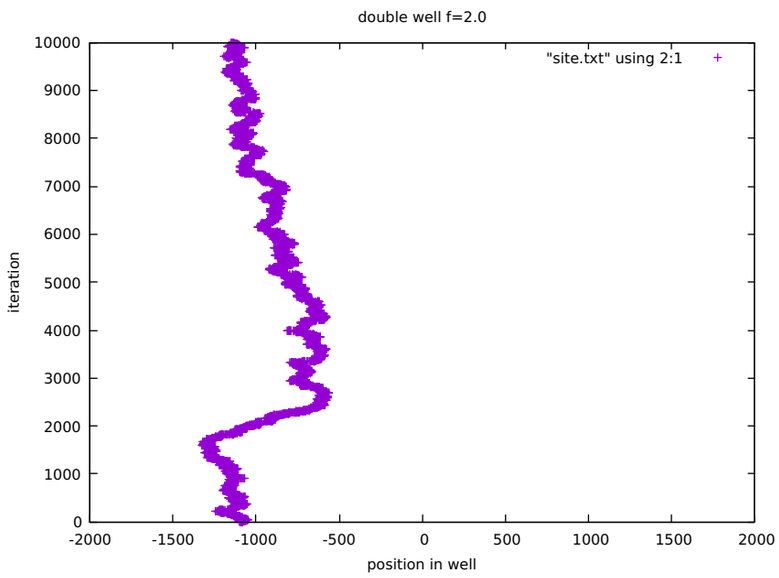

Figure 8: Double well with $f = 2$