Statistical Analysis for Monte Carlo Simulations

Markov Chains

- A Markov Chain is a random walk in state space
  - $S_1 \rightarrow S_2 \rightarrow S_3$ …
- The transition probability that takes us from one state to the next is a stochastic matrix:

$$P(s \rightarrow s') \geq 0 \; and \; \sum_i P_i = 1$$

- The probability of states at a given iteration

$$f_{n+1}(s') = \sum_s f_n(s) P(s \rightarrow s')$$

- The stationary states of this are eigenstates of P
- If the random walk is ergodic, there will be a unique equilibrium state

The Monte Carlo methods we will learn about are largely Markov Chains

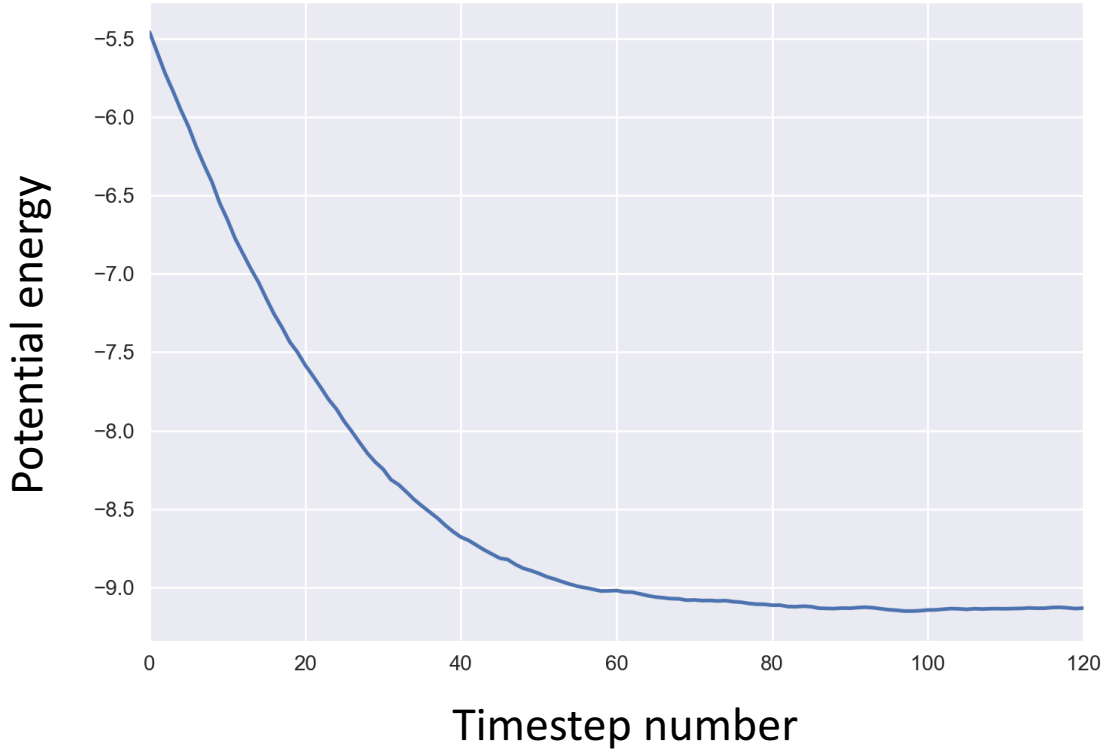
- Need to be able to understand the statistical data from them
- A classic example of a Markov Chain is Brownian Motion
  - Imagine scattering some larger particles in a fluid
  - The initial positions are random
  - The positions of the particles evolve by collisions with the rest of the fluid
  - After some equilibration, the position of the particles satisfies the diffusion equation

Example Code

- In the Statistics directory of the git repository (https://github.com/lkwagner/Telluride School2017.git)
- The code RandomWalk.py simulates a random walk of N particles in a quartic potential
- Parameters such as the time step of the propagation, the number of walkers and the total number of time steps can be changed in the top of the file

Example Code continued

- Run the application with 128 walkers and a timestep of 0.01 to generate a time series like the following:
- This is can be divided into two phases, equilibration and equilibrium



Exercise #1

- Create an algorithm to divide the time series into equilibration and equilibrated phases
  - Hint: start from the end of the calculation

Calculating properties

- Now that equilibrium phase is identified, want to calculate statistics
  - Sample Mean: $\bar{a} = \sum_i \frac{a_i}{N}$
  - How well is this mean known?
    - If all samples from the Markov chain were uncorrelated:
      - standard deviation of the samples: $\sigma = \sqrt{\frac{1}{N-1}\sum_i (a_i - \bar{a})^2}$
      - standard error of the mean: $err(\bar{a}) = \sigma/\sqrt{N}$

Calculating properties

- Now that equilibrium phase is identified, want to calculate statistics
  - Sample Mean: $\bar{a} = \sum_i \frac{a_i}{N}$
  - How well is this mean known?
    - If all samples from the Markov chain were uncorrelated:
      - standard deviation of the samples: $\sigma = \sqrt{\frac{1}{N-1}\sum_i (a_i - \bar{a})^2}$
      - standard error of the mean: $err(\bar{a}) = \sigma/\sqrt{N}$
  - Unfortunately, the data is NOT uncorrelated
  - Knowing where the walkers were at step n gives a good idea of where they will be at step n+1
  - Need to know the time it takes to “forget” this history (autocorrelation time)

Two commonly used methods of handling this

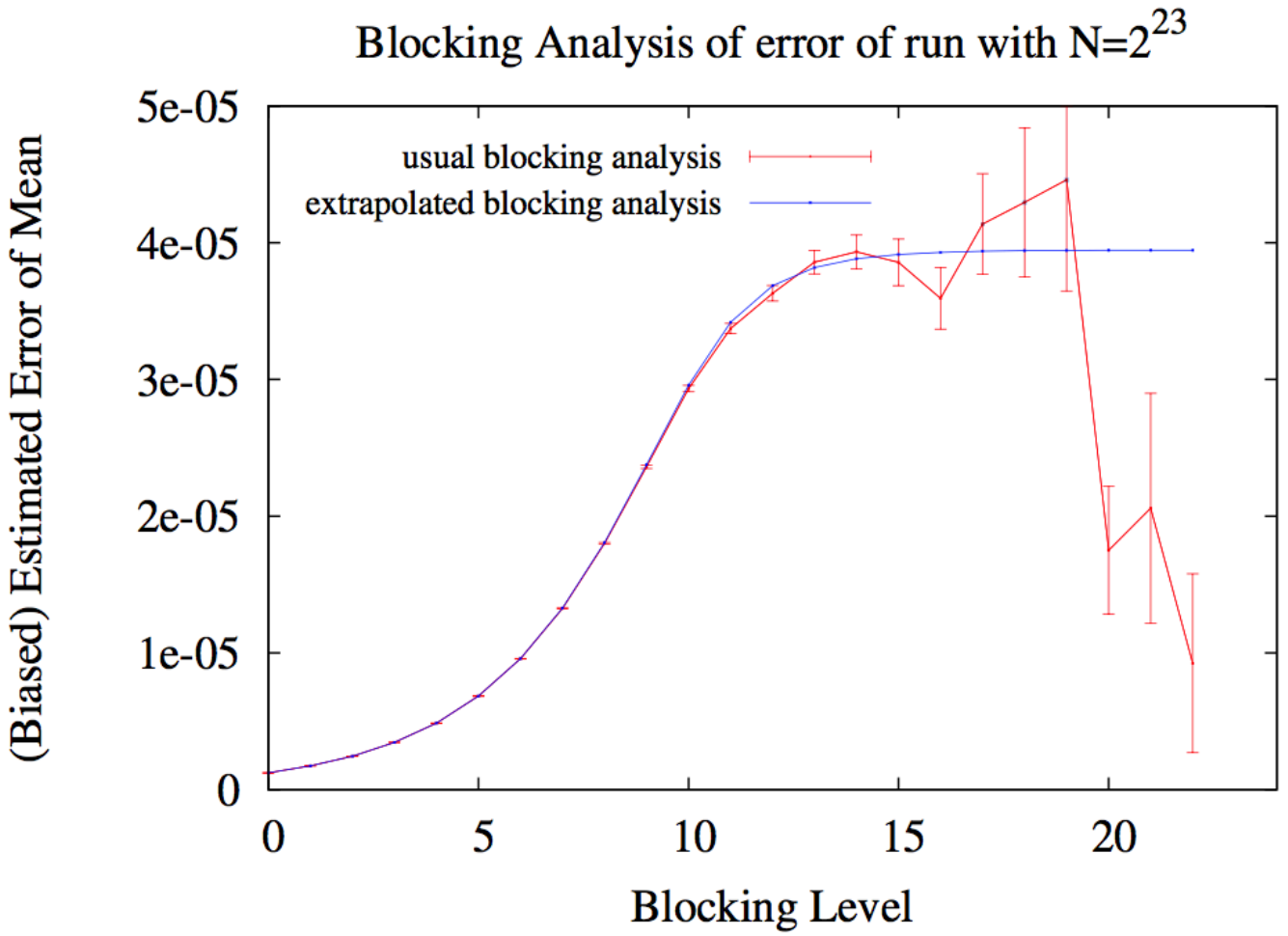
- Blocking
- Direct use of autocorrelation function

Blocking method

- Group data into successively larger blocks and compute the error using this blocked data
  - For correlated data this will increase until roughly block size equals the autocorrelation time
  - Specifically, calculate $\frac{1}{N_b(N_b-1)}\sum_{i=1}^{N_b}(m_i-\bar{a})^2$
    - Where the number of samples in each block is growing (doubling is convenient) and m_i is the average of the data in each block
    - Can also estimate the error in the error estimates as: error estimate / $\sqrt{2(N_b-1)}$

Blocking method example (courtesy C. Umrigar)

Blocking Analysis for error of mean of autocorrelated variables



Direct calculation of autocorrelation function

- Calculate the autocorrelation function: f(k) = $\frac{\sum_{i=1}^{N-k}(a_i-\bar{a})(a_{i+k}-\bar{a})}{\sum_{i=1}^{N}(a_i-\bar{a})^2}$
  - Measure of how correlated data is with data k units of time away
- Define autocorrelation time: $\kappa = 1 + 2\sum_{i=1}^{N} f(k) \approx 1 + 2\sum_{i=1}^{f(k)>0} f(k)$
- Now the number of samples used in determining the error is reduced by a factor of κ
- $err(\bar{a}) = \sigma/\sqrt{N/\kappa}$

Exercise #2

- Implement either the blocking method or the autocorrelation method to determine the average and error in that average for data from the equilibrated part of a Markov chain

Exercise #3

- Combine the algorithms from exercises 1 and 2 and analyze the data from RandomWalk.py
  - Perform an initial calculation with 128 walkers, 2000 steps and a timestep of 0.005
  - Using the same number of walkers, how many steps would be necessary to get the same error bar with a timestep of 0.0025?
  - If you double the number of walkers, how many steps are necessary to get the same error bar?

Digression on ergodicity

- In everything previous we have assumed ergodicity
  - Basically that given enough propagation time, a walker can come arbitrarily close to any point in space
  - Must always be aware of the possibility that this could be violated

Exercise #4

- Alter RandomWalk.py so that walkers are initially distributed from -4 to 4 (rather than 2-3, see line ~45)
- For various values of the timestep is it possible to enter into a nonergodic calculation?
  - Look at timeseries of energies
  - Use histogram of final positions (example code at end of file)

A set of advice from a D. Ceperely lecture…

- "Shouldn't the energy settle down to a constant"
  - NO. It fluctuates forever. It is the overall mean which converges.
- "The cumulative energy has converged".
  - BEWARE. Even pathological cases have smooth cumulative energy curves.
- "Data set A differs from B by 2 error bars. Therefore it must be different".
  - This is normal in 1 out of 10 cases. If things agree too well, something is wrong!
- "My procedure is too complicated to compute errors"
  - NO! NEVER! Run your whole code 10 times and compute the mean and variance from the different runs. If a quantity is important, you MUST estimate its errors.

Example on this last point

- Imagine you have performed a series of Monte Carlo calculations of the energy of a dimer as a function distance between the atoms
- You want to fit this to a Morse potential to determine the equilibrium lattice constant and vibrational frequency:
  - $E(r) = ae^{-2b(r-d)} - 2ae^{-b(r-d)} + c$
  - r_e=d
  - $\omega = 2\sqrt{\frac{2a}{\mu}} - \frac{b^2}{\mu}$
- Find data in cn-energies.csv
- Initial fitting routine in fit-cn.py

Exercise #5

- How would you go about calculating error estimates on the equilibrium separation and vibrational frequency?
- If you were to perform another calculation, what distance would you use to decrease the errors the most?

- Hint: “Run your whole code 10 times and compute the mean and variance from the different runs. If a quantity is important, you MUST estimate its errors.”
  - Can’t do this, but can it be done in a synthetic way?