Probablistic Interpretation

Outline for remainder:

Thursday: Scalar fields

Recall from last time:

$$\langle f(x_1 \dots x_N) \rangle = \int dx_1 \dots dx_N \, f(x_1, \dots x_N) p(x_1, \dots x_N) \quad \text{with} \quad p(x_1 \dots x_N) = \frac{e^{-S_E(x_1 \dots x_N)}}{\int dx_1 \dots dx_N e^{-S(x_1 \dots x_N)}} \quad \text{(probability density)}$$

We want to generate our paths (eventually gauge fields) based on this distribution. Important sampling is necessary since we can only compute a finite number of paths. Formally the path integral is over infinite.

Markov Chain:

Start with arbitrary configuration then construct a stochastic sequence of configurations that eventually follows the equilbrium distribution.

$x_0 \to x_1 \to x_n$ (updating the configuration as we go through the chain)

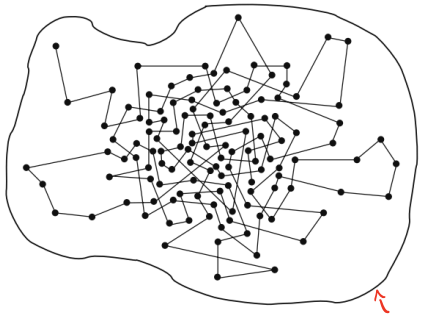
We characterize a Markov process with a transition probability

$$P(x_n = x' | x_{n-1} = x) = T(x'|x)$$

The transition probability has the following properties

- $T(x'|x) \geq 0$ for all x, x' and $\sum_x T(x'|x) = 1$ — $T(x'|x)$ is a probability distribution in x' for any x
- $\sum_x P(x) T(x'|x) = P(x')$ for all x' — Equilbrium distribution should be preserved
- $T(x|x) > 0$ for all x — "aperiodicity", Markov process does not get trapped in cycles
- If X is a non-empty proper subset of states, there exists two states $x \in X$ and $x' \notin X$ such that $T(x'|x) > 0$. — or in subset of states "ergodicity"

Can eventually reach all states



Space of all configuration

Example of MCMC algorithm: Metropolis

The metropolis algorithm is one of the simplest (though not the best) for generating configurations. The algorithm for randomizing x_j at the j^{th} site is:

- generate a random number ζ, with probability uniformly distributed between $-\epsilon$ and ϵ for some constant ϵ.
- replace $x_j \to x_j + \zeta$ and compute the change in the action ΔS caused by this change
- If $\Delta S < 0$, retain new value and move to the next site
- if $\Delta S > 0$, generate a random number η uniformly distributed between 0 and 1; retain new value x_j if $\exp(-\Delta S) > \eta$, otherwise restore the old value.

If ϵ is too large, then most new x_j will be rejected. If too small, most will be accepted. Want ϵ tuned so that 40-60% of x_j are changed in each pass. Then ϵ is of order typical of quantum fluctuations in theory

This can be generalized to gauge fields also. Fermions are harder!

Exercise: Run through the provided jupyter notebook that runs the metropolis algorithm for the harmonic oscillator.

Analyzing the Data

Based on the spectral decomposition of the two-point function, we can perform a fit to obtain amplitudes or ground/excited state energies. However in this scenario, we will be extracting the energy splitting via a ratio of correlators

$$\Delta E_n = \log(C(t)/C(t+a)) \xrightarrow{\text{large } t} (E_1 - E_0)a$$

* Try to derive

Since the correlator, and therefore the energy, are random variables, we need to understand the statistics.

The data generated is the result of a (computer-) time series in our Monte Carlo simulation; therefore, there is a high chance that observables are correlated. This so called autocorrelation leads to non-vanishing auto-correlation function

$$C_X(X_i, X_{i+t}) = \langle (X_i - \langle X_i \rangle)(X_{i+t} - \langle X_{i+t} \rangle) \rangle = \langle X_i X_{i+t} \rangle - \langle X_i \rangle \langle X_{i+t} \rangle$$

* Maybe discuss integrated autocorrelation time

For a Markov chain in equilbrium, the autocorrelation function only depends on the (computer time) separation

$$C_X(t) = C_X(X_i, X_{i+t})$$

Typically, the normalized correlation function Γ_X exhibits exponential behavior asymptotically for large t

$$\Gamma_X = \frac{C_X(t)}{C_X(0)} \sim \exp\left(\frac{-t}{\tau_{X,exp}}\right)$$ where τ_X is the exponential autocorrelation time for X. The autocorrelation time is the supremum of values $\tau_{exp} = \sup_X \tau_{X,exp}$.

Its important to understand autocorrelations as they lead to errors $\mathcal{O}\left(\exp\left(\frac{-t}{\tau_{exp}}\right)\right)$

* errors grow like $\frac{\sigma_X^2}{N} 2\tau_{X,int}$ $\tau_{int} = \frac{1}{2} + \sum_{t=1}^{N} \Gamma_{X(t)}$

In the metropolis algorithm, two successive paths might be very similar (highly correlated), thus limiting the information gained. We can limit the effects by only including every N_{cor}^{th} update which you've seen in the notebook.

→

We can also reduce autocorrelations by binning. If we have N time series measurements of some observable, we can divide the measurement into B bins and average amongst those bins

Resampling methods

Recall that we want to compute $\overline{\Delta E}$ and $\sigma_{\Delta E}$. Since we integrate over N_{cf} configurations, we only have one data point. We could generate multiple sets of N_{cf} configurations, but this is computationally expensive. This is where resampling comes in.

Stastical Bootstrap: Given a set of N data, we are interested in some observable θ. We recreate from the original sample repeatedly other samples by choosing randomly N data out of the original set. Assume we have done this K times and thus K sets of N data

$$\tilde{\theta} = \frac{1}{K}\sum_{K=1}^{K}\theta_K \qquad \sigma^2_{\hat{\theta}} = \frac{1}{K}\sum_{K=1}^{K}(\theta_K - \tilde{\theta})^2$$

Jackknife: Start with data set of size N and an observable θ. One now constructs N subsets by removing the n^{th} entry of the original set $(n=1,\dots,N)$ and determines θ_n for each set.

$$\tilde{\theta} = \frac{1}{N}\sum_{n=1}^{N}\theta_n \qquad \sigma^2_{\hat{\theta}} = \frac{N-1}{N}\sum_{n=1}^{N}(\theta_n - \hat{\theta})^2$$

Exercise: Apply the above methods to find the energy splitting for the harmonic oscillator

Fitting Correlated Data

In the above example, we computed the energy splitting through simple arithmetic. In practice, we often perform fits to determine energies/amplitude/matrix elements We do this by minimizing

$$\chi^2 = \sum_{n_t,n_t'=n_{min}}^{n_{max}} (C(n_t) - f(n_t))\, w(n_t,n_t')\, (C(n_t') - f(n_t'))$$

$f(t)$ is our fit/hypothesis function e.x. $f(t) = \sum_n a_n e^{-E_n t}$

where $w(n_t,n_t') = Cov^{-1}(n_t,n_t') \rightarrow Cov(n_t,n_t') = \frac{1}{N-1}\langle (C(n_t) - \langle C(n_t)\rangle_N)(C(n_t') - \langle C(n_t')\rangle_N)\rangle_N$ is the inverse covariance matrix. This takes into account correlations along time slices.

We can also alter the measured covariance matrix in the following ways:

- Ignoring cross correlations: $w(n_t,n_t') = \delta_{n_t,n_t'}/\sigma(n_t)^2$
- Applying SVD cuts
- Applying Shrinkage (linear/shrinkage)

* high correlations caused by: Parameters/Algorithms/Excited state contamination/FV effects/Smearing/Noise dominated data

Only want part of action that depends on x_j. Focusing on the kinetic term

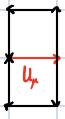
$$S_{kin} = \frac{1}{2a}(x_{j+1}-x_j)^2 + \frac{1}{2a}(x_j - x_{j-1})^2$$

$$= \frac{1}{2a}(x_{j+1}^2 - 2x_j x_{j+1} + x_j^2) + \frac{1}{2a}(x_j^2 - 2x_j x_{j-1} + x_j^2)$$

$$= \frac{1}{2a}(2x_j^2 - 2x_j x_{j+1} - 2x_j x_{j-1}) \qquad \text{ignoring the terms that dont contain } x_j$$

$$\boxed{= \frac{1}{a} x_j (x_j - x_{j+1} - x_{j-1})}$$

This is computationally cheaper than computing the entire action before and after. For the gauge theory case, which would be significantly more expensive, we only compute the following staple of link variables



$$S_{wil} = U_\mu(x)\Gamma_\mu(x) = U_\mu(x)\left(U_\nu(x+\hat{\mu})U_\nu^\dagger(x+\hat{\nu})U_\nu^\dagger(x) + U_\nu^\dagger(x+\hat{\mu}-\hat{\nu})U_\mu^\dagger(x-\hat{\nu})U_\nu(x-\hat{\nu})\right)$$