NOTES: Quantum Harmonic Oscillator On a Computer

Sahaj Porwal

December 3, 2020

0.1 First Implementation 30/9/2020

Preliminary code structure:

1. *main* method for main simulation and recording observables(WIP)
2. *harmonic_osc* method for iteration(WIP)
3. *calc_action* method for calculation of action(WIP, need to check if correct)
4. *calc_ham* method for calculation of hamiltonian
5. *calc_delh* method for calculation of derivative of hamiltonian wrt x

Some details of methods need to be checked.

0.2 Meeting 5/10/2020

Feedback: Split up program into Classes, which individually implement different parts of the code:

1. Action
2. State
3. Metropolis
4. Statistical Measurements

Important: Difference between Action and Hamiltonian(use the former not the latter) Re-scale x & t Write unit tests x updates either + or - the lattice spacing 50-80% acceptance rate is good

0.3 Second Implementation

UPDATE: New structure of code:

1. 'main' method for simulation.
2. 'Action' class to implement action of various systems. Contains the harmonic action, and the change in harmonic action methods
3. 'Observe' class to implement statistical measurements. Contains variables to store sum of positions and sum of squared positions for mean and variance calculations.
4. 'Metropolis' class to implement Metropolis time-step. Implement one single metropolis time step

Implemented a single lattice point quantum harmonic oscillator.

0.4 Meeting 12/10/20

Feedback:

1. Need to implement a lattice of points, each of which is dependent on the nearby points for it's action calculation and hence the metropolis step.
2. Need to calculate the autocorrelation function and the correlation function.
3. Need to plot theoretical predictions and compare with them. Gaussian distribution of positions, Slope of correlation function, etc.
4. Consider creating a State class, which would hold the state of the system. This can be passed to methods so that individual positions do not need to be passed around.

0.5 Improvements to Second Iteration

1. Created a State class to store lattice properties and state.
2. Now the action is implemented independently for each point on the lattice
3. Implemented Autocorrelation and Correlation functions.
4. Plotted the theoretical Gaussian and compared histogram of positions.

0.6 Meeting 19/10/20

1. Need to get potentials from the proposed change in x in the metropolis step.
2. Need to implement periodic boundary conditions (ie. if the index goes beyond length of lattice, wrap around to the first lattice point).
3. Plot the autocorrelation function and observe the noise.

0.7 Implementing observables and Plotting

1. Implemented Autocorrelation and Correlation functions.
2. Plotted the histogram of positions.
3. Calculation of the mean position and the mean squared position implemented

0.8 Meeting 26/10/20

1. Coupling between nearest neighbours: The nearest neighbours in the lattice need to be considered for the action calculation for each point.
2. First make a naive implementation: Calculating the the action before and after proposed change in x, then taking the difference. Then improve this by implementing just a function to calculate the difference in action in the relevant part of the lattice.
3. Switch to 1 dimensional parameter(ω or ω^2).
4. Work on Draft Report

0.9 Full Quantum Harmonic Action, Plotting of Theoretical Guassian, Plotting of Autocorrelation, Correlation and their logs, Comparison to theoretical correlation function slope, Draft Report

1. Worked on Draft Report
2. Implemented full lattice action with influence of nearest neighbours. Naive implementation.
3. Plotted theoretical Gaussian distribution of x and compared the histogram. Found close agreement.
4. Plotted the autocorrelation function, correlation function and their logs. Compared correlation function log slope to the theoretical slope($-\omega$)

0.10 Meeting 2/11/20

1. Draft Feedback:
   - More detail required.
   - Need to run longer simulation
   - Use a longer lattice length
   - Show some working/justify theoretical predictions
2. Determine error bars for observables.

3. Simplify the difference in action before and after proposed change. Make it more efficient.
4. Calculate Integrated autocorrelation to get a good measure of errors.
5. Discuss why Metropolis is used in the report
6. Discuss why Markov Chain converges.
7. Try running cold and hot starts.

0.11 Efficiency of Action Calculation, Errors

1. Made the calculation of the difference in action more efficient by only considering the nearest neighbourhood for the lattice points.
2. Added error bars for the auto-correlation and the correlation.
3. Calculated integrated auto-correlation and appropriately scaled error to account for the noise using this.

0.12 Meeting 10/11/20

1. The harmonic oscillator giving reasonable results
2. Implement the an-harmonic case

0.13 An-harmonic case and Testing the Harmonic case

1. Testing harmonic case for larger lattice for longer time
2. Implemented An-harmonic case using quartic perturbation to harmonic case.

0.14 Meeting 16/11/20

1. Discussed an-harmonic case behaviour.
2. Discussed parameters for an-harmonic case
3. Issue with an-harmonic parameters. Need to explore in the following week
4. Visualised the potential for both cases.
5. Made a separate potential method and used this in the action instead of explicitly calculating in action. Makes the action more general as a sum of kinetic and potential energy

0.15 An-harmonic Oscillator

1. Tried various sets of parameters for an-harmonic case.
2. Minimised an-harmonic potential to determine theoretical minima and compared with the plots.
3. Some work on report.

0.16 Meeting 23/11/2020

1. An-harmonic showing reasonable results.
2. Important points for report: Labelling/Captioning of graphs, Significant figures on numbers reported, put more equations in methods section, show last certain number of steps to more easily see the path(harmonic), make an-harmonic histogram more symmetric by altering the scale of potential/running longer simulation