THE UNIVERSITY *of* EDINBURGH
School of Physics
and Astronomy

Masters Project:Fourier Accelerated Dynamics in HMC Simulations of Lattice Field Theory

Aneirin John Baker

supervised by
Brain Pendleton and Roger Horsley

Abstract

I present results for a Fourier Accelerated version of Hybrid Monte Carlo specifically looking for an increase in speed towards observables. I also present a new algorithm designed specifically to deal with quartic terms which provide problem cases in many simulations.

January 4, 2021

Contents

1 Personal Statement

I spent the first 2 weeks of the project reading the material surrounding my project - mainly [1] and [2].I also began to plan out how I would implement the algorithms in C++, in doing this I gained an understanding of what the main goals of the first half of my project would be and how they could be achieved. I identified which Monte Carlo observables would be useful to measure in these simulations.

For the next eight weeks I implemented the Hybrid Monte Carlo algorithm and debugged my code whilst developing analysis tools in python. I compared the results from my simulations to the results from [3] (for the harmonic oscillator) and [4]for the anharmonic oscillator. Having obtained positive results for the harmonic oscillator I started reading up on Fourier Acceleration. I examined how to integrate a fast Fourier Transform into the simulation in order to produce the most efficient simulation - the solution I decided on was to use a package called FFTW[5].

Following this I began to integrate the Fourier Acceleration into my code and test it against the regular HMC algorithm. In addition to this I ran longer HMC simulations to verify my findings from the previous semester.

At the beginning of the second semester I finished implementing Fourier acceleration into my code and began looking into how to quantify if there was any improvement in speed gained by this algorithm. As the semester progressed I started looking into how to integrate the Anharmonic Oscillator into my code - this was the most complicated part of the project, as discussed in the body of this report.

For the final parts of semester two I worked mainly on perfecting the algorithm for the harmonic oscillator and trying to implement the anharmonic oscillators algorithm. I spent the final two weeks of semester two working on the write up for this project.

2 Introduction

Lattice Field Theory(LFT) has been a successful tool in simulating Quantum Chromo Dynamics(QCD) since its inception in 1974. Recently there has been quite a bit of interest [6][7] in improving the main algorithm within LFT, this algorithm is Hybrid Monte Carlo (HMC). Any speed up to this algorithm would be extremely useful to the lattice community as LFT simulations often take months to complete, hence a speed up of even a small factor could have a large impact. This project aims to explore the possibility of improving the HMC algorithm with a method called Fourier Acceleration.

I will first establish a base line by replicating the results found in [3] for a harmonic and anharmonic oscillator. Then I will describe how i obtained my

new algorithms for Fourier Accelerted HMC in both the harmonic and anharmonic cases and present my results for the harmonic case. Unfortunately there was not enough time to complete the simulation of the anharmonic algorithm although with a little more work this could be achieved perhaps in a summer project. My simulations will only be toy models as they will only be one dimensional simulations but this is only a test case to prove that the They will be written in C++ and Python all of the code used will be my original work, apart from a Fast Fourier Transform package (FFTW [5]) which will be used for the fourier acceleration as it is the most efficient way of computing a discrete fourier transform

3 Monte Carlo Methods

A lot of integrals within physics are too complicated to evaluate analytically to calculate them we must turn to numerical/computational methods to find an answer. When the dimensionality of these integrals is low enough these can be evaluated using techniques such as Tranpzoid rule or Simpsons rule.However techniques often break down when then dimension gets large - usually around $d = 4$ - when this occurs the only technique which has been proven useful is the Monte Carlo Method.

The Monte Carlo Method is a way to estimate integrals based on generating random numbers or states of a system. In a generic Monte Carlo simulation a random system is generated using a pseudo random number generator usually drawn from a specific distribution. An energy function is then calculated using the random system which was just generated. This energy function is a measure of how close the system is to equilibrium in the sense that it is minimised when the system is in equilibrium.

A basic Monte Carlo Algorithm would then accept all of these states and compare the final distribution of states to some criteria to determine the value of the integral in question - in most cases the average value of the integral.

The method can be improved upon by adding in an accept/reject step where the states are accepted or rejected based on the probability

$$P = min[1, exp(-\Delta H)] \tag{1}$$

Where ΔH is the change in the Hamiltonian of the system. If the new state is accepted the old state is replaced with the new state and the old one is discarded, if the state is rejected then the old state of the system is kept and a new random state is generated.

We can use the configurations generated by this update procedure to estimate integrals which represent physical quantities.Using the fact that a

d dimensional integral may be calculated using the statistical average

$$I = \int_V d^d \boldsymbol{r} f(\boldsymbol{r}) = \frac{\int_V d^d \boldsymbol{r} f(\boldsymbol{r})}{\boldsymbol{V}} \boldsymbol{V} = \bar{\boldsymbol{f}_V} \boldsymbol{V} \tag{2}$$

Where $\bar{\boldsymbol{f}_V}$ is the average of the function over some volume $\boldsymbol{V}$

For large dimensional integrals Monte Carlo methods are much more efficient than standard discretization methods. The Monte Carlo estimates can be evaluated by using the statistical average

$$I \approx \frac{\sum_{i=0}^{n} f(\boldsymbol{r_i})}{n} \boldsymbol{V} \tag{3}$$

As n approaches infinity this approximation will converge to the true value of I.

These Monte Carlo methods can be used to simulate a variety of dynamical systems ranging from simulations of molecular dynamics to the spread of diseases.

There is an issue with Monte Carlo methods concerning the generation of random states. If we use a uniform distribution the danger of generating states, which are physically not very realistic, is much higher than we would want. Although these states would be immediately rejected by the update procedure it would still slow down the simulation significantly. To get around this problem several methods are employed the first being Importance which involves generating states through a biased method. In this way we ensure the states which are physically more realistic are generated with a higher probability. In the case of statistical mechanics or molecular dynamics the usual practice is to pull the random numbers from the Boltzmann distribution. In this way the equilibrium state is much more likely to be produced and so the simulation will converge to the value of $< A >$ much quicker, where A is some observable given by

$$< A >= \frac{1}{N} \sum_{i=0}^{N} A_i \tag{4}$$

as N tends to infinity this Monte Carlo average tends to the true value. This is called Importance Sampling, it is a start to improving the basic Monte Carlo algorithm so it can be applied to larger and more complex systems.However the editing the probability in this way that means we need to compensate for this further on in the calculation so as to not change the probability of going from one state to another. The second method which we use within the algorithm is the Metropolis Hastings algorithm, this is a type of Markov Chain algorithm which allows the system to evolve towards equilibrium quicker.

3.1 Markov Chain Monte Carlo

3.1.1 Markov Chains

A Markov chain is a dynamical rule which completely determines the future of a system. If a system is in a state μ then in a finite amount of steps a Markov chain can evolve a system into a state ν. We choose a distribution to sample from to be the equilibrium distribution which we wish to simulate. This way the Markov Chain then evolves towards this state quicker than a regular Monte Carlo algorithm.

4 Hamiltonian Monte Carlo

Markov Chain Monte Carlo simulations produced results which were accurate when compared to theory [3]. However these methods were not suited for simulations of Lattice Field Theory due to the Grassman nature of fermions, and hence the sign problem [8]. To combat this new versions of Monte Carlo were employed to help with the calculations. These were formalised in the 1987 paper, Hybrid Monte Carlo(HMC) [1]. This new algorithm modified the update step of the Monte Carlo algorithm so as to include a more directed approach to exploring phase space - where as the standard Monte Carlo algorithm uses a more random walk to explore phase space. This new approach uses Hamiltonian dynamics in the update equations so that the non-physical regions of phase space are less likely to be explored and thus the simulations will converge to equilibrium much quicker.

4.1 Simulation of a Path Integral

As mentioned previously this project aims to investigate ways of speeding up algorithms within Simulations of Lattice QCD. In this vein the Path Integral approach to Quantum Mechanics was used. To numerically simulate a path integral first space and time were discretised, to simulate a continuous path integral we place oscillators - or more generally the system we wish to examine - at every point on a lattice. We then associate each point on the lattice with a point in time, the separation between each of these points is called the Lattice spacing (in our equations a).These oscillators will then be evolved using the HMC algorithm forward in Monte Carlo/Computer time until they reach equilibrium. With all of this in mind we have now a lattice of oscillators from which we can collect simulation data and calculate various Monte Carlo Observables which, as $a \to 0$, will converge to the correct continuum limit.This type of Monte Carlo simulation of a quantum system was first done by Creutz and Freedman [3] in 1981 however not using HMC.

4.2 Hamiltonian Dynamics

The main body of Hybrid Monte Carlo (or Hamiltonian Monte Carlo) simulations are based in Hamiltonian Dynamics. A quick overview of the subject is provided and then a description of how these equations can be changed into update equations through discretisation which can then be implemented into an algorithm. We will then explore further why the discretised algorithm can still be used even though space and time are not modelled as continuous.

The central equations of Hamiltonian Dynamics are Hamilton's equations

$$\begin{aligned} \frac{\partial q_i}{\partial t} &= -\frac{\partial H}{\partial p_i} \\ \frac{\partial p_i}{\partial t} &= \frac{\partial H}{\partial q_i} \end{aligned} \tag{5}$$

for $i = 1, ..., d$ where d is the dimension of the system, p is the conjugate momentum and q is the position of the system. These equations can be integrated to obtain the equations of motion of the system defined by the function H the Hamiltonian, defined in terms of the Lagrangian of the system L through the Lagendre transform, however for our systems we can think of it as the sum of the Kinetic and Potential energy of the system.

$$H(q, p) = \dot{q}\frac{\partial L}{\partial \dot{q}} - L(q, \dot{q}) \tag{6}$$

We now examine some properties of Hamiltonian dynamics which are central in recasting them in terms of a Markov Chain Monte Carlo update. First we note that Hamiltonian dynamics are classical and hence deterministic, they should then be reversible, that is if we integrated forward to time T we should be able to reverse that integration by setting $T \to -T$ and then integrating backwards along the trajectory and return to the initial state.We expect that the discretised update equations we are looking for should also have this property The next property of Hamiltonian dynamics is that along these classical trajectories the Hamiltonian (or energy in classical systems) is conserved i.e.

$$\frac{dH(q, p)}{dt} = 0 \tag{7}$$

This can be proved using (5).

$$\frac{dH(p, q)}{dt} = \sum_{i=1}^{d} \big[\frac{dq_i}{dt}\frac{\partial H}{\partial q_i} + \frac{dp_i}{dt}\frac{\partial H}{\partial p_i}\big] = \sum_{i=1}^{d} \big[\frac{\partial H}{\partial p_i}\frac{\partial H}{\partial q_i} + \frac{\partial H}{\partial q_i}\frac{\partial H}{\partial p_i}\big] = 0 \tag{8}$$

The conservation of the Hamiltonian implies that the acceptance probability within the Monte Carlo simulation would be one (note that this is all in the

continuum limit). However if the Hamiltonian is only partially conserved the acceptance probability will never be 100%, but it can be made close to 100% by taking the discretisation parameters to 0. As we know there is a conserved quantity if the Hamiltonian is explicitly independent of t - as will be the case - then we can use Noethers theorem to find the conserved quantity. After the application of Noethers theorem we find that the energy of the system will be conserved - or only partially in our case as we will find in the next section. This gives us a effective check as to whether or not the system is working.

4.3 Discretisation of the Hamiltonian Equations

To create a set of equations which can be used in a computer simulation, Hamilton's equations need to be discretised. To do this we approximate a derivative with a finite difference (in this case a forward difference) so that we can put the equations into the necessary form.

A forward difference can be thought of as just the definition of differentiation.

$$\lim_{h\to 0} \frac{f(x+h)-f(x)}{h} = f'(x) \tag{9}$$

For finite differences we simply don't take the limit as h tends to 0 and instead keep it at some finite number. After redefining $f(x) \to q_i$ and $f(x+h) \to q_{i+1}$ and replacing the small difference h with our discretisation parameter a, we can make the following deductions.

$$\frac{\partial H}{\partial q} = -\dot{p} \to -\frac{p_{i+1}-p_i}{a} \tag{10}$$

$$\frac{\partial H}{\partial p} = \dot{q} \to \frac{q_{i+1}-q_i}{a} \tag{11}$$

These equations can now be applied to a Hamiltonian of our choosing to produce the update equations needed for the simulation.

4.4 Conservation of the Hamiltonian

As we would like the simulation to be as physically realistic as possible. One condition for this is that we need to ensure that the Energy of the system remains constant through out the simulation (in this case the energy of the system is numerically equivalent to the value of the Hamiltonian).Hence we require that the Hamiltonian should be conserved even after discretisation. To prove that the Hamiltonian will still be conserved with this discretisation we consider a transition matrix which would take the the original coordinates (p, q) to new coordinates $(p*, q*)$ with a mapping T_s. To prove that this

mapping preserves the volume of this system in phase space we examine the Jacobian of this mapping. This mapping can be written as

$$T_\delta(p,q) = \begin{bmatrix} p \\ q \end{bmatrix} + \delta \begin{bmatrix} \frac{\partial p}{\partial t} \\ \frac{\partial q}{\partial t} \end{bmatrix} + termsoforder\delta^2 \tag{12}$$

where δ is a small parameter close to 0

With this mapping we can construct a Jacobian which describes how the volume of phase space will change. Specifically the determinant of the Jacobian will determine the factor by which the volume changes. If the determinant is 1 then the volume in phase space will not change. From the mapping above we can construct the Jacobian

$$B_\delta = \begin{bmatrix} 1 + \delta \frac{\partial^2 H}{\partial p \partial q} & \delta \frac{\partial^2 H}{\partial p^2} \\ \delta \frac{\partial^2 H}{\partial q^2} & 1 - \delta \frac{\partial^2 H}{\partial q \partial p} \end{bmatrix} + termsoforder\delta^2 \tag{13}$$

The determinant of this is

$$det(B_\delta) = 1 + \delta \frac{\partial^2 H}{\partial p \partial q} - \delta \frac{\partial^2 H}{\partial q \partial p} + termsoforder\delta^2 \tag{14}$$

where the terms of order δ cancel out so that the determinant is 1. This means that the phase space volume is preserved under Hamilton's equations and in this case the discretised Hamilton's equations. This implies that the Hamiltonian is conserved, being conserved means that in our update algorithm we do not need to take account of any volume differences in our probability distribution. If there was some difference in volume the Jacobian of the transformations would need to be taken into account in the Metropolis Update which in some cases may not be computable.

4.5 Update Method

Now we have the Hamiltonian equations of motion in differential equation form we can apply standard methods in order to produce results for any Hamiltonian. In general a Hamiltonian will not be integrable and so we will need to approximate a solution, that is we will need to discretise the equations and produce further equations, known as update equations, which will approximate the trajectory of particles as they move forward in time. This can be achieved by a number of methods some better than others, here we will use a method known as Leapfrog Integration for the reasons given below.

4.5.1 Leapfrog Method

The Leapfrog method is a method of integrating equations in a reversible manner. It uses the momentum (or velocity) half way between the position

updates so as to obtain a better approximation [9] to the actual velocity of the oscillator at that point in time. The general update equations are given below

$$\begin{aligned} p_i(t+\frac{\epsilon}{2}) &= p_i(t) - \frac{\epsilon}{2}\frac{\partial U}{\partial q_i}(q_i(t) \\ q_i(t+\epsilon) &= q_i(t) + \epsilon\frac{p_i(t+\frac{\epsilon}{2})}{m} \\ p_i(t+\epsilon) &= p_i(t+\frac{\epsilon}{2}) - \frac{\epsilon}{2}\frac{\partial U}{\partial q_i}(q(t+\epsilon)) \end{aligned} \tag{15}$$

where U is the potential associated with the system which these equations are being used to simulate (which comes from the Hamiltonian being split up into $H = K + U$ where K is the kinetic energy of the system).ϵ is the time step which each leapfrog step takes, in a sense it is the measure of how close to an analytic integration the method is. These steps will be repeated n times so that the algorithm will be evolved over a time $T = n\epsilon$.

This method is more effective than other simple integration methods such as the Euler method, for a more in depth discussion of Leapfrog and other update methods see [9]

4.5.2 Detailed Balance

The Metropolis Update described in the Monte Carlo section obeys a condition called Detailed Balance.Simply put this leaves the distributions from which the states are being sampled invariant after the Metropolis Update steps. This characteristic is needed in the more complex algorithms as we too need the distributions to remain invariant. We now prove that the Markov Chain Monte Carlo which is being proposed here also obeys Detailed Balance.

We can partition phase space into different sections where A_k is the initial state and B_k is the image of the initial state after N Leapfrog steps. Due to the fact that the Leapfrog Integration scheme is reversible both A and B will partition phase space. We now note that Detailed Balance will hold if the following condition holds, which is just a re statement of the usual detailed balance formula from Statistical Mechanics

$$P(A_i)T(B_j|A_i) = P(A_j)T(B_i|A_j) \tag{16}$$

where $P(A_i)$ is the Canonical Distribution which is being used to generate the states from A and $T(A_i|B_j)$ is the conditional probability of going from state A to B. It is easy to see that when $i \neq j$ there is no probability of going from state A_i to state B_j that is $T(A_i|B_j) = 0$ (ie when the partitions are not in the same set then the transition will be impossible from one to another).The transition probability from A to B is defined to be

$$exp(-H_{A_k})min[1, exp(-H_{B_k}+H_{A_k})] = exp(-H_{B_k})min[1, exp(-H_{A_k}+H_{B_k})] \tag{17}$$

which can easily be seen to be true for the cases where $H_{B_k} > H_{A_k}$ and vice versa. This proves that Detailed Balance is obeyed for the Metropolis Update which is being used in the HMC algorithm. To now see that this Detailed Balance proves that the canonical distribution is left invariant it will need to be shown that the probability will remain invariant under a Leapfrog update step. Letting $R(X)$ be the probability of a rejection we can write the probability of a state being in a region of phase space as the probability of the update step being reject plus the probability that another state has moved into that region of phase space. This can be written as

$$\begin{aligned} & P(B_k)R(B_k) + \sum_i P(A_i)T(B_k|A_i) \\ &= P(B_k)R(B_k) + \sum_i P(B_k)T(A_i|B_k) \\ &= P(B_k)R(B_k) + P(B_k)\sum_i T(A_i|B_k) \\ &= P(B_k)R(B_k) + P(B_k)(1 - R(B_k) = P(B_k) \end{aligned} \tag{18}$$

which proves that the probability distributions are left invariant after the HMC algorithm.

4.6 HMC Algorithm

Using the update methods described above we can simulate a Markov Chain which can more efficiently evolve the system towards equilibrium, which is better than a totally random Monte Carlo simulation.

There are two main steps within the hybrid (Hamiltonian) Monte Carlo algorithm. First the new samples of momentum are drawn from a Gaussian distribution with mean of 0 and a variance of 1. These new momentum variables are then used along with the previous position variables to update the position and momentum for n Leapfrog steps with a step size of ϵ. This then provides us with a set of variables (p^*, q^*) which are the new variables that have been evolved along a trajectory in phase space of length $n\epsilon$. Where n is the number of Leapfrog steps which have been taken.

A Metropolis update is then performed where the difference in the Hamiltonian between the beginning and end of the leapfrog steps is used as the argument for the exponential

$$AcceptanceProbability = min[1, exp(-H(p^*, q^*) + H(p, q)] \tag{19}$$

If the proposed state is not accepted the old variables (p, q) are used in the next loop through the algorithm. In the first iteration of the HMC algorithm one of two possibilities will occur: either the simulation starts off in what is known as a cold state where all of the position variables (in this case) are set to 0, or they are drawn from a uniform random distribution and the simulation proceeds from there. As this process should always evolve the system through phase space to its equilibrium position -as we have specified the equilibrium distribution by hand - then the starting point should not make any difference. However it is often easier computationally to vary the starting parameters according to the type of simulation running. In this project it was found that the system would converge much quicker if we took a random distribution ranging from -1 to 1.

Algorithm 1 Hybrid Monte Carlo Algorithm

1: Pick Random momenta From Gaussian Distribution
2: **for** `N iterations` **do**:
3: $p_{i+1} = p_i - \frac{1}{2}\Delta t \frac{\partial H}{\partial q_i}$
4: **for** `n iterations` **do**:
5: $q_{i+1} = q_i + \Delta \frac{\partial H}{\partial p_i}$
6: **if** j $\neq$ n-1 **then**:
7: $p_{i+1} = p_i - \Delta t \frac{\partial H}{\partial q_i}$
8: $p_{i+1} = p_i - \frac{1}{2}\Delta t \frac{\partial H}{\partial q_i}$
9: Perform Metropolis Update on the new state (p^*, q^*)

A more in depth explanation of the algorithm can be found in any of the following [10][2][11]

Over the next sections findings for the harmonic and anharmonic oscillators will be presented, comparing them to known discretised results and the theoretical results in the continuum limit.

4.7 Harmonic Oscillator

To Simulate a Quantum Harmonic Oscillator we start with continuum action

$$S_E[q(t)] = \int dt \left(\frac{1}{2} \partial_t q(t) \partial^t q(t) + V(q(t)) \right) \tag{20}$$

We then discretise it and transform to a Hamiltonian formulation to obtain the equations

$$\begin{aligned} H(p,q) &= \frac{p_i^2}{2m} + S \\ S(p,q) &= a \sum_{i=1}^{N} \left(\frac{1}{2} \frac{(q_{i+1} - q_i)^2}{a^2} + V(q_i) \right) \end{aligned} \tag{21}$$

Where a is the lattice spacing and the potential $V(q_i)$ determines whether the simulation is a harmonic or anharmonic oscillator in this case the potential is

$$V(q_i) = \frac{1}{2} \mu^2 q_i^2 \tag{22}$$

Note that the first term in (20) corresponds to the coupling term in our discrete action. Hamiltons equations are now applied to the Hamiltonian above and then the equations are discretised accordingly so that usable update equations are produced. The resulting equations are

$$\begin{aligned} q_{i+1} &= q_i + \frac{\Delta t}{m} p_i \\ p_{i+1} &= p_i - \Delta t \left(m q_i a + \frac{m}{a} (2q_i - q_{i-1} - q_{i+1}) \right) \end{aligned} \tag{23}$$

These equations were then applied in the HMC algorithm with a large array of oscillators. N, the number of oscillator, was chosen to be 1000 so as to avoid any finite size effects.

Periodic Boundary conditions were implemented and hence the simulation was really a simulation of oscillators on a circle. The Lattice spacing was also chosen so as to approximate the continuum limit. For the initial runs a was chosen to be 1 for simplicity, however when the simulation was found to agree with theoretical calculation a was reduced to approximate and test the continuum limit.

4.7.1 Results

In this section the results of the simulations which have been run. All the parameters which were used in the simulations will be noted below the result tables or graphs which are presented.However through out the simulations of the harmonic oscillator the mass was set to 1.0 in accordance with the results we were comparing to. To obtain these results the simulation went through what is known as a burn period this is where it is not in equilibrium but by the end of the period it should be equilibrium. This can be seen from example graph below

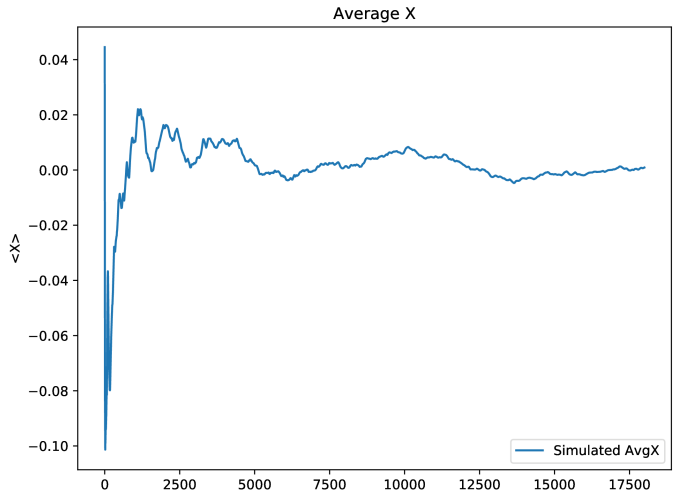


Figure 1: This graph of $< x >$ vs Monte Carlo iterations shows that there is large statistical fluctuations up until 7500

When the burn period for this simulation is included we find that the results become more accurate and the error bars on the results decrease. Hence all these simulations include a burn period of 5000 iterations out of a total of 50,000 Monte Carlo iterations. Throwing away 10% of the data is more than enough here however, to ensure that the system is in equilibrium.

With this burn period in our simulation we now turn to the results of this basic Hybrid Monte Carlo Algorithm. We start with the Group State Wave Function for a harmonic oscillator. Since we are on a discrete lattice there are some corrections which occur [3] which need to be taken into account. These corrections amount to a change in the oscillation frequency of

$$\omega^2 = \mu^2(1 + \frac{a^2\mu^2}{4}) \tag{24}$$

This leads to a wavefunction which has the form

$$|\Psi(x)|^2 = (\frac{\omega}{\pi})^2 exp(-\omega x^2) \tag{25}$$

as we take $a \to 0$ we see that this equation reduces to the usual Wavefunction for a harmonic oscillator.

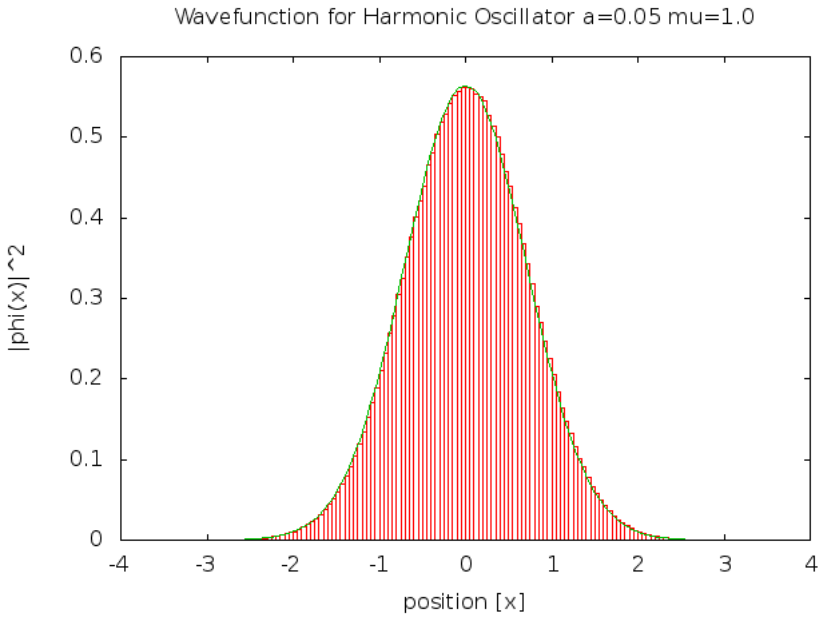


Figure 2: Wave function of the Harmonic Oscillator with discrete theory wave function plotted in Blue. For parameters $\mu = 1.0$ and a = 0.05

For this graph the parameters a=1 and μ= 1. To get this wave function the positions of an oscillator along the lattice were binned into a Histogram for each iteration. In this way the sample which is obtained is much more representative of the simulation than just one iteration. The blue line plotted is the discrete wave function described in (25) and the dotted line is the continuum theorem.

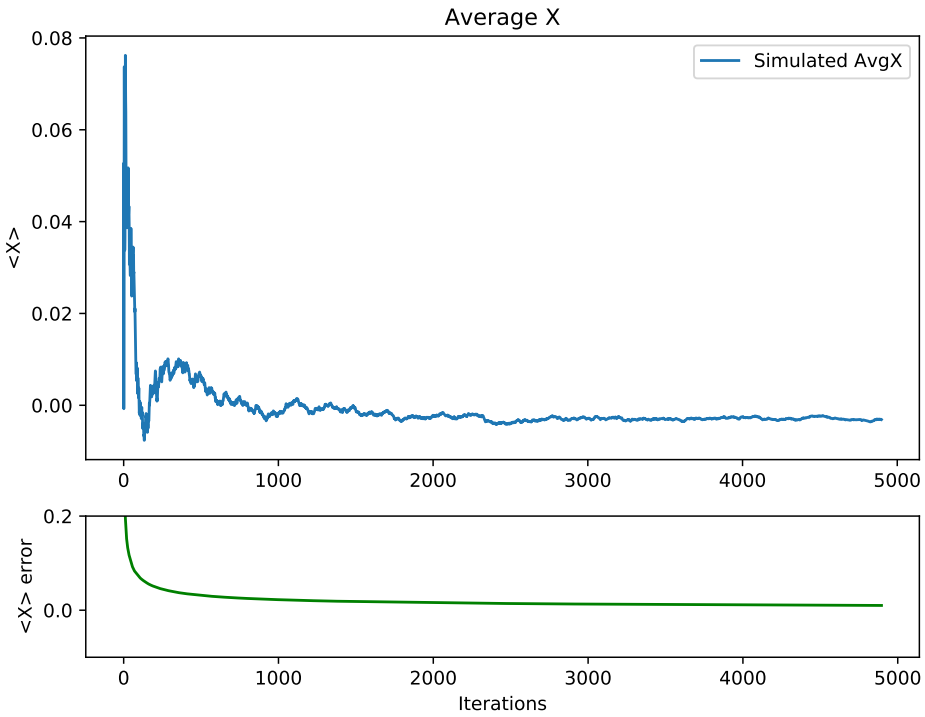
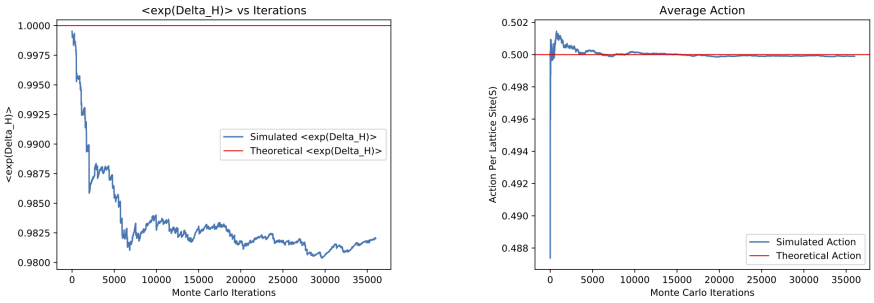


Figure 3: Graph of average x over the simulation for a=1.0 and mu = 1.0

Figure 3 shows how the average x varies over the simulation. There are several things to take note of in this graph. First and most obvious is that the graph does not deviate much from the theoretical line (in red) after going into equilibrium meaning that the simulation remains in equilibrium after the initial equilibration. This gives some evidence that the simulation is physical as it does not deviate from equilibrium. The second point is that the simulation tends to the correct average x value of 0. As the harmonic potential is symmetric about 0 the integral

$$< x >= \int dx x e^{-x^2} = 0 \tag{26}$$

evaluated to 0 and hence the average x should go to 0.



(a) Plot of Average exp(-delta H) (b) plot of Average Action

Figure 4: Plots of Action and exp(-delta H) using a=μ=1.0

The average action shows much the same as figure 3. The more interesting graph here is the average ΔH. This graph shows the difference between the Hamiltonian at the beginning of the Leapfrog algorithm and at the end. This difference quantifies how close to equilibrium the system is. We can see from the scale of this graph the simulation is very close to equilibrium and that it stays in equilibrium when the graph flattens out.

The main way of knwoing whether or not this simulation was working was to look at the ground state energy. To calculate this we use the Virial Theorem for a general quartic or anharmonic oscillator.

$$E_0 = \mu^2 < x^2 > -3\lambda < x^4 > \tag{27}$$

Lattice Spacing	E_0 Simulated	E_0 Discrete Theory	Ratio Between simulated and Discrete Theory
1	0.447284	0.447214	1.00015652
0.5	0.485350	0.485071	1.00057517
0.1	0.498354	0.499376	0.99795345
0.05	0.499583	0.499844	0.9994774

Table 1: Table for the Ground state energy calculated from the virial theorem

values here taken from equations found in [3] namely

$$< x^2 >= \frac{1}{2\mu(1+\frac{a^2\mu^2}{4})^{\frac{1}{2}}}\left(\frac{1+R^N}{1-R^N}\right) \tag{28}$$

Where R^N takes into account the finite lattice size with R given by

$$R = 1 + \frac{a^2\mu^2}{2} - a\mu\left(1+\frac{a^2\mu^2}{4}\right) \tag{29}$$

For large enough N the ratio at the end of this equation is approximately 1 and so can be ignored (e.g. when N=100 a=1.0 and $\mu = 1.0$ R^N =6.22 $\times$ 10^{-61} so essentially 0) . We can now input values for a and μ into the equation to get values for E_0 in the discrete theory.The continuum theory of this equation is

$$< x^2 >= \int dx x^2 e^{-x^2} = \frac{1}{2} \tag{30}$$

Related to (28) above the autocorrelation function can be used to calculate the excited states of the system. Using the same method as Creutz et al.[3] used to evaluate the autocorrelation function and obtain the first excited state, we plot the correlation function and then fit an exponential to the graph which gives us the difference between the ground and first excited state. An alternate (and equivalent method) is to use the equation

$$E_1 = \frac{-1}{\Delta\tau} ln \left[\frac{< x(0)x(\tau + \Delta\tau >)}{< x(0)x(\tau) >} \right] + E_0 \tag{31}$$

where τ is the time index along the lattice which the oscillators lie on. To generate the theoretical values we can use a generalised version of (28)

$$< x_i x_{i+j} >= \frac{1}{2\omega(1 - R^N)} (R^i + R^{N-j}) \tag{32}$$

this equation was used to generate the theoretical values in the graph below.

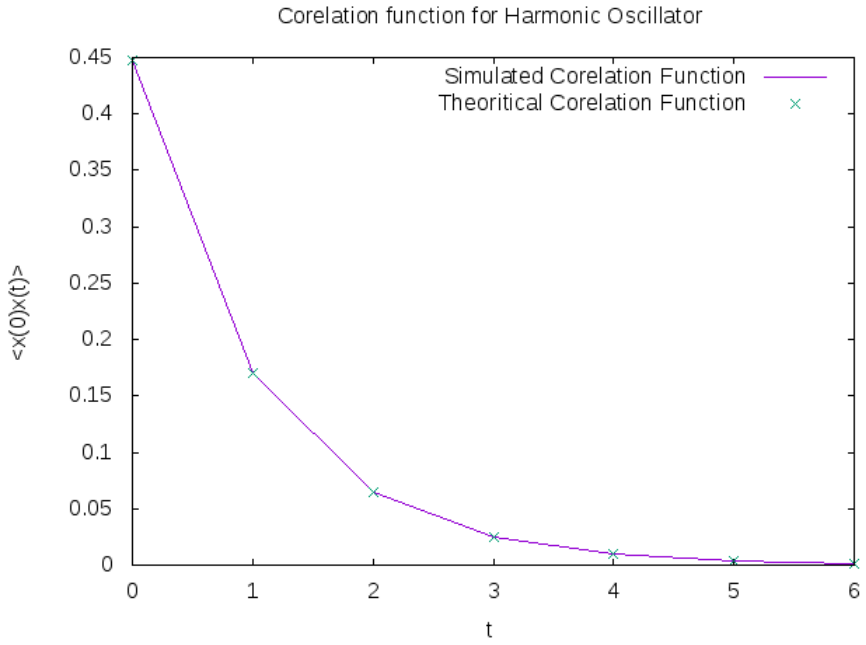


Figure 5: Autocorrelation for the harmonic oscillator for $a = \mu = 1.0$.With Theoretical and simulated values plotted, this graph is used to determine the first excited state.

From this graph which shows the corelation time along the lattice we calculate the difference between the ground and first excited state to be $E_1 - E_0 = 0.95486$ which is in good agreement with the theoretical value of 0.96242.

All this evidence shows that the HMC algorithm works in the case of the harmonic oscillator. This will now be used as a base case to show where the HMC algorithm falls down and thus an improvement in speed is needed.

4.8 Anharmonic Oscillator

The Anharmonic oscillator differs from the harmonic oscillator by a quartic term. This term generates a double welled potential. It is this well that causes the difficulty in simulating an anharmonic oscillator. In this case we parameterised the potential in terms of a new parameter f which made it easier to identify the locations of the wells.

$$V(q) = \lambda(q^2 - f^2)^2 \tag{33}$$

Here λ is the parameter which controls the depth of the potential wells and f describes the location along the x axis.There is also a constant term

λf^4 included in this potential, however it is easy to correct for this in our equations as it is only a constant and will not make any difference to the physics - as we can just add a factor of λf^4 to the ground state energy to correct for this. When plotted this equation is

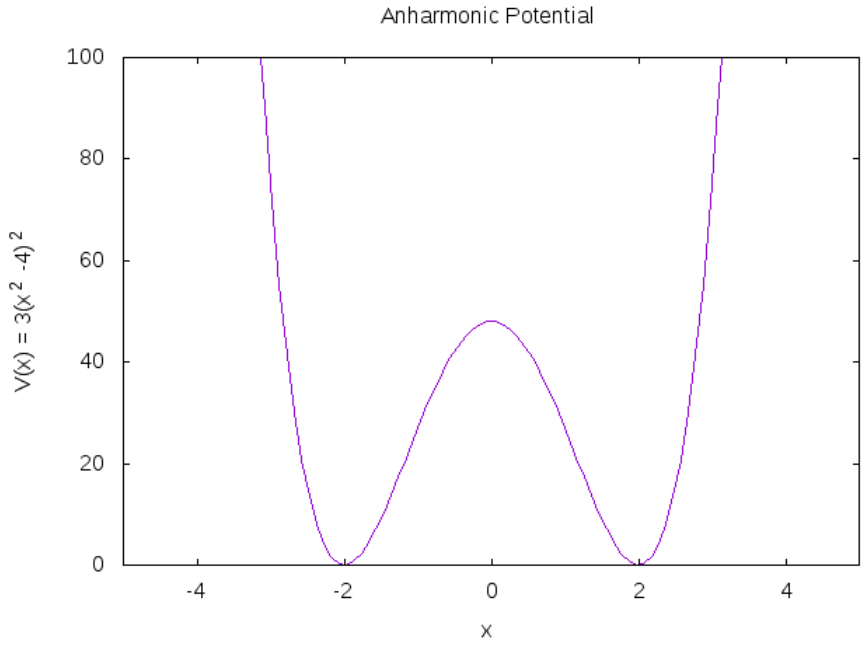


Figure 6: Anharmonic Potential for $\lambda = 3$ and $f^2 = 4$ with $m = 0.5$ to make the potential agree with results from[4]

The double well as mentioned earlier is the problem with this system. The oscillators require a larger momentum to get over (tunnel through) the potential barrier and into the other side. By the very nature of large numbers they are not very common in a Gaussian distribution (the distribution which we are sampling from in this simulation) and so the system does not often tunnel from one side of the potential well to the other. This produces results which are not representative of the whole of phase space, we can see this from the plots of the Wave function such as this one, note that for these simulations for the anharmonic oscillator the mass was set to 0.5 to bring our results in line with [4]

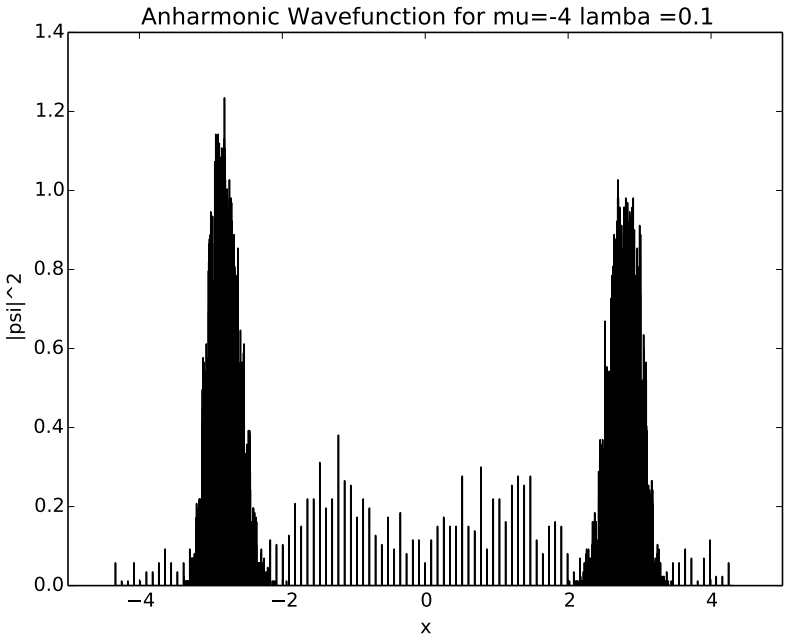


Figure 7: Wavefunction for an anharmonic oscillator with $\lambda = 1$ and f^2=9, noting the asymetrical peaks.

Here we notice that the peaks are of different heights in the wavefunction. This gives evidence that the simulation is staying in one side of the potential well more than the other. This is not what is desired in this simulation and will give results which are not valid. This graph also shows some of the tunneling. The small peaks in the middle of the graph are the simulation tunneling from one well to another. This is the quantity that we wish to accelerate with the new algorithm proposed.

This can also be seen in the calculations for the ground state energy. Using the values found in [4] for the harmonic oscillator (making sure that the Hamiltonian we use is the same as the one used in the above paper)we can compare our simulated values to theoretical values for an anharmonic oscillator. In doing this we find that the anharmonic oscillator is not as stable as the harmonic oscillator. In the sense that it takes a long time to evolve to equilibrium and the time step needs to be taken towards 0 for the acceptance rate to be non zero. This can also be seen in the acceptance rates of the states within the simulation. At the same lattice spacings and time step, as for a Harmonic oscillator,the acceptance rate is very small and so the time step has to be reduced in order that the equations of motion are not integrated as far and thus have less chance to deviate significantly from

the continuum trajectory.

f	E_0 Simulated	E_0 Continuum Theory	Ratio
-1	2.595	2.677	0.969
0	1.035	1.060	0.976
1	1.150	1.137	1.011
2	1.93	2.289	0.843

Table 2: Table of values for the ground state energy of an anharmonic oscillator for a lattice spacing of 0.1

In comparison with the same values or ratios for the harmonic oscillator these values are much less accurate. We note that there is no analytic solution for the anharmonic oscillator In comparison with the same values for the harmonic oscillator these values are much less accurate. This could be due to the comparison being between the simulated discrete values and the continuum values, as we saw that there can be a significant difference between the discrete and continuum values. However the continuum values should be reached when a is taken to 0 as shown in the table below.

Lattice Spacing a	Ground State Energy E_0	$< \Delta H >$
1.0	0.5649	-0.08
0.8	0.6469	-0.04
0.5	0.8043	0.02
0.2	0.9569	-0.6
0.1	1.03534	0.2
0.05	1.2468	2.44

Table 3: Table showing how the ground state energy for $f^2 = 0$ varies with the lattice spacing. The average ΔH has been added to show that when we take a too small the simulation breaks down, it should stay close to 0.

5 Fourier Acceleration

Now we have seen that the HMC algorithms works for a basic harmonic oscillator and an anharmonic oscillator we can now look for some way of improving this algorithm. As noted in section 4.8 above, the anharmonic oscillator is nowhere near as accurate as the harmonic oscillator. This is due to the $\lambda(q^2 - f^2)^2$ term in the action, this potential has a double well and hence some of the states can get stuck in one side of the potential. In a real anharmonic oscillator there should be an equal probability of the oscillators being in either well. However if some of the oscillators get stuck in one of the wells this statement will not be true in our simulation. This is due to the basic fact that large numbers by their very definition are uncommon in a Gaussian distribution (which is where we are drawing our numbers from). These large numbers are needed to generate a momentum which can push the oscillators over the potential barrier and into the other well, known as tunneling. We need to find some way of improving the amount of tunneling, known as the tunneling rate, in this situation our method of choice will be to explore Fourier Acceleration.

5.1 Explanation of Fourier Acceleration

The idea of Fourier Transform is to change the momentum slightly so we emphasis the low lying modes in the simulation. This then makes the configurations where the system will tunnel more likely to appear in the simulation thus fixing the problem which we identified in the basic HMC case. Another way of thinking about Fourier acceleration is to note that now all of the oscillators are uncoupled they can evolve freely without any interference from their neighbors and hence can evolve to equilibrium quicker than when they were coupled. For more information on Fourier acceleration see [6].

This idea of Fourier Acceleration is based on boosting the low lying fourier modes in this simulation which are suppressed. Boosting these modes then allows the simulation to more efficiently tunnel between the two allowable states. We first need to define what the Fourier transform on a lattice (or discrete Fourier transform) is. For our purposes we define it to be

$$\tilde{f}(k) = a \sum_x f(x) e^{-ikx} \tag{34}$$

In a continuum Fourier transform the volume of the system is assumed to be infinite but as we have periodic boundary conditions in our system there are some conditions which need to be imposed on the variable k. Our periodic boundary conditions require the following

$$f(x + N) = f(x) \tag{35}$$

where N is the size of our lattice. We now apply this condition to our definition of the Fourier transform. Where we define $x' = x + N$

$$\tilde{f}(k) = a\sum_x f(x+N)e^{-ikx} = a\sum_x f(x')e^{-ik(x'-N)} = a\sum_x f(x')e^{-ikx'}e^{ikN} \tag{36}$$

As we require the RHS to be equal to the LHS we find that e^{ikN} must be of the form $e^{2\pi m}$ where m in some integer. Hence we arrive at a condition on k being that $k = \frac{2\pi m}{aN}$.

We now examine the continuum limit of our Action term which would be

$$S = \int dt \frac{1}{2}\partial_\mu q(t)\partial^\mu q(t) + V(q(t)) \tag{37}$$

For this manipulation the form of $v(q)$ does not matter and so we will ignore it for the time being, in later explorations of Fourier Acceleration the form will matter much more.

Now we integrate by parts so that we obtain a double derivative on one of the fields

$$S = -\int dt \frac{1}{2} q(t)\partial^2 q(t) \tag{38}$$

This operator can be approximated, much the same way we did in the previous section, using a finite difference method. This time for a second derivative which will be

$$f''(x) = \lim_{h\to 0} \frac{f(x-h) - 2f(x) + f(x+h)}{h^2} \tag{39}$$

applying this to our equation and approximating the integral with a sum we obtain the following

$$S \approx -\frac{a}{2}\sum q(x)\frac{q(t-a) - 2q(t) + q(t+a)}{a^2} \tag{40}$$

we now apply the discrete Fourier transform to this

$$S \approx -\frac{a}{2}\sum_{t,k,k'} q(\tilde{k'})e^{-ik't}\frac{q(\tilde{k})e^{-ik(t+a)} - 2q(\tilde{k})e^{-ikt} + q(\tilde{k})e^{-ik(t-a)}}{a^2} \tag{41}$$

we can now take out the factors of e^{ikx} and $e^{ikx'}$ and combine them into a delta function as

$$\delta(x-x') = \sum_k e^{ik(x-x')} \tag{42}$$

giving us the expression

$$S \approx \sum_k q(\tilde{-k})q(\tilde{k})\frac{e^{-ika} - 2 + e^{ika}}{a^2} \tag{43}$$

using a trigonometric identity and the fact that $\phi(\tilde{-k}) = \phi^*(\tilde{k})$ we see that

$$S \approx \sum_k |q(\tilde{k})|^2 \frac{2cos(ka) + 2}{a^2} \tag{44}$$

Showing that we have decoupled the oscillators on our lattice and have hence obtained the Hamiltonian ,as the Fourier transforms of the other terms in H are simple, for a harmonic oscillator in Fourier space

$$H_k = \frac{|p_k|^2}{2} + a\frac{|q_k|^2}{2}\big(\mu^2 + \frac{4}{a^2}sin^2(\frac{ka}{2})\big) \tag{45}$$

Where again we have used a trig identity to simplify the equation. This Hamiltonian can now be made into update equations which can be implemented in our HMC simulation. Further background reading can be found in the following papers [12]

5.2 Methods and Implementation

For the harmonic oscillator this transformation was made much easier by the quadratic nature of the potential in Fourier space this potential

$$V(q_i) = \frac{\mu^2 q_i^2}{2} \tag{46}$$

becomes

$$V(q_k) = \frac{\mu^2 |q_k|^2}{2} \tag{47}$$

which is easy to implement in momentum space. The HMC algorithm now has two extra steps in its implementation

Algorithm 2 Fourier Accelerated Hybrid Monte Carlo Algorithm

Pick Random momenta From Gaussian Distribution
for `N iterations` **do**:
    Transform the momentum and position into Fourier Space
    $p_{i+1} = p_i - \frac{1}{2}\Delta t \frac{\partial H}{\partial q_i}$
    for `n iterations` **do**:
        $q_{i+1} = q_i + \Delta \frac{\partial H}{\partial p_i}$
        if j $\neq$ n-1 **then**:
            $p_{i+1} = p_i - \Delta t \frac{\partial H}{\partial q_i}$
    $p_{i+1} = p_i - \frac{1}{2}\Delta t \frac{\partial H}{\partial q_i}$
    Transform to position Space
    Perform Metropolis Update on the new state (p^*, q^*)

However when we examine the anharmonic oscillator we note that the quartic nature of the potential will prove difficult to deal with in this situation. The potential will transform to

$$V(q_k) = \lambda\delta(k - k' - k'' - k''')(q_k q_{k'} q_{k''} q_{k'''} - f^2)^2 \tag{48}$$

Which is not a very useful equation to deal with on a computer. It would be possible to implement this however it is much easier to search for another solution. To do this we refer to a method which may seem to be slightly odd at first but due to the freedom in choosing the momenta in HMC we can perform this algorithm.

We now choose our Hamiltonian, in the continuum limit, to be the following

$$H = \frac{p^2}{M^2 + \partial^2} + V(q) \tag{49}$$

This when we discretise the system will amount to having the coupling in the momenta and not in the position. Since only the potential of the system describes the physical properties of the system we are free to choose whatever momenta we like provided we make the appropriate changes to our algorithm. These would be

- Change the Distribution which the momenta are drawn from in the beginning
- Change the EoM appropriately to accommodate this new Hamiltonian
- Account for the changes in degree of freedom.

The reason this Hamiltonian was chosen was that previously we transformed the Laplacian operator and found that in discrete fourier space it had the very pleasing form of

$$\partial^2 \to \frac{4}{a^2} sin^2(\frac{ka}{2}) \tag{50}$$

As we know how that operator transforms in Discrete momentum space [13] we now know how the entirety of the momentum term in our Hamiltonian changes and we can propose a new algorithm which will provide the necessary improvement in speed we are looking for.It should be noted here that as this next algorithm is going to make heavy use of the Fourier Transform as this the code uses a package called FFTW [11] (an open source FFT package). This insured that the cost of adding a Fourier tranform to the algorithm was only $O(nlog(n))$ as opposed to a slow Fourier transform which would have been of the order N^2.

The new algorithm is now

Algorithm 3 Anharmonic Fourier Accelerated Hybrid Monte Carlo Algorithm

1: Pick Random momenta From Gaussian Distribution in Fourier Space
2: **for** `N iterations` **do**:
3: $\tilde{p_k} = \frac{|p_k|^2}{1+\frac{4}{a^2}sin^2(\frac{ka}{2}}$
4: Fourier Transform back to position space
5: $q_{i+1} = q_i - \frac{1}{2}\Delta t \tilde{p_i}$
6: **for** `n iterations` **do**:
7: $p_{i+1} = p_i - \Delta \frac{\partial H}{\partial q_i}$
8: **if** j $\neq$ n-1 **then**:
9: Fourier Transform to momentum space
10: $\tilde{p_k} = \frac{|p_k|^2}{1+\frac{4}{a^2}sin^2(\frac{ka}{2}}$
11: Fourier Transform back to position space
12: $q_{i+1} = q_i - \Delta t \tilde{p_i}$
13: $\tilde{p_k} = \frac{|p_k|^2}{1+\frac{4}{a^2}sin^2(\frac{ka}{2}}$
14: Fourier Transform back to position space
15: $q_{i+1} = q_i - \frac{1}{2}\Delta t \tilde{p_i}$
16: Perform Metropolis Update on the new state (p^*, q^*)

The above algorithm is quite a bit more complicated than its Harmonic counterpart this is due to the issues raised in (48).There is also the added complication of degrees of freedom. As Fourier transformed variables are complex we have gone from having N degrees of freedom to having $2N$ Degrees of Freedom (DoF) hence we need to find some way of reducing these DoF down back to N.

To do this we have to examine the discrete Fourier transform. We note that when we take the complex conjugate of (34) we find

$$\tilde{D}(k)^* = \sum_{m=0}^{N-1} e^{-i\frac{2\pi}{N}mk} D(m) = \sum_{m=0}^{N-1} e^{i\frac{2\pi}{N}m(N-k)} D(m) \tag{51}$$

Now using $e^{2\pi im} = +1$

Hence we end up with the relation

$$\tilde{D}(k)^* = \tilde{D}(N-k) \tag{52}$$

This gives a restriction on the Fourier transformed variables, this (when applied to the system) reduces the number of variables in the system from $2N$ to N which is needed due to the complex nature of Fourier space. This restriction is applied in the algorithm for the anharmonic oscillator.

5.2.1 Harmonic Oscillator

To check that Fourier Acceleration still produces valid results we compare the results from the Fourier accelerated simulations to the previous results. Here we find that the Fourier Acceleration does not affect the results in a negative way and produces results which are as accurate as the regular HMC results.

Lattice Spacing	E_0 Simulated	E_0 Continuum Theory	Ratio
1.0	0.447128± 0.00012723	0.447214	0.999807
0.5	0.484628±0.000345008	0.485071	0.999087
0.1	0.498186±0.006664186	0.499376	0.997617

Table 4: This shows that Fourier Acceleration still agrees with the theoretical results.

5.3 Comparison of Fourier Acceleration to Regular HMC

To show that an improvement in speed has been achieved using Fourier Acceleration, a parameter was chosen which is known to a high accuracy and looked at how each of the simulations converged to that value. To do this values of that parameter were taken as the simulation evolved in computer time and the squared difference taken from the theoretical value and plotted the difference as a function of computer time

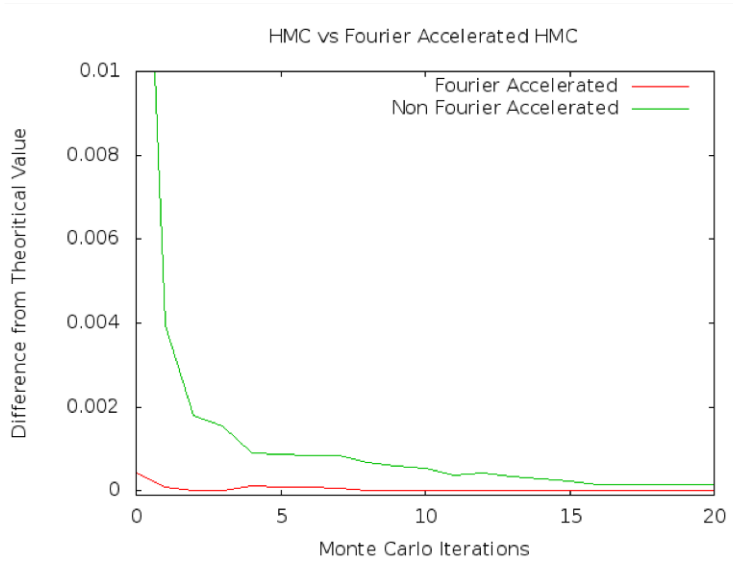


Figure 8: Comparison Graph between HMC and Fourier Accelerated HMC showing that the fourier acceleration converged to the theoretical value much quicker and more accurately than regular HMC

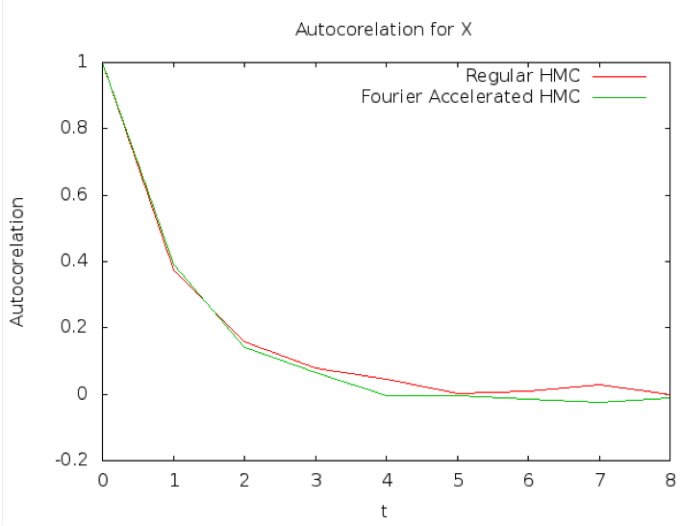


Figure 9: Autocorelation function for x in the Fourier Accelerated harmonic oscillator vs regular HMC

We can also examine the improvement in the integrated autocorrelation time when compared to Non Fourier Accelerated HMC. The autocorrelation time is defined as the amount of iterations (or time) which is needed for the two variables to be statically uncorrelated, i.e.they are not effected by each other. In the case of this simulation we measure the autocorrelation time along the Time Lattice and hence measures how independent two oscillators are. We want this to be as small as possible because in the continuum limit each oscillator in phase space is thought of as independent.

We can look at the integrated autocorrelation time to gain a more quantitative idea of this improvement.

Algorithm	Autocorrelation Time
HMC	1.014 $\pm$0.1
FA HMC	0.944 $\pm$0.11

Table 5: Integrated Autocorelation time for x

6 Conclusion

In this project I have shown that it is possible to use Hybrid Monte Carlo methods to simulate a Quantum Harmonic oscillator and anharmonic oscillator. These results have been in line with previous results from this area of study which have used similar methods. This implementation has been made general in the package which I have written and so theoretically any system which takes the form used in this project could be simulated using this piece of software.

I have also shown that it is possible to apply the Fourier Acceleration algorithm to a harmonic oscillator both in theory and practice, this was implemented into the software package i wrote.I simulated the Fourier Accelerated harmonic oscillator, proved it was still consistent with theoretical results and then showed that a speed up was achieved by examining the Autocorrelation time and the difference from the ground state energy compared to the regular HMC algorithm.

As a natural extension to this I created a new algorithm which could be used to improve the simulation for an anharmonic oscillator which ,as outlined in this report, has issues due to the complex nature of the potential compared to the harmonic oscillator. I attempted to include this algorithm into the package however there were a lot of technical issues which needed to be addressed and time constraints did not allow me to fully complete this project.

Whilst a speed up for the identified problem case has not been achieved some evidence has been presented which shows that this new algorithm has some potential to speed up the HMC algorithm within Lattice Field Theory. More work would need to be done to extend these results to a full Lattice Field theory simulation and a little more work would need to be done to implement the new anharmonic algorithm into the HMC simulation.

7 Acknowledgements

I would like to thank my supervisors Brian and Roger for their invaluable input through out this project. I would also like to thank Jack Frankland for his helpful discussions whilst starting off this project both inside and outside meetings.

Code

If you wish to examine the Package i wrote for this project all of the relavent files can be found here: https://github.com/QuidditchFIsh/Mphys2

The main Algorithm for Harmonic and Anharmonic Oscillator is located on the master branch with the fourier accelerated version on the branches fourier and Anharmonic-fourier respectively. The latter branches will need the FFTW package installed and the correct PATH set.the Analysis packages are all written for Python3.

References

[1] S Duane, A D Kennedy, B J Pendelton, and D Roweth. Hybrid monte carlo. *Physics Letters B*, 195(2):216–222, Sep 1987.

[2] Radford M. Neal. MCMC using Hamiltonian dynamics. 2012.

[3] M. Creutz and B. Freedman. A statistical approach to quantum mechanics. *Annals of Physics*, 132(2):427–462, 1981.

[4] R. Blankenbecler, T. Degrand, and R. L. Sugar. Moment method for eigenvalues and expectation values. *Physical Review D*, 21(4):1055–1061, 1980.

[5] Matteo Frigo and Steven G. Johnson. The design and implementation of FFTW3. *Proceedings of the IEEE*, 93(2):216–231, 2005. Special issue on "Program Generation, Optimization, and Platform Adaptation".

[6] Guido Cossu, Peter Boyle, Norman Christ, Chulwoo Jung, Andreas Jüttner, and Francesco Sanfilippo. Testing algorithms for critical slowing down. *EPJ Web Conf.*, 175:02008, 2018.

[7] Claudio Bonati and Massimo D'Elia. Topological critical slowing down: seven variations on a toy model. 2017.

[8] Lorenzo Bongiovanni. *Numerical methods for the sign problem in Lattice Field Theory*. PhD thesis, Swansea U., 2015.

[9] Richard P. Feynman, Robert B. Leighton, and Matthew L. Sands. *The Feynman lectures*. California Institute of Technology, 1963.

[10] Marise J. E. Westbroek, Peter R. King, Dimitri D. Vvedensky, and Stephan Dürr. User's guide to Monte Carlo methods for evaluating path integrals. 2017.

[11] Andreas Wipf. *Statistical approach to quantum field theory: an introduction*. Springer Berlin Heidelberg, 2013.

[12] Raul Toral and A. L. Ferreira. Generalized hybrid Monte Carlo. 1994.

[13] Jan Smit. *Introduction to quantum fields on a lattice: a robust mate*. Cambridge University Press, 2006.