Simulating the Fermi-Pasta-Ulam-Tsingou Problem

Abstract

This is the investigation of the Fermi-Pasta-Ulam-Tsingou problem, which is a series of masses connected by non-linear springs. A code was written in Python v3.10 which simulated the forces between particles, calculating and plotting a variety of characteristics about the system, including the modal energy; the total energy; the position of each particle and the average Lyapunov exponent (a quantitative measure of chaos). It was expected that the energy would start in one mode, and then spread out to all other modes, following the equipartition of energy theorem as the system becomes thermalised. However, when the simulation was run this happened initially, but after equipartition was reached, the energy then went almost entirely back into the first mode. As the simulation was allowed to run for longer times, super-recurrences were seen where the initial state is recovered with an even higher accuracy. This was investigated by varying the types (quadratic vs cubic) and the strength of forces, the initial displacements were varied, so they started with different amplitude and mode of sine wave.

The outcome of the investigation was that as strength of force, non-linearity of force, amplitude of oscillations and starting mode of initial sine wave increased, the shorter the lifetime of the recurrence region and the quicker it decayed into equipartition of energy. Abstract work count: 220. Report word count: 2,960

1 Introduction

In 1955 the MANIAC computer was christened by performing calculations to see how quickly thermalisation happens due to non-linear interactions.[1],[2]

This was achieved by modelling a line of particles connected by springs as an analogue for atoms in a one dimensional crystal. These springs obey Hooke's law but also have small non-linear forces between them (in this model it was the two simplest cases; quadratic and cubic).[3] They then induced displacements to the particles corresponding to the first normal mode (i.e. a half sine wave with fixed ends). If the force between particles was purely linear the energy should stay in the first mode, which was seen. However if non-linear are forces included it is expected that the energy stored in the first mode slowly seeps into other modes, causing the equipartition of energy is reached, so the system becomes thermalised.[3],[4] The simulation was run for a relatively short amount of time and this was seen with equipartition reached relatively quickly. However, when the simulation was accidentally left for a longer period of time, they saw that after equipartitioning was reached, the system then left it and 97% of the energy went back into the first mode in a recurrence.[3] As the program was allowed to run for longer times, this pattern repeated, to show super-recurrences where the initial state is recovered with even higher precision.[5]

2 Analysis of the Computational Physics Aspects of the Problem

2.1 Theory

A reason for this quasi-periodicity was proposed to be solitons (solitary wave pulse that travels along the chain) as these waves pass through each other without losing their identity.[6] The nature of solitons makes it harder for thermalisation and ergodicity to occur (i.e. for all modes to be equally activated, on average).[7]

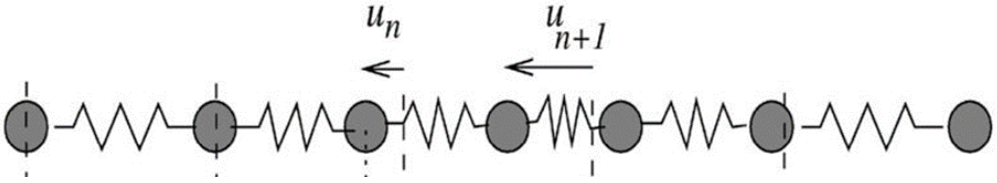


Figure 1: This shows a chain of particles connected by springs, indicating the n^{th} displacement and the $n+1^{th}$ displacement.[3]

The equation of motion for the n^{th} particle connected by linear springs is given by:

$$m\ddot{u}_{\mathrm{n}} = k(u_{\mathrm{n+1}} + u_{\mathrm{n\text{-}1}} - 2u_{\mathrm{n}}) \tag{1}$$

Where m = mass and k = linear spring constant, however in this problem we take them both to be 1, which gives:

$$\ddot{u}_{\mathrm{n}} = (u_{\mathrm{n+1}} + u_{\mathrm{n\text{-}1}} - 2u_{\mathrm{n}}) \tag{2}$$

If the quadratic and cubic forces are included, the equation is changed to:

$$\ddot{u}_{\mathrm{n}} = (u_{\mathrm{n+1}}+u_{\mathrm{n\text{-}1}}-2u_{\mathrm{n}})+\alpha((u_{\mathrm{n+1}}+u_{\mathrm{n}})^2-(u_{\mathrm{n}}+u_{\mathrm{n\text{-}1}})^2)+\beta((u_{\mathrm{n+1}}+u_{\mathrm{n}})^3-(u_{\mathrm{n}}+u_{\mathrm{n\text{-}1}})^3) \tag{3}$$

With N particles, and one fixed particle at either end (so N+2 particles), the energy in each mode, k, is given by:[3]

$$E_{\mathrm{k}} = \frac{\dot{A}_{\mathrm{k}}{}^2 + \omega_{\mathrm{k}}{}^2 A_{\mathrm{k}}{}^2}{2} \tag{4}$$

Where:[3]

$$A_{\mathrm{k}} = \sqrt{\frac{2}{N+1}} \sum_{n=1}^{N} u_{\mathrm{n}} \sin \frac{nk\pi}{N+1} \tag{5}$$

And:[3]

$$omega_{\mathrm{k}}{}^2 = 4 \sin \left(\frac{k\pi}{2N+2}\right)^2 Eqn[6] \tag{6}$$

2.2 Possible Computational Solutions

The main differences in computational solutions occur in the integration scheme; there are a variety of methods one could use. Since the focus of this problem is on energy, it is necessary to focus on an integrator that specifically conserves energy of the system. A symplectic integrator does this since it is a canonical transformation and hence conserves the Hamiltonian.[8] There are various orders of symplectic integrator where the higher the order of the integrator, the more accurate it is.

One of the simplest forms of this integrator is the 2^{nd} order velocity Verlet integration scheme, which is executed in four separate steps:[8]

$$v(t + \frac{\Delta t}{2}) = v(t) + \frac{1}{2}a(t)\Delta t \tag{7}$$

$$u(t+\Delta t) = u(t) + v(t+\frac{\Delta t}{2})\Delta t \tag{8}$$

$$\text{Calculate } a(t+\Delta t) \text{ using } x(t+\Delta t) \tag{9}$$

$$(t+\Delta t) = v(t+\frac{\Delta t}{2}) + \frac{1}{2}a(t+\Delta t)\Delta t \tag{10}$$

Then iterate this for each particle, where *u(t)*, *v(t)* and *a(t)* are the positions, velocities, and accelerations of particles at time, t. However, since the acceleration only depends on position, equation 7 can be implemented in equation 8, and then combined with equation 9 to produce 3 steps:

$$u(t+\Delta t) = u(t) + v(t) + \frac{1}{2}a(t)\Delta t \tag{11}$$

$$\text{Calculate } a(t+\Delta t) \text{ using } x(t+\Delta t) \tag{12}$$

$$v(t+\Delta t) = v(t) + \frac{a(t)+a(t+\Delta t)}{2}\Delta t \tag{13}$$

For reasons discussed in section 4.1, this was the integration scheme used, however two other integration methods that were investigated: symplectic 4^{th} order and the Runge-Kutta 4^{th} order integrator. Both schemes have more steps by a factor of 2-3, so should take a longer time to execute, but have differing levels of accuracy (symplectic 4^{th} order should be more accurate than the Verlet scheme and the Runge-Kutta method should be less accurate). Their steps are described in the appendix.

3 Details of Implementation of Algorithm

This problem was simulated in Python (v3.10) using the symplectic 2^{nd} order, 4^{th} order and Runge-Kutta 4^{th} order integration schemes. In this program there are N+2 particles with two fixed ends, leaving N moving particles. The initial displacement is a sine wave of variable mode and amplitude, and for every simulation, there is a second string of masses identical to the first, except the middle N masses are displaced by epsilon. This is to investigate the chaotic nature of the system by calculating the average Lyapunov exponent of the simulation.

The energy of the first six modes is plotted against time and against the number of time periods of the first mode. The corresponding heatmap is also shown. The energy of the system is calculated (by integrating all forces and

adding to the kinetic energy) and plotted against time. To check if energy is conserved, the average start and final energies are calculated and displayed as well. The energy of each recurrence relative to the original energy is shown on a graph to investigate how these change with time. The Lyapunov exponent is calculated and averaged throughout the simulation and plotted against time to indicate how chaotic the system is. If the exponent is positive it indicates the system is chaotic. If it's negative, the system attracts to a stable point (non-chaotic), and if it's zero, the system is in a steady state and paths altered by a small amount stay roughly altered by that same amount.[9] The phase space of a particle is shown for the first few steps, with position on the x axis and velocity on the y axis, as well as contour plot of the position, with time on the x axis and particle position on the y axis. A short MP4 file is also downloaded which is an animation of the particles' position over a short period of time at the start of the simulation. The specific implementation of some functions is described in the appendix (e.g. calculation of the Lyapunov exponents).

4 Results and Discussion

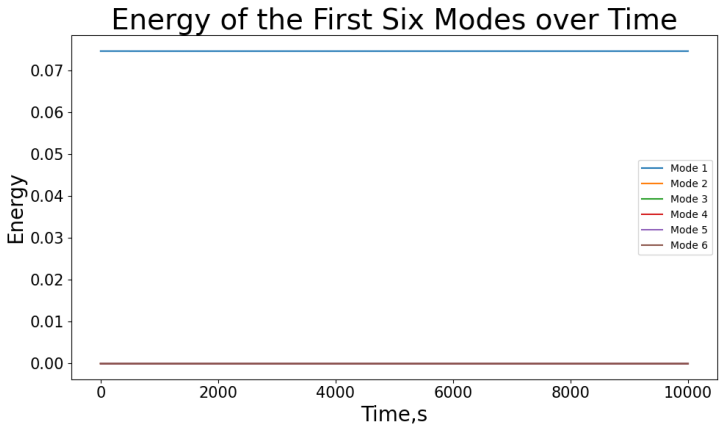
The results shown are for 32 moving particles (34 total particles) and have the starting positions of a half sine wave or a full sine wave with fixed ends. To investigate my simulation I will vary the starting displacements; starting amplitude; quadratic and cubic force.

4.1 Investigating Different Integration Schemes

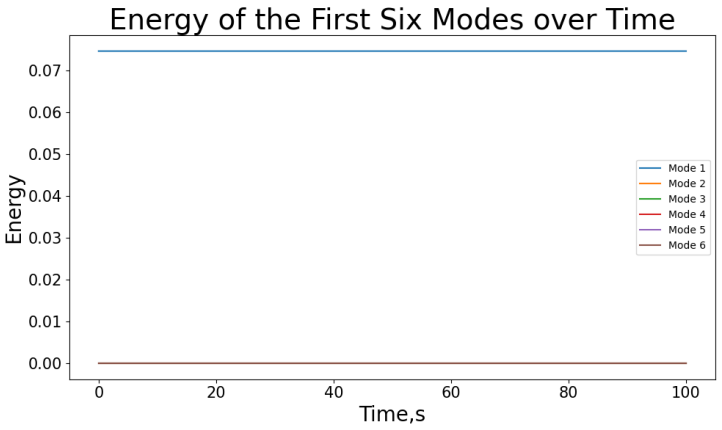
In my code there are three different integration schemes that can be used, Verlet scheme, symplectic 4^{th} order and Runge-Kutta 4^{th} order. These three schemes will be investigated in the linear regime to determine which is best to run the rest of the simulations on.

The first analysis was under-taken for the Verlet scheme, with the starting mode being a half sine wave of amplitude 1.0 and time-step of 0.3. Figure 2(a) shows the energy staying constant in the first mode, as expected, figure 2(b) shows the same thing, except under a shorter time-scale. The percentage energy difference from start to end is $3.36*10^{-4}\%$ and the time taken for Google Colab to run the simulation of 10,000 seconds with this method was 16.60 seconds. However if there are quadratic forces then the percentage difference

is $3.33*10^{-3}\%$. When the Runge-Kutta 4^{th} order method was used, the graphs look the same, the only difference in the percentage energy difference, which is $2.47*10^{-5}\%$ and the time taken to run it, being 26.36 seconds. However percentage energy difference is 0.15% when there are quadratic forces. If the 4^{th} order symplectic scheme was used, the energy oscillates with an amplitude of roughly 0.001, (roughly 3% of the total energy), with a percentage energy change of 0.010%, if it is plotted on a shorter time-scale it is easier to see the oscillations (shown in figure 3(c)). If a time-step of 0.03 is used (figure 3(d)), the amplitude of oscillations decreases significantly to almost 0, however the time taken to execute the program increases from 27.57 seconds, with a 0.3 time-step, to 185.44 seconds with a 0.03 time-step for 10,000 seconds of simulation. The percentage energy difference is 0.018% with quadratic forces, using a 0.3 time-step. The percentage energy differences show the Verlet scheme is more accurate than the other two with non-linear forces, however with linear forces the Runge-Kutta method is most accurate. The accuracy of the 4^{th} order symplectic scheme stays roughly constant, despite the quadratic forces, this integration scheme is consistently accurate but the least stable. Since we want to preserve precision while making the program as efficient as possible, the Verlet scheme will be used, as it is most accurate for non-linear forces, taking the least amount of time.

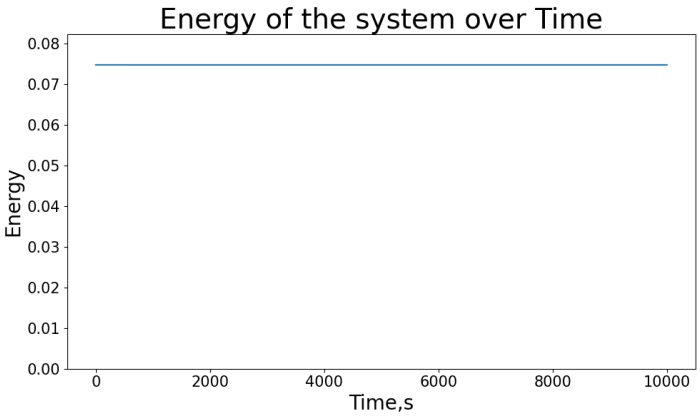


(a) Energy of the first 6 modes over time, using the Verlet scheme.

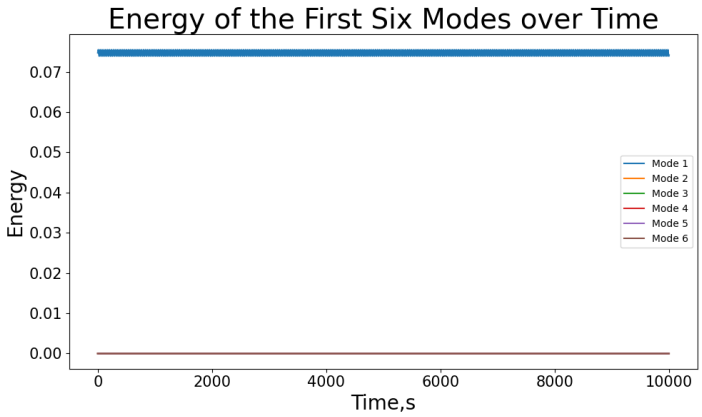


(b) Energy of the first 6 modes over a shorter time, using the Verlet scheme.

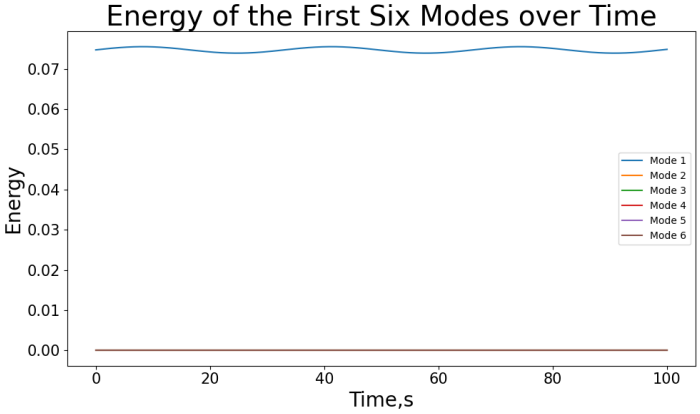
Figure 2



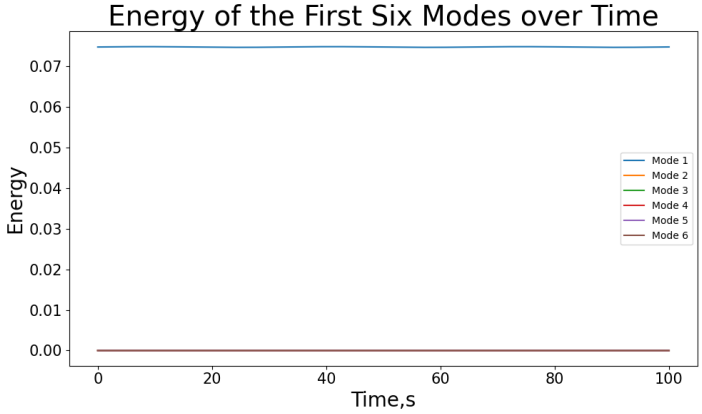
(a) Energy of the system over time, using Verlet scheme.



(b) Energy of first 6 modes over time, using symplectic 4^{th} order scheme.



(c) Energy of first 6 modes over shorter time, using symplectic 4^{th} order scheme, time-step of 0.3



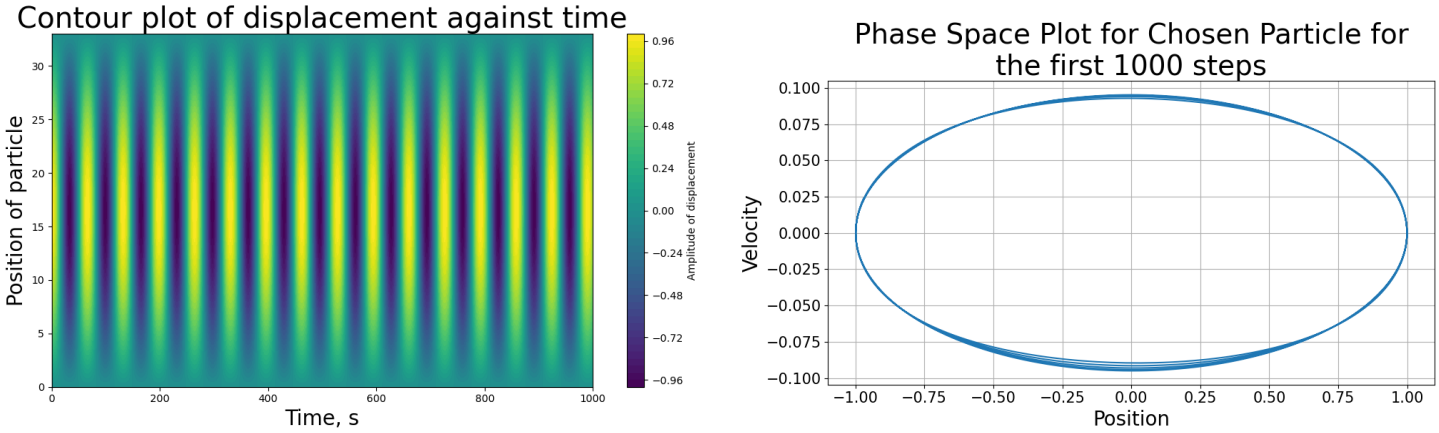
(d) Energy of first 6 modes over shorter time, using symplectic 4^{th} order scheme, time-step of 0.03

Figure 3

4.2 Linear Springs

To test my simulation in a known scenario the system was run with linear forces, since it is known that the energy should stay in one mode. In the last section a linear spring was simulated and worked as expected, since in figure 2(a) the energy of first mode stays constant and all other modes gain no energy. The total energy is also shown to be exactly constant (figure 3(a)) at the same value as the mode 1 energy, which shows that both methods

of calculating the energy (equation (4) and the method described in section 3) agree. Figures 4(a) and (b) show the contour plot of the displacements against time and phase space of particle 17, these serve as sanity checks and show the code works ideally for a linear spring.



(a) Contour displacement of particles.

(b) Phase space plot of the 17^{th} particle.

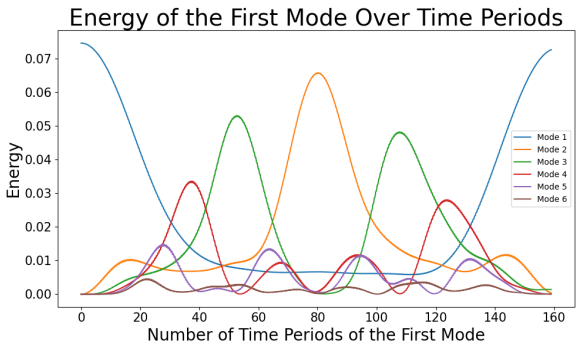
Figure 4

4.3 Quadractic Springs

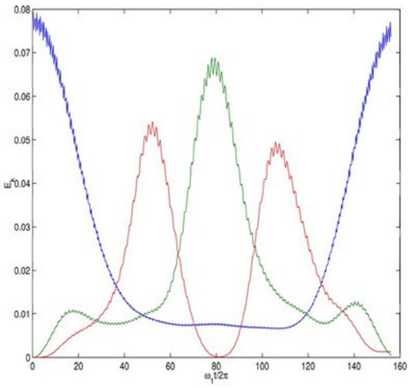
4.3.1 $\alpha = 0.25$, $\beta = 0.0$, $Amplitude = 1.0$

The system was simulated using weak quadratic forces with a half sine wave initiation. Figure 5(a) shows the energy of the first six modes vs time-period of the first mode. This is very similar to the simulation in [3] (figure 5(b)), with the same parameters and initial starting positions. They both have a recurrence at roughly 160 time periods. Figure 5(c) shows the heatmap of modal energy which makes it easier to spot characteristics of the plot, such as after the second mode, each mode gains two more peaks in modal energy, whilst getting weaker. In the contour plot in figure 5(d) there is a region of not much displacement in the slightly darker "X". Both characteristics need to be investigated in more depth to determine their origins.

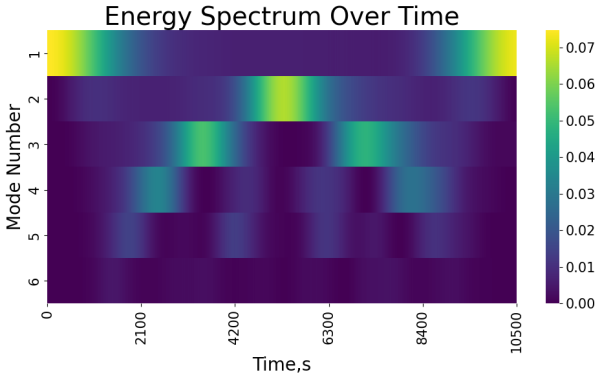
If the simulation is allowed to run for a longer period, a super-recurrence is seen (figure 5(e)). The period of this is approximately 250,000 seconds. The Lyapunov exponent is $1.25*10^{-5}$, which implies the system is not very



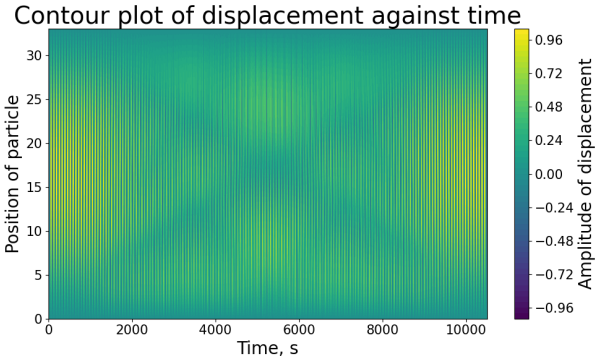
(a) Energy of the first 6 modes with quadratic forces, with initial half-sine wave displacement.



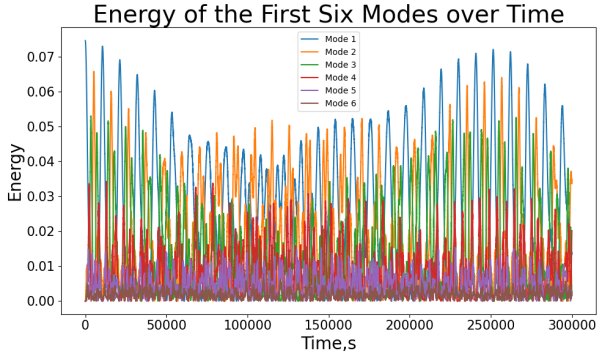
(b) Energy of the first 6 modes with quadratic forces, with initial half-sine displacement.[3]



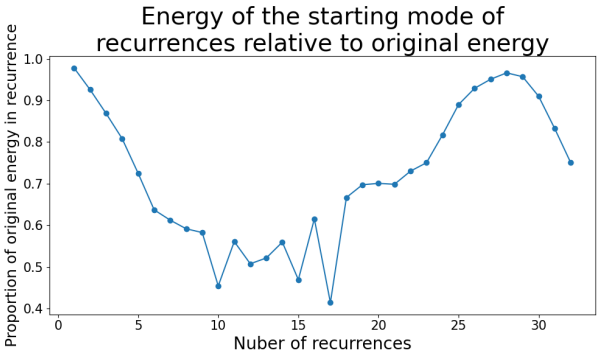
(c) Heatmap of the energy of the first 6 modes with quadratic forces.



(d) Contour plot of displacements of particles for the entire simulation.



(e) Energy of the first 6 modes over a long period of time, with initial half-sine displacement.

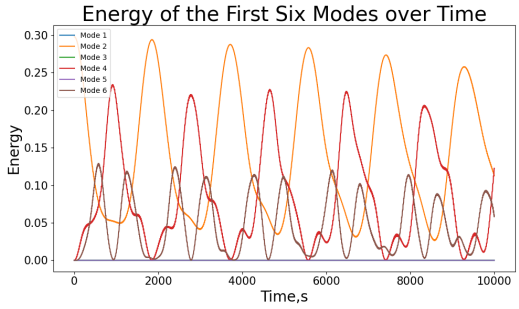


(f) Relative energy of recurrences of the first mode compared to original energy.

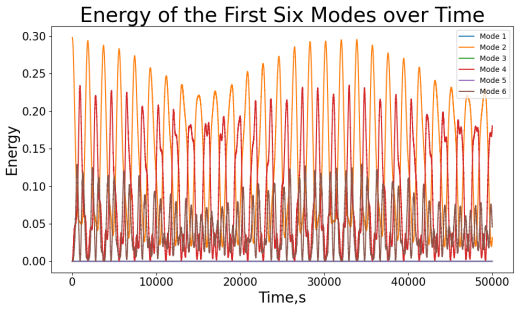
Figure 5

chaotic, as seen. There are also super-recurrences seen for other modes, odd modes have the same approximate time-period, however even modes have a slightly different recurrence mode pattern, with the energy of their recurrences oscillating more erratically. The animation of the positions for the first 500 time-steps is shown in the "Position_Animation" file.

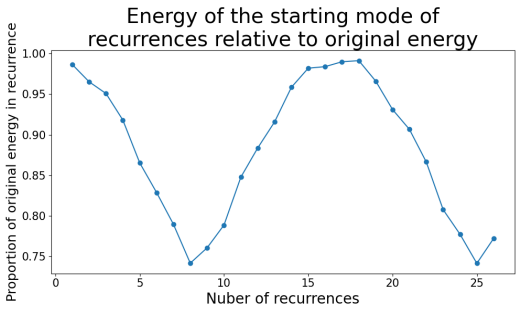
When the system was initialised with a full sine wave a similar pattern of recurrences was seen, except it only activated even modes. The pattern of number of modal energy peaks increasing by two is seen, however, only on every second mode. If left for long periods of time, a super-recurrence is seen in the second mode (figure 6(b)), however after a much shorter period than for the half sine wave initialisation, with time-period of roughly 35,000 seconds.



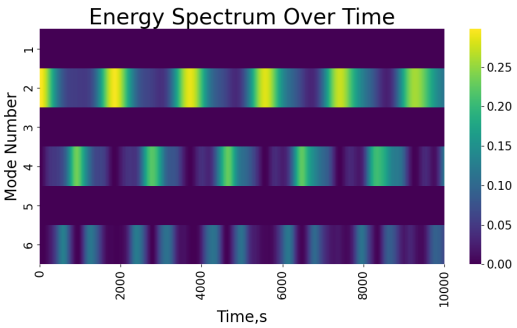
(a) Energy of the first 12 modes over time, with initial sine wave displacement.



(b) Energy of the first 6 modes over a longer period of time, with initial sine displacement.



(c) Relative energy of recurrences of the second mode compared to original energy.

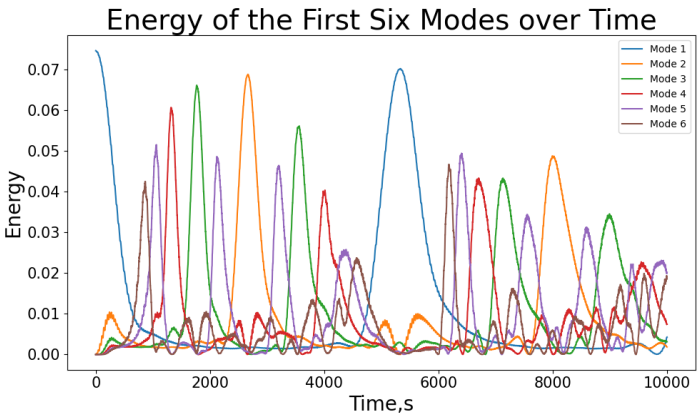


(d) Heatmap of the energy of the first 6 modes with quadratic forces, with initial sine displacement.

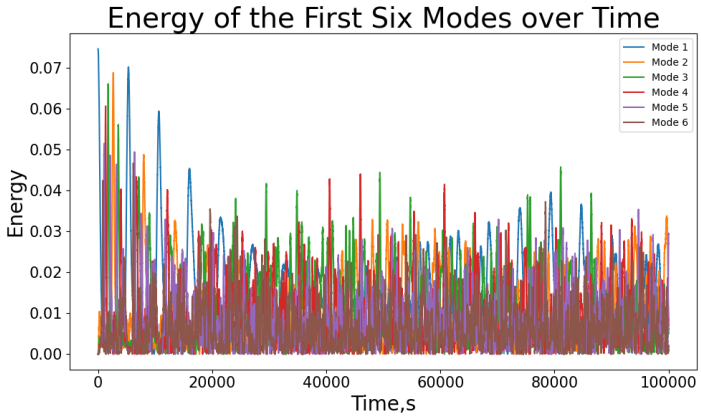
Figure 6

4.3.2 $\alpha = 1.0$, $\beta = 0.0$, $Amplitude = 1.0$

When investigated with stronger quadratic forces, it is seen that the period for recurrence to occur is approximately half the time when alpha was 0.25, and the spike of energy of each different mode is much higher, however, the pattern is still similar to before. If left for long periods of time there is a super-recurrence: this time its time-period is roughly 80,000 seconds, and the energy returned to the original mode is far less than at the start. It is obvious the system is more chaotic, still with some structure. This implies the onset of ergodicity since all modes are being somewhat activated. This is reflected in the Lyapunov exponent, which is 0.000930 (still small but almost 10-fold larger than previously).



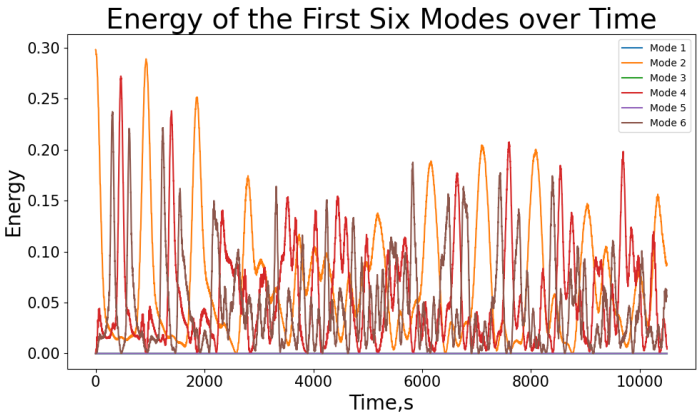
(a) Energy of first 6 modes with stronger quadratic forces and half-sine initialisation.



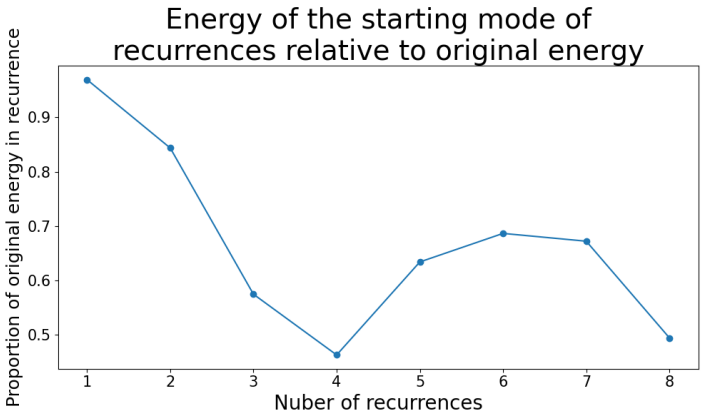
(b) Energy of first 6 modes with stronger quadratic forces over a longer period of time.

Figure 7

This was repeated for an initial displacement corresponding to a sine wave, and the super-recurrence is visible, but with a lot more noise than previously, this is indicated in the Lyapunov exponent of 0.00180. This shows that as quadratic forces increase, ergodicity increases as well.



(a) Energy of first 6 modes with stronger quadratic forces and sine wave initialisation.



(b) Relative energy of recurrences of the second mode compared to original energy.

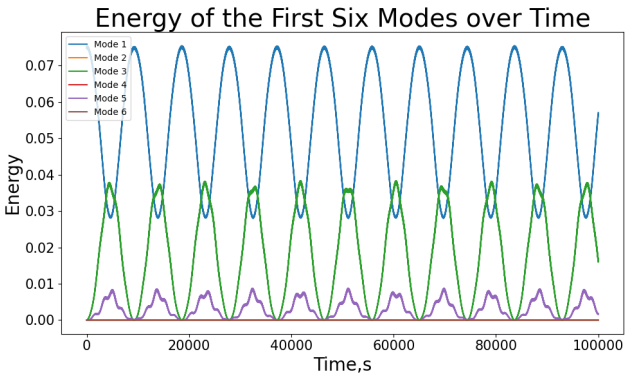
Figure 8

4.4 Cubic Springs

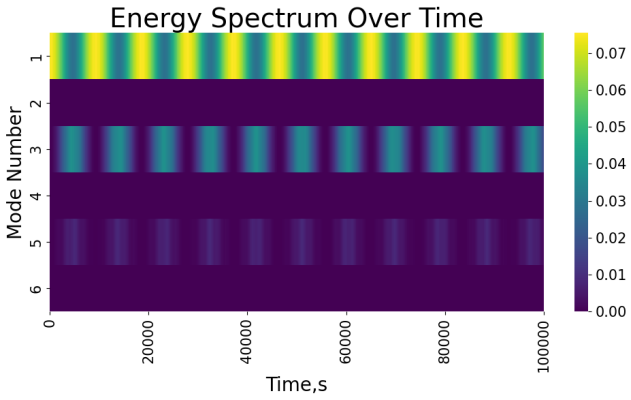
4.4.1 $\alpha = 0.0$, $\beta = 3.0$, $Amplitude = 1.0$

Figure 9(a) shows the modal energy against time-period of initial mode with initial displacement of a half sine wave under cubic forces. Only odd numbered modes are activated, and there is a recurrence in all of them (however increasingly weaker as the mode number increases), with time period of approximately 10,000 seconds. However, the energy of each recurrence is approximately equal, with the period of recurrence being far shorter than for the quadratic case.

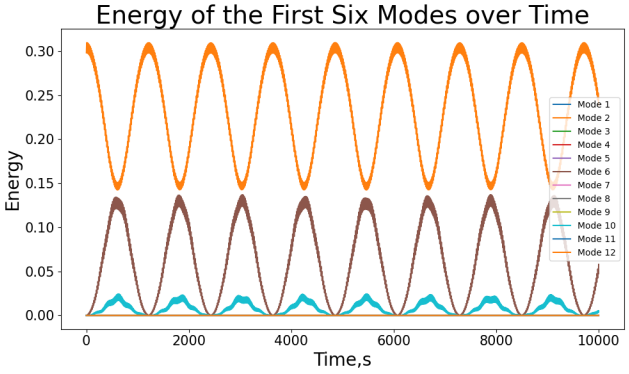
The simulation was repeated for an initial displacement of a sine wave, which produces figure 9(c) similar to figure 9(a), except with far shorter period for recurrence. It activates mode 2 and every fourth mode after that. There are recurrences seen in all activated modes, with the same time-period of roughly 1,300 seconds, however the minimum energy of the second mode is limited to approximately half of the original energy. If it is left for long periods of time, the original pattern suddenly decays and a more chaotic system occurs, showing the onset of the equipartition of energy, this is demonstrated by an even larger Lyapunov exponent, 0.00132. There is still some pattern since there are recurrences of all modes in this regime. The chaotic nature would require more investigation to figure out why it happens.



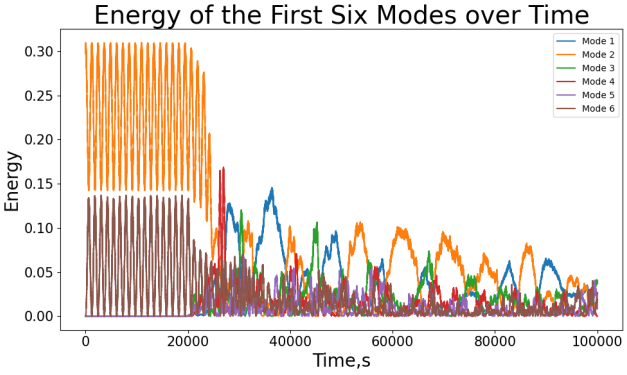
(a) Energy of first 6 modes with cubic forces and half-sine initialisation.



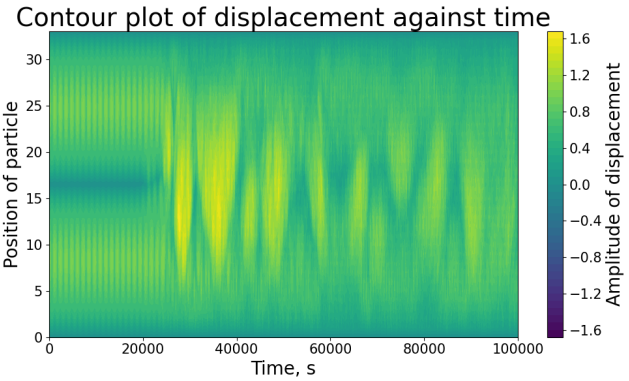
(b) Heatmap of energy of first 6 modes with cubic forces.



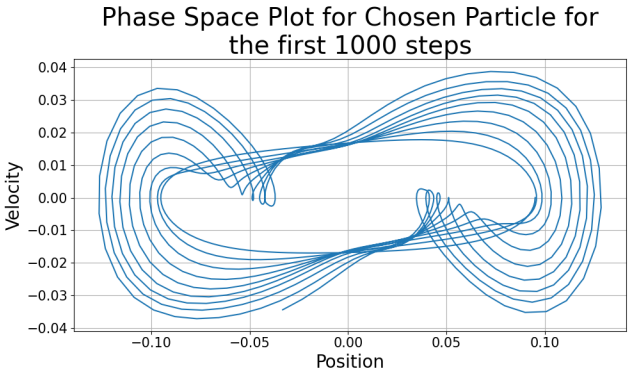
(c) Energy of the first 12 modes with cubic forces, with sine wave initialisation.



(d) Energy of the first 6 modes with cubic forces over a longer time.



(e) Contour displacement plot over long period of time with cubic forces.



(f) Phase space of the 17^{th} particle, with cubic forces.

Figure 9

4.4.2 $\alpha = 0.0$, $\beta = 6.0$, $Amplitude = 1.0$

When stronger cubic forces are introduced, a similar graph is seen to figure 9(a), except the frequency of recurrences is increased, the minimum energy of mode 1 is decreased and it decays into semi-chaotic behaviour, still with recurrence of various modes, specifically the second mode most. This, again, implies the recurrence regime has a lifetime before it decays into a more chaotic, but still somewhat controlled system. The Lyapunov exponent is 0.000461.

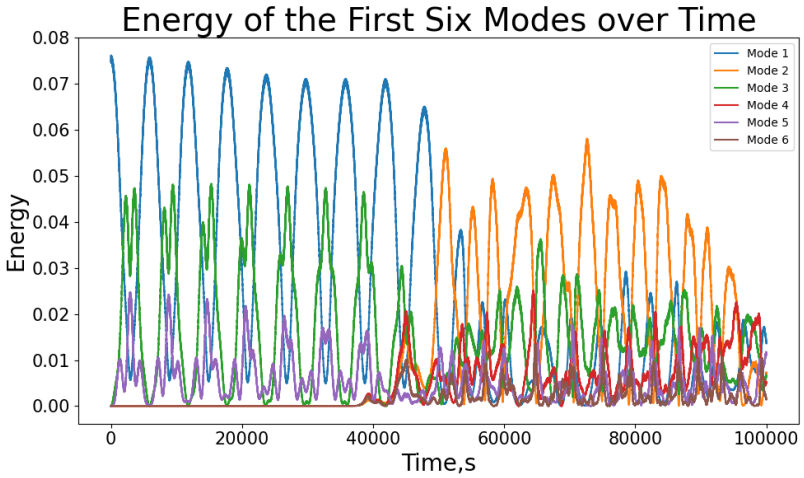
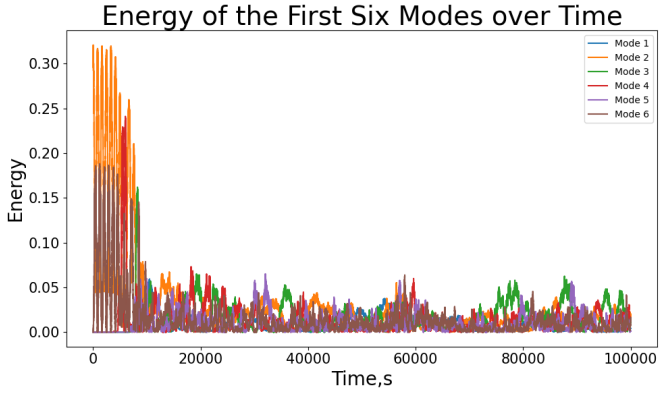
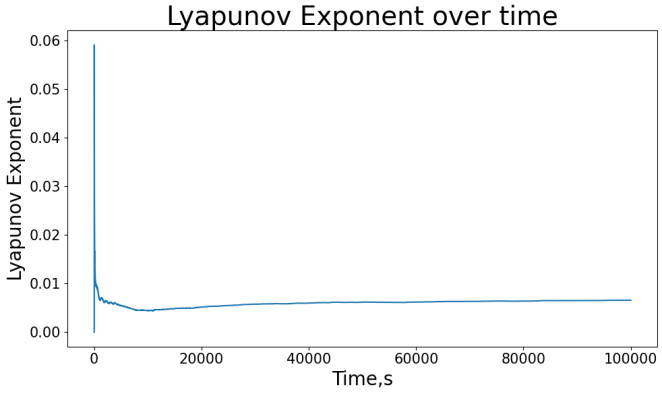


Figure 10: Energy of first 6 modes with stronger cubic forces and half-sine initial displacement.

When the initial mode is 2, the same thing is seen as when $\beta = 3.0$, except the frequency of recurrences is increased, the minimum energy of mode 2 is decreased and ergodicity sets in much faster. The system seems more chaotic, with fewer recurrences in this chaotic regime than previously, this is shown by a higher Lyapunov exponent, of 0.00657.



(a) Energy of the first 6 modes, with stronger cubic forces and sine wave initialisation.

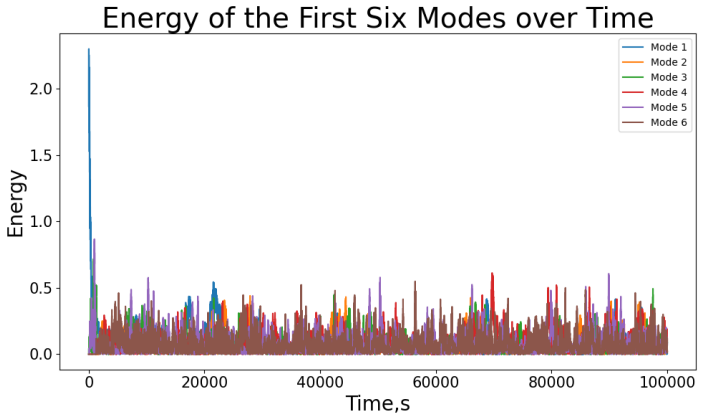


(b) Lyapunov exponent over time, with strong cubic forces and sine wave initial displacement.

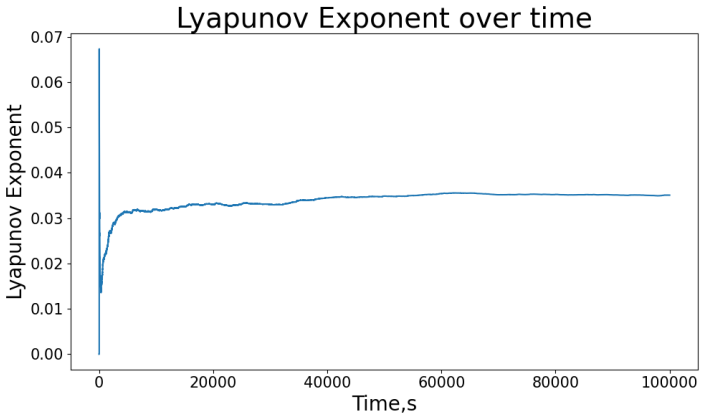
Figure 11

4.4.3 $\alpha = 0.0$, $\beta = 3.0$, $Amplitude = 5.0$

As expected, as the amplitude increases, total energy also increases, however the lifetime of predictable recurrences decreases dramatically, with the system decaying almost straight away to chaotic ergodicity. This is implied by the largest Lyapunov exponent yet when the starting mode is 1, which is 0.0351. When the starting mode is 2, the model crashes because the system is extremely chaotic.



(a) Energy of first 6 modes with cubic forces, using a half-sine initialisation with a larger amplitude.



(b) Lyapunox exponent over time, with cubic forces and large amplitude half-sine initialisation.

Figure 12

4.5 Quadratic and Cubic Springs

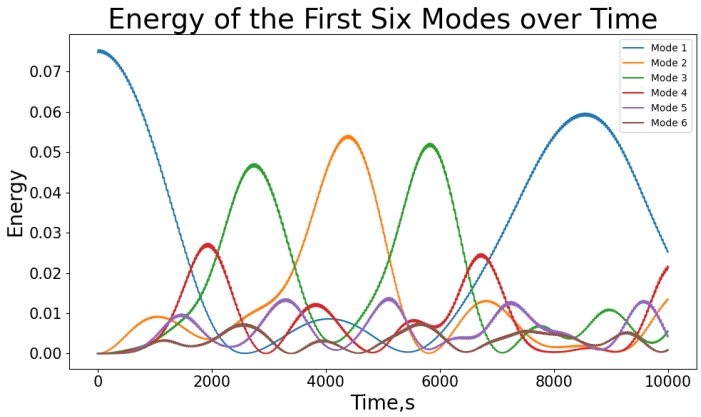
4.5.1 $\alpha = 0.25$, $\beta = 3.0$, $Amplitude = 1.0$

With cubic and quadratic forces, the initial modal energy plot (figure 13(a)) looks similar to a purely quadratic one, except slightly skewed towards the origin. As this is extended for longer periods, it looks somewhat like a quadratic energy plot, with recurrences of varying energy, however more chaotic than previously. It has a Lyapunov exponent of 0.000377.

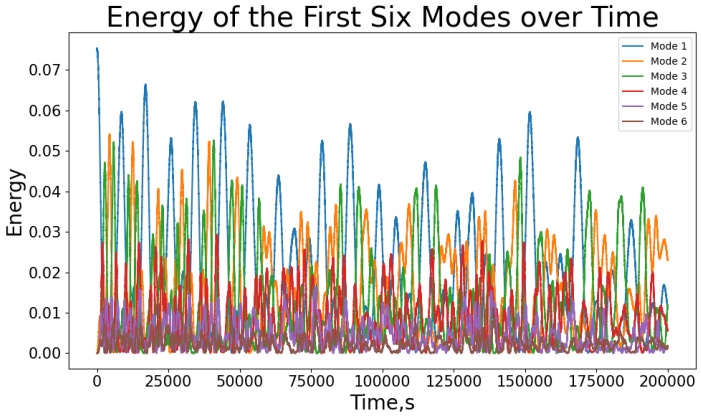
4.5.2 $\alpha = 1.0$, $\beta = 6.0$, $Amplitude = 1.0$

If the quadratic and cubic forces are stronger and left for a long time, a similar plot (figure 13(d)) is seen to figure 7(b), except all recurrences have

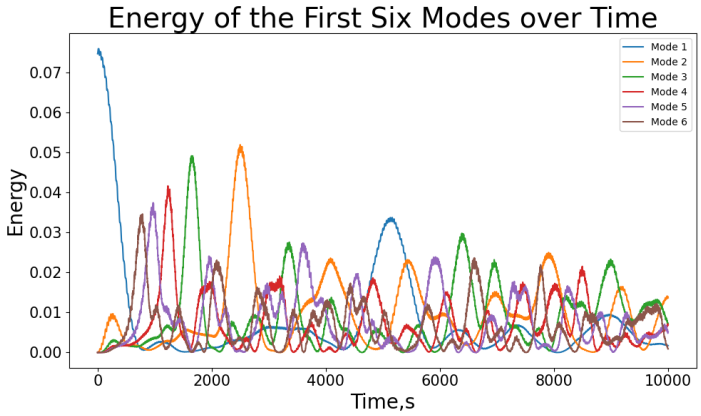
lower energies. The time-period of recurrence in the first mode is smaller and it decays into a more chaotic regime, which agrees with its Lyapunov exponent of 0.00208.



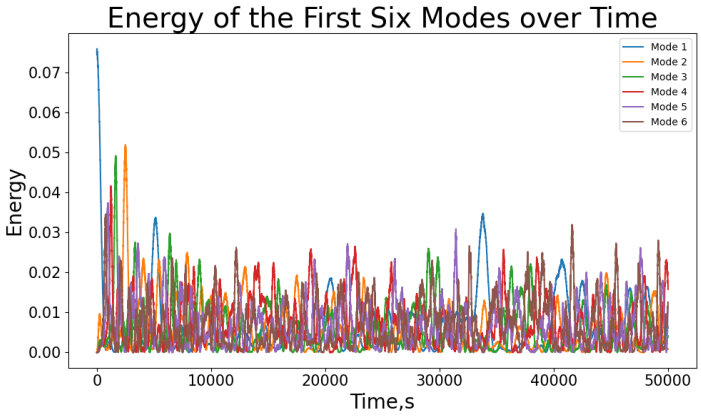
(a) Energy of first few modes using weak quadratic and cubic forces, with half-sine initialisation.



(b) Energy of first few modes using weak quadratic and cubic forces, over a longer period of time.



(c) Energy of first few modes using strong quadratic and cubic forces, with half-sine initialisation.



(d) Energy of first few modes using strong quadratic and cubic forces, over a longer period of time.

Figure 13

5 Conclusion

This code has simulated the FPUT problem, and shows the regime where the equipartition of energy does and does not happen. The findings varied depending on strength of force; which non-linearity it was; amplitude of the original starting displacements and mode of the original sine wave. In general, the higher the starting mode; the larger the amplitude; the stronger the force and the higher the non-linearity of force; the quicker the system decays into ergodicity. The specific details of the investigation are as follows:

When quadratic forces are introduced, the system exhibits recurrences of the original mode (generally decreasing then increasing), but at a different energy to the original energy, and a super-recurrence, where the energy of the original mode is almost completely regained. When the starting mode was 1, all modes showed recurrences, and when it was 2, only the even modes exhibited recurrences. As this quadratic force was increased, the system became more chaotic, still displaying super-recurrences, but much lower energy.

If cubic forces were used and the starting mode was 1, recurrences of every odd mode are seen, seeming very stable. However, if the starting mode is 2, there are recurrences seen in the second mode, and every fourth mode after that. It also decays to equipartition of energy, still with some modes recurring. If the strength of the cubic force is increased, and the starting mode is 1, the system decays to semi-chaotic behaviour, still with some recurrences. If the starting mode is 2, the ergodic behaviour takes over far quicker, and if the amplitude of the starting displacements increases, the lifetime of recurrence behaviour decreases further.

When there were cubic and quadratic forces, the plot of modal energies looks like when there were just quadratic forces, however as time went on, it was still mostly ordered but it had a slightly different pattern to the quadratic case.

If this investigation was continued further, it would be a good idea to look into when the system turns from having defined recurrences to chaotic, and quantify this in terms of amplitude, type of force, strength of force and starting mode. It would also be useful to investigate why only some modes are activated during the simulation, as well as the pattern where the number of modal energy peaks increases by two for each mode. Investigation into the simulation using more particles, different masses or different types of non-linear forces would be interesting.

A Different Integration Schemes

A slightly more complex integrator than the Verlet method, but more accurate, is the 4th order symplectic integrator, which uses a similar method to the velocity Verlet scheme, but with different coefficients, given by Yoshida:[8]

$$c_1 = c_4 = \frac{1}{2(2 - 2^{\frac{1}{3}})} \tag{14}$$

$$c_1 = c_4 = \frac{1 - 2^{\frac{1}{3}}}{2(2 - 2^{\frac{1}{3}})} \tag{15}$$

$$d_2 = \frac{-2^{\frac{1}{3}}}{2 - 2^{\frac{1}{3}}} \tag{16}$$

$$d_2 = \frac{-2^{\frac{1}{3}}}{2 - 2^{\frac{1}{3}}} \tag{17}$$

$$d_4 = 0 \tag{18}$$

Then iterate through j = 1 to 4 for every particle at every time step (i):

$$x_j{}^{\mathrm{i}} = x_j{}^{\mathrm{i\text{-}1}} + c_i v_j{}^{\mathrm{i\text{-}1}} \Delta t \tag{19}$$

$$\text{Calculate } a_j{}^{\mathrm{i}} \text{ using } x_i{}^{\mathrm{j}} \tag{20}$$

$$v_j = v_j + d_i a_j{}^{\mathrm{i}}(t) \Delta t \tag{21}$$

$$x_{j+1} = x_j{}^4 \tag{22}$$

$$v_{j+1} = v_j{}^4 \tag{23}$$

This requires double the number of computations per time step but should produce a more accurate solution in the end.

To compare to a non-symplectic integrator and investigate how much better symplectic integrators are than their counterparts, I decided to implement the Runge-Kutta 4th order integration scheme, as well. Because the velocity and position both need to be iterated, the RK4 method is used for both the position and velocity, and is given by:[9]

$$k_1 = hf(x_n, y_n) \tag{24}$$

$$k_2 = hf(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_1) \tag{25}$$

$$k_3 = hf(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_2) \tag{26}$$

$$k_4 = hf(x_n + h, y_n + k_3) \tag{27}$$

$$y_{n+1} = y_n + \frac{k_1 + 2k_2 + 2k_3 + k_4}{6} + O(h^5) \tag{28}$$

B Structure of Source Code

The code is all included in one file with the first section importing the necessary packages and libraries; the second part initialising all functions that are needed; the third section running the loop of the simulation; and the fourth part part plotting various graphs to investigate the simulation.

The first function initialises all arrays that are needed for the simulation and the second initialises the displacements of each particle. The next three functions calculate specific properties of the system, this is the energy in each mode, the total energy and the acceleration of each particle due to the springs. The next four functions are the three different integration schemes; a symplectic 2^{nd} and 4^{th} order integrator and a Runge-Kutta 4^{th} order scheme, and the integrator selector. The subsequent two functions are analysis; calculating the Lyapunov exponent and the peaks in modal energies. The final nine functions are plotting graphs of various aspects of the simulation, such as modal energy over time and a contour plot of the positions over time. Subsequently the system is simulated, the parameters are defined, such as the number of particles, and quadratic and cubic force coefficients, then the arrays are initialised. Two sets of arrays for position, velocity and acceleration, which are slightly perturbed from each other, are defined to calculate the Lyapunov exponent. The simulation is then looped over the number of steps to calculate the motion of particles and properties of the system as it progresses. Finally all graphs are plotted using the functions defined earlier.

In this code the NumPy library [10] is used for arrays and maths tasks, the Matplotlib [11] library is used to plots graphs and animations, the gaussian_filter1d package from the SciPy [12] library is used to smooth out the modal energy over time to find the maximum. The time [13] library is used to determine the run time of the program and the Seaborn [14] library is used to create a heatmap of the modal energy.

C Implementation of Various Aspects

C.1 Runge Kutta 4^{th} Order

The code uses vectorised calculation of arrays, since, if the NumPy package is used, this is much faster than using for loops to do calculations.[15] This was used in any calculation which was iterated over all the particles.

The first two integration schemes were easy to integrate into code, however the Runge-Kutta 4th order method proved more difficult, instead of the basic k values given earlier, in section 2.2, the k values were calculated using these formulae:

$$k_{1,\text{pos}} = v(t-\Delta t)\Delta t, k_{1,\text{vel}} = a(t-\Delta t, u)\Delta t$$

$$k_{2,\text{pos}} = (v(t-\Delta t) + \frac{k_{1,\text{pos}}}{2})\Delta t, k_{2,\text{vel}} = (a(t-\Delta t, u) + \frac{k_{1,\text{pos}}}{2})\Delta t$$

$$k_{3,\text{pos}} = (v(t-\Delta t) + \frac{k_{2,\text{pos}}}{2})\Delta t, k_{3,\text{vel}} = (a(t-\Delta t, u) + \frac{k_{2,\text{pos}}}{2})\Delta t$$

$$k_{4,\text{pos}} = (v(t-\Delta t) + \frac{k_{3,\text{pos}}}{2})\Delta t, k_{4,\text{vel}} = (a(t-\Delta t, u) + \frac{k_{3,\text{pos}}}{2})\Delta t$$

$$u(t) = u(t-\Delta t) + \frac{k_{1,\text{pos}} + 2k_{2,\text{pos}} + 2k_{3,\text{pos}} + k_{4,\text{pos}}}{6}$$

$$v(t) = v(t-\Delta t) + \frac{k_{1,\text{vel}} + 2k_{2,\text{vel}} + 2k_{3,\text{vel}} + k_{4,\text{vel}}}{6}$$

C.2 Energy in Each Recurrence

To investigate how much energy is transferred back to the original mode, the peak of each super-recurrence is found. However since the energy of each mode has a very small, but very frequent oscillation, the maximum energy of each super-recurrence was calculated by smoothing the function out, using the gaussian_filter1d package from scipy.ndimage [15], and then just compared adjacent points to see if it was greater than the last and the next point. This finds where the maximum roughly is, then the maximum value from within 100 time steps of this point was determined to find the true maximum.

C.3 Lyapunov Exponent of System

To investigate the chaotic nature of the system, the Lyapunov exponent was calculated at each time step and averaged over the simulation. To calculate the exponent, the formula below was used:[16]

$$\lambda = \lim_{t\to\infty} \lim_{\delta Z(0)\to 0} \frac{1}{t} \log \frac{|\delta Z(t)|}{|\delta Z(0)|}$$

Where $\delta Z(t)$ is the divergence between two paths at a time t, the limit means the perturbation is very small at the beginning. The simplest divergence between two paths is the Euclidean norm, which is given by:[17]

$$Divergence = \sqrt{\sum_{i=0}^{N+2} (x_{\mathrm{i}} - x_{\mathrm{i}}{}')^2 + (v_{\mathrm{i}} - v_{\mathrm{i}}{}')^2}$$

Where x_i and v_i are the displacement and velocity of the i^{th} particle, and $x_i{}'$ and $v_i{}'$ are that of the perturbed simulation. In my code, if the two simulations ever strayed too far apart from each other, the perturbed system is rescaled, bringing the two systems closer together, since the idea of chaos is the measurement of how quickly small changes grow, the calculation becomes useless if the two simulations stray too far away from each other.

C.4 Animation

The animation works by storing a list of subsequent scatter plots of the system, using the position data stored during the simulation. It then cycles through them, showing each plot for a fixed amount of time (e.g. 10ms).

References

[1] G. P. Berman and F. M. Izrailev, "The Fermi-Pasta-Ulam problem: 50 years of progress," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 15, no. 1, p. 015104, Mar. 2005, doi: 10.1063/1.1855036.

[2] E. Fermi, P. Pasta, S. Ulam, and M. Tsingou, "STUDIES OF NONLINEAR PROBLEMS," Los Alamos National Laboratory (LANL), Los Alamos, NM (United States), LA-1940, May 1955. doi: 10.2172/4376203.

[3] T. Dauxois and S. Ruffo, "Fermi-Pasta-Ulam nonlinear lattice oscillations," *Scholarpedia*, vol. 3, no. 8, p. 5538, Aug. 2008, doi: 10.4249/scholarpedia.5538.

[4] A. Larkoski, "Lecture Notes on the Fermi-Pasta-Ulam-Tsingou Problem." Accessed: Mar. 02, 2024. [Online]. Available: `https://www.physics.ucla.edu/~larkoski/FPUT.pdf`

[5] J. Ford, "The Fermi-Pasta-Ulam problem: Paradox turns discovery," *Physics Reports*, vol. 213, no. 5, pp. 271–310, May 1992, doi: 10.1016/0370-1573(92)90116-H.

[6] N. J. Zabusky and M. D. Kruskal, "Interaction of 'Solitons' in a Collisionless Plasma and the Recurrence of Initial States," *Physical Review Letters*, vol. 15, no. 6, pp. 240–243, Aug. 1965, doi: 10.1103/PhysRevLett.15.240.

[7] "Symplectic integrator," *Wikipedia*, Feb. 28, 2024. Accessed: Mar. 02, 2024. [Online]. Available: `https://en.wikipedia.org/w/index.php?title=Symplectic_integrator&oldid=1210885818`.

[8] H. Yoshida, "Construction of higher order symplectic integrators," *Physics Letters A*, vol. 150, no. 5, pp. 262–268, Nov. 1990, doi: 10.1016/0375-9601(90)90092-3.

[9] E. W. Weisstein, "Runge-Kutta Method." Accessed: Mar. 02, 2024. [Online]. Available: `https://mathworld.wolfram.com/Runge-KuttaMethod.html`

[10] NumPy — *NumPy*. Accessed: April 27, 2024. [Online]. Available: https://numpy.org/.

[11] Matplotlib — Visualization with Python, Accessed: April 27, 2024, [Online]. Available: https://matplotlib.org/.

[12] "Multidimensional image processing (scipy.ndimage) — SciPy v1.12.0 Manual." Accessed: Mar. 03, 2024. [Online]. Available: https://docs.scipy.org/doc/scipy/reference/ndimage.html

[13] Python documentation: *time — Time access and conversions*. Accessed: April 27, 2024. [Online]. Available: https://docs.python.org/3/library/time.html.

[14] M. Waskom, "seaborn: statistical data visualization," *Journal of Open Source Software*, vol. 6, no. 60, p. 3021, April 2021, doi: 10.21105/joss.03021.

[15] shivam bhatele, "Vectorization in Python- An Alternative to Python Loops," *The Pythoneers*. Accessed: Mar. 02, 2024. [Online]. Available: https://medium.com/pythoneers/vectorization-in-python-an-alternative-to-python-loops-2728d6d7cd3e

[16] G. Elert, "4.3 Lyapunov Exponent," in *The Chaos Hypertextbook*, hypertextbook, 1997. Accessed: Mar. 02, 2024. [Online]. Available: https://hypertextbook.com/chaos/lyapunov-1/

[17] "Norm (mathematics)," *Wikipedia*, Mar. 05, 2024. Accessed: Mar. 10, 2024. [Online]. Available: https://en.wikipedia.org/w/index.php