Project work 1: Harmonic oscillator

Markus Rauhalahti, Alex Odiyo, and Juho Karhu

1 Introduction

Dynamics of a harmonic oscillator was modelled numerically using the leapfrog algorithm. The program was implemented in Python 3.5. The effect of varying the timestep and spring constant was studied. The simulation results were compared to those predicted by the analytical solution and they were shown to be physically meaningful, as they fulfilled the conservation laws of energy and momentum.

2 Methods

2.1 Theory

Two particles interarction through a harmonic potential were simulated. The harmonic potential is

$$U = \frac{1}{2}k(r - r_e)^2, \tag{1}$$

where r is the distance between the two particles, r_e is the equilibrium distance and k is the spring constant. The form of the potential energy differs from the one given in instructions as the factor $\frac{1}{2}$ is not included in the spring constant. The analytical solution for the position of the particle is

$$r - r_e = r_0 \cos \omega t + \frac{v_0}{\omega} \cos \omega t \tag{2}$$

where r_0 is the initial deviation from the equilibrium position and v_0 is the initial relative velocity between the particles. The frequency ω is given by:

$$\omega = \sqrt{\frac{k}{\mu}} \tag{3}$$

where μ is the reduced mass of the particles. In this work we set the individual particle masses to unity for simplicity, from which we get

$$\omega = \sqrt{2k} \tag{4}$$

The force due to the potential is

$$F(r) = -\nabla U = -k(r - r_e) \tag{5}$$

The kinetic energy of the system is

$$T = \frac{1}{2}\sum_j v_j^2 \tag{6}$$

where the sum runs over the particles and v_j is the velocity of each particle.

The numerical solution was propagated using a leapfrog algorithm, where positions and velocities of the particles were updated consecutively in discrete steps. The equations iterated are

$$x_i = x_{i-1} + v_{i-1/2}\Delta t, \tag{7}$$
$$a_i = F(x_i) \tag{8}$$
$$v_{i+1/2} = v_{i-1/2} + a_i \Delta t, \tag{9}$$

from which we obtain the working equations by inserting the force from eq. 5

$$x_i = x_{i-1} + v_{i-1/2}\,\Delta t, \tag{10}$$
$$v_{i+1/2} = v_{i-1/2} - k(x_i - x_{eq})\,\Delta t, \tag{11}$$

2.2 Implementation

The program was written using python 3.5 and given in `molmod-p1.py`. Libraries numpy, scipy and matplotlib were used for linear algebra calculations and plotting.

The program takes as input parameters the initial positions and velocities of the two particles, the equilibrium distance, the spring constant, time step and the total simulation time.

Initial positions and velocities are given as 2×dim arrays, where dim is the dimension of the vector. The program was tested in R^3.

The final program does not work for one dimensional particles as some of the linear algebra calculations fail. As this would lead to major refactoring, the dynamics of a 26 dimensional object, perhaps from bosonic string theory, is showed here for compensation in figure 1.

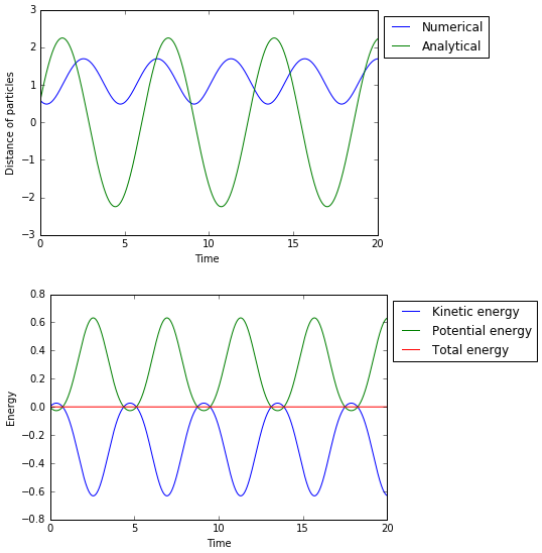


Figure 1: Results for 26 dimensional particles. Up: The deviation of the distance between the particles from the equilibrium distance, calculated both numerically and with the analytical formula. Below: The change in potential, kinetic and total energies, relative to their starting values.

The leapfrog procedure is iterated for $t/\Delta t$ steps. At each step, the positions of both particles are first updated, then the velocities of the particles is updated using the new positions. Potential and kinetic energies are calculated at each step.

After simulation is finished, two plots are created. One plot shows the difference of the particle distance and their equilibrium distance, and one showing how the system's total, potential and kinetic energy change during the simulation. The distance plot is also compared to an analytical form of the harmonic motion.

Standard deviation of the difference in the particle distance between numerical and analytical solutions, and for the conservation of the energy were computed and used in quantifying the accuracy of the simulation.

3 Results

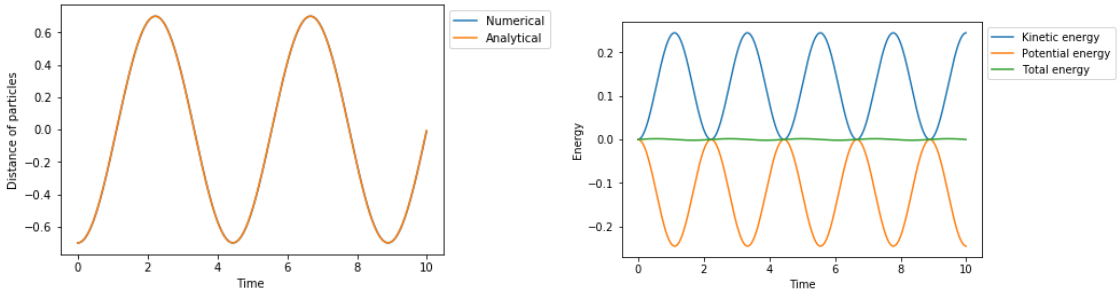
3.1 Effect of time step and spring constant

In order to determine the optimal time step the simulation was ran using time steps ranging from 0.001 to 1. In these simulations starting distance was set to near the equilibrium position and a small low spring constant was used.

We keep tabs on the output in relation to to the exact solution(analytical solution). The time step of 0.01 produced results that concurred with the analytical solutions over a wide range of parameters. Below is the output from one such test run:

```
Harmonic oscillator with the leapfrog algorithm.
Parameters:
– Equilibrium distance: 2
– Spring constant: 1
– Timestep: 0.01
– Simulation time: 10
Standard deviation between numerical and analytical solution: 3.45e−03
Standard deviation for total energy: 1.22e−03
```

The energies and positions as a function of time using these parameters are shown in figure 2. The numerical and the analytical solutions and the standard deviation is less than 1% of the amplitude of the oscillation and the plots are equivalent. Similarly one can observe the conservation of the energy. The spring constant was varied using the timestep of 0.001. Reasonable results were



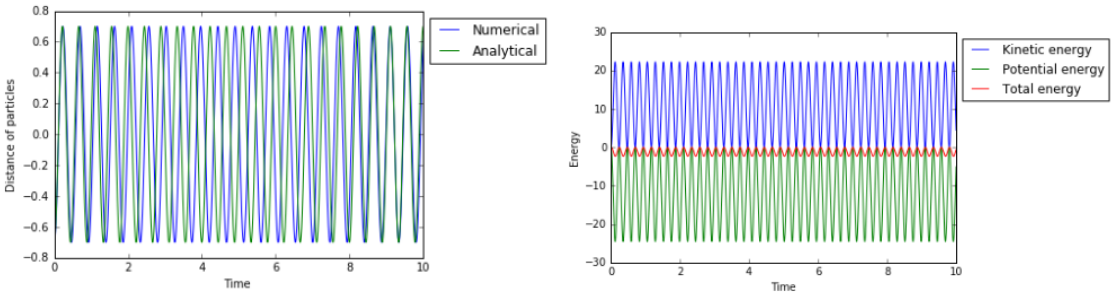
(a) (b)

Figure 2: a) The deviation of the distance between the particles from the equilibrium distance, calculated both numerically and with the analytical formula. b) The change in potential, kinetic and total energies, relative to their starting values.

obtained with values of the order $k = 10^2$. With larger values the numerical results shift out of phase relative to the analytical solution and the total energy starts to oscillate. These are shown in figure 3.

3.2 Visualizing the results

The visualization was primarily done using VMD (file `vmd2video.mpg`) and later compared with Ovito (file `ovito2.avi`). The simulation time step (dt) was chosen to be 0.01 and the total time 100. The data was saved in an .xyz file format which could easily be read by the visualization software.



(a) (b)

Figure 3: a) The analytical and numerical results for the distance and b) the energy using a spring constant of 100.

The procedure involved loading the data in VMD. Under the graphics tab and representation options a few modification are done to obtain nice spheres, and manipulate the size, color and geometry. The option VDW and index, makes visualization easy. The last section was to create frames for movie files. Under the extensions tab and visualize and movie maker options frames were generated. Frames for which all the particles were totally out of the simulation region were excluded in making of the movie for trivial reasons such as redundancy of information and control of movie file size.

The visualization movie shows a spring like motion of two particles. In fast video motion, the actual motion can be easily be seen in which one particle moves the bond elongates and subsequent action of the restoration force is observe to be pulling the other particle. second observation is that the particles stretch and attempt to restore their initial position, the particles come out of the simulation box range. This is due unrestricted boundary condition that have been imposed on the system.

3.3 Conservation of linear and angular momentum

In the simulation runs that were animated, one of the particles was given an initial velocity in a direction perpedicular to the molecular axis. The animation shows the particle system moving to the same direction with the initial velocity over the whole simulation, as is expected due to the conservation of momentum. Since the initial velocity is perpendicular to the acceleration caused by the harmonic force, it induces a rotation in the two-particle system. The rotation continues over the whole simulation, demonstrating the conservation of angular momentum.