Computational Physics - Project 4

Johannes Scheller, Vincent Noculak, Lukas Powalla, Richard Asbah

December 2, 2015

Contents

1 Introduction

In this project we are going to simulate open clusters using Newton's law of universal gravitation. We are going to look at the time an open cluster with a cold collapse would need to collapse into singularity and observe the energy conservation and distribution of potential and kinetic energy. On what properties does it depend how many particles get ejected from the cluster and how do those particles change the energy of the system? The equilibrium state and its radial partial density are going to be observed too.

We are doing these calculation by using the Velocity Verlet algorithm and the fourth order Runge-Kutta method. First we are going to compare both algorithms by simulating a to body system. Then we are going to study, which algorithm is more suitable for doing open cluster simulations and continue to simulate the open cluster with the more fitting algorithm.

2 Theory

2.1 Newton's law of universal gravitation

Newton's law of universal gravitation states that every point mass attracts every single other point mass by a force pointing along the line intersecting both points. The force two points of mass with the masses m_1 and m_2 and a distance of r to each other attract each other is given by

$$\mathbf{F_G} = G\frac{m_1 \cdot m_2}{r^3} \cdot \mathbf{r} \tag{1}$$

Where G is the gravitational constant. In our project we have a system with n stars which get approximated as points of mass. Hence, to get the gravitational force acting on one star, we just need to sum over all $\mathbf{F}_G$'s (having the form of equation (1)) that are acting on this star due to the other stars.

$$\mathbf{F}_{Gi} = \sum_{j\neq i}^{n} G\frac{m_i \cdot m_j}{r_{ij}^3} \cdot \mathbf{r}_{ij} = Gm_i \sum_{j\neq i}^{n} m_j \frac{\mathbf{r}_{ij}}{r_{ij}^3} \tag{2}$$

Then the acceleration of star i is given by

$$\mathbf{a} = G\sum_{j\neq i}^{n} m_j \frac{\mathbf{r}_{ij}}{r_{ij}^3} \tag{3}$$

2.2 Open Clusters

Open clusters are groups of a few thousand stars that were formed from the same giant molecular cloud and have roughly the same age. They have yet only been found in spiral and irregular galaxies, in which active star formation is occurring. Many open clusters ate unstable and have a small enough mass that the escape velocity of the system is lower than the average velocity of the constituent stars. These clusters will rapidly disperse within a few million years[1].

The open cluster we simulate in this project will have the starting conditions of a cold collapse. This means, that the particles start with little or no initial velocity. In addition to the that, the N particles, we simulate, will be uniformly distributed at the starting time inside a sphere with given radius R_0. Our particles masses will be distributed by a Gaussian distribution around ten solar masses with a standard deviation of one solar mass.

2.3 Verlet algorithm

The Verlet algorithm is one method that is important for this project to numerically solve the equation of motion having the form of equation (2).

Using a taylor expansion for a function $x(t)$ around the point $t_0 = t_i$ and then setting $t = t_i + h$ and $t = t_i - h$, we get the equations

$$x(t_i + h) = x_i + hx^{'}(t_i) + \frac{h^2}{2}x^{''}(t_i) + \frac{h^3}{3!}x^{'''}(t_i) + \sigma(h^4) \tag{4}$$

$$x(t_i - h) = x_i - hx^{'}(t_i) + \frac{h^2}{2}x^{''}(t_i) - \frac{h^3}{3!}x^{'''}(t_i) + \sigma(h^4) \tag{5}$$

For an easier overview it is useful to use the notation $x(t_i) = x_i$, $x(t_i \pm h) = x_{i\pm1}$ and $x^{''} = a(x,t)$. By adding up equation equation (4) and (5) we come to the equation

[1]Source: `https://en.wikipedia.org/wiki/Open_cluster`

$$x_{i+1} = 2x_i - x_{i-1} + a(x_i, t_i) \cdot h^2 + \sigma(h^4) \tag{6}$$

Which gives us a method to determine all x_i, if the boundary conditions are known. It can be seen that due to the x_{i-1} part, the algorithm is not self-starting. Hence x_1 must be found, using an other method (for example Euler's method).

2.3.1 Velocity Verlet algorithm

The Velocity Verlet is similar to the Verlet algorithm. Here we use the Taylor expansion with $t_0 = t_i$ and $t = t_i + h$ to get the formula:

$$x_{i+1} = x_i + h \cdot v_i + \frac{h^2}{2} a_i(x_i, t_i) + \sigma(h^3) \tag{7}$$

With $v(t) = \frac{dx}{dt}$. And we use the Trapezoidal rule to get the formula:

$$v_{i+1} = v_i + \frac{h}{2} \cdot (a(x_i, t_i) + a(x_{i+1}, t_{i+1})) + \sigma(h^2) \tag{8}$$

The algorithm starts with calculating equation (7) for $i = 0$ to get x_{i+1}. With this value, $a(x_i, t_i)$ and $a(x_{i+1}, t_{i+1})$ can be calculated. Then it is possible to calculate v_{i+1} with equation (8). These steps can now be repeated for $i = i + 1$, allowing to get the values for all x_i and v_i.

2.4 Runge-Kutta methods

Runge-Kutta methods use the following basic equation of integration to numerically approximate Ordinary differential equations.

$$x_{i+1} = x_i + \int_{t_i}^{t_{i+1}} dt' \, v(x, t') \tag{9}$$

With $x(t_i) = x_i$, $x(t_i \pm h) = x_{i\pm1}$ and $\frac{dx}{dt} = v$. The integral can be approximated using numerically integration methods, as for example the trapezoidal rule or Simpson's rule. In Runge-Kutta methods to calculate a step x_{i+1}, an intermediate step between x_i and x_{i+1} gets used.

2.4.1 fourth order Runge-Kutta method (RK4)

In this project we are going to use the fourth order Runge-Kutta method. This method solves the integral in equation (9) with help of simpson's method. By applying this method we get

$$\int_{t_i}^{t_{i+1}} dt' \, v(t', x) = \frac{h}{6}[v(t_i, x_i) + 4v(t_{i+\frac{1}{2}}, x_{i+\frac{1}{2}}) + v(t_{i+1}, x_{i+1})] + \sigma(h^5) \tag{10}$$

By setting (10) in (9):

$$x_{i+1} = x_i + \frac{h}{6}[v(t_i, x_i) + 2v(t_{i+\frac{1}{2}}, x_{i+\frac{1}{2}}) + 2v(t_{i+\frac{1}{2}}, x_{i+\frac{1}{2}}) + v(t_{i+1}, x_{i+1})] + \sigma(h^5) \tag{11}$$

$x_{i+\frac{1}{2}}$ and x_{i+1} are unknown values. Now we define $k_1 = v(t_i, x_i)$, first approximate $x_{i+\frac{1}{2}}$ with Euler's method

$$x_{i+\frac{1}{2}} \approx x_i + \frac{h}{2} v(t_i, x_i) = x_i + \frac{h}{2} k_1 \tag{12}$$

and define

$$k_2 = v(t_{i+\frac{1}{2}}, x_i + \frac{h}{2} k_1) \tag{13}$$

Next we calculate $x_{i+\frac{1}{2}}$ with Eulers's method using k_2 instead of k_1 and then define

$$k_3 = v(t_{i+\frac{1}{2}}, x_i + \frac{h}{2} k_2) \tag{14}$$

We predict x_{i+1} with k_3 using Euler's method

$$x_{i+1} = x_i + hk_3 \tag{15}$$
$$k_4 = v(t_{i+1}, x_i + hk_3) \tag{16}$$

Now we can calculate our final x_{i+1} by inserting our k_j's in equation (11).

$$x_{i+1} = x_i + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) + \sigma(h^5) \tag{17}$$

2.4.2 RK4 in this Project

In section 2.4.1 it could be seen, that with the fourth order Runge-Kutta method, first-order ODE's (ordinary differential equations) can be solved. In this project however, we need to numerically solve the equation of motion for multiple particles, which is a second-order ODE (see equation (3)). This ODE can be split up into two first-order ODE's by writing

$$\frac{d\mathbf{r}}{dt} = \mathbf{v} \tag{18}$$

$$\frac{d\mathbf{v}}{dt} = G\sum_{j\neq i}^{n} m_j \frac{\mathbf{r}_{ij}}{r_{ij}^3} \tag{19}$$

Now we can use the RK4 method $\mathbf{r}$ and $\mathbf{v}$ simultaneously. Because we use this method in three dimensions and the fact, that we have a second-order ODE, the equations for the RK4 method look different than in section 2.4.1. The following steps give in the right order, how to approach the system, when we do a numerically calculation from the time t_i to t_{i+1}. It has to be noted, that when we do a calculation for one particle, directly after it, the same time the calculation gets performed for the other particles. The index on the upper left side refer to the number of the particle and the index on the upper right side gives the information, if the value is belogs to the RK4 method for the location or the velocity.

$$\mathbf{k}_1^r = \mathbf{v}_i \tag{20}$$

$$\mathbf{k}_1^v = \mathbf{a}({}^1\mathbf{r}_i, {}^2\mathbf{r}_i, ..., {}^N\mathbf{r}_i) \tag{21}$$

$$\mathbf{k}_2^r = \mathbf{v}_i + \frac{h}{2}\mathbf{k}_1^v \tag{22}$$

$$\mathbf{k}_2^v = \mathbf{a}({}^1\mathbf{r}_i + \frac{h}{2}\cdot{}^1\mathbf{k}_1^r, {}^2\mathbf{r}_i + \frac{h}{2}\cdot{}^2\mathbf{k}_1^r, ..., {}^N\mathbf{r}_i + \frac{h}{2}\cdot{}^N\mathbf{k}_1^r) \tag{23}$$

$$\mathbf{k}_3^r = \mathbf{v}_i + \frac{h}{2}\mathbf{k}_2^v \tag{24}$$

$$\mathbf{k}_3^v = \mathbf{a}({}^1\mathbf{r}_i + \frac{h}{2}\cdot{}^1\mathbf{k}_2^r, {}^2\mathbf{r}_i + \frac{h}{2}\cdot{}^2\mathbf{k}_2^r, ..., {}^N\mathbf{r}_i + \frac{h}{2}\cdot{}^N\mathbf{k}_2^r) \tag{25}$$

$$\mathbf{k}_4^r = \mathbf{v}_i + h\cdot\mathbf{k}_3^v \tag{26}$$

$$\mathbf{k}_4^v = \mathbf{a}({}^1\mathbf{r}_i + h\cdot{}^1\mathbf{k}_3^r, {}^2\mathbf{r}_i + h\cdot{}^2\mathbf{k}_3^r, ..., {}^N\mathbf{r}_i + h\cdot{}^N\mathbf{k}_3^r) \tag{27}$$

The accelerations can always get calculated by using equation (3). After the steps of equations (20) to (27) are executed, the velocity and positions of the particles for the time t_{i+1} can be calculated using formula (17).