Computational Physics Assignment 1: N-Body Simulation

R. Abele[a] and N. Matera[a]

[a]*Eberhard Karls University of Tuebingen*

Abstract

The following exercises showcase several integrators to solve the n-body problem. We were given initial conditions in the form of particle positions and velocities.

1. Background

The following exercises showcase the implementation of several integrators to simulate the n-body problem. For simplicity, the total mass M as well as the gravitational constant G are set to 1. The coordinates and velocities of the particles are normalized so that the simulation is entered and does not drift over time. We were provided with initial conditions in the form of masses, positions, and velocities for $n = 2, 3, 100, 1000$ 3-dimensional particles.

2. The Program

The first part of out program, found under `01-nbody/code/data_normalization` directory, finds the center of mass and calculates the velocity of the center of mass. This information is then used to shift all of the positions and velocities of the particles so that the system does not drift over time.

*

The main part of the assignment, the actual simulation of planetary movement, can be found in `01-nbody/code/simulation`. This simulation works by first calculating the attractive forces between all of the objects and then determining by how much every particle should be moved in a given time step considering considering the particle's mass and current velocity. The velocities of the particles are also updated accordingly and the cycle repeats for the next time step.

Extrapolation of future particle positions and velocities using the known information can be performed with varying degrees of accuracy, varying not only the size of the time step, but also the integrator used.

Integrators, as the name suggests integrate the changes in particle position over time and can be used to predict a future position. There are many kinds of integrators, but the ones implemented in this assignment are the *Euler, Euler-Cromer, velocity-Verlet, Heun, Hermite, iterated Hermite, Heun,* and *Runge-Kutta 4* integrators.

Implementing most of the integrators involves directly following the directions presented in the manual, but a small calculation involving the jerk and high-order acceleration derivatives was required to determine the final form of the Hermite integrator. From the manual, we know that:

$$\frac{1}{2}a_n^{(2)} = -3\frac{a_n - a_{n+1}^p}{\Delta t^2} - \frac{2\dot{a}_n + \dot{a}_{n+1}^p}{\Delta t}, \tag{1a}$$

$$\frac{1}{6}a_n^{(3)} = 2\frac{a_n - a_{n+1}^p}{\Delta t^3} + \frac{\dot{a}_n + \dot{a}_{n+1}^p}{\Delta t^2}. \tag{1b}$$

Furthermore, we are given that the *corrected* positions and velocities calculated by the Hermite integrator can be written as follows:

$$v_{n+1}^c = v_{n+1}^p + \frac{1}{6}a_n^{(2)}\Delta t^3 + \frac{1}{24}a_n^{(3)}\Delta t^4 \tag{2a}$$

$$r_{n+1}^c = r_{n+1}^p + \frac{1}{24}a_n^{(2)}\Delta t^4 + \frac{1}{120}a_n^{(3)}\Delta t^5 \tag{2b}$$

Some rearranging and substitution results in the following for the corrected velocity and position:

$$v_{n+1}^c = v_{n+1}^p + \frac{1}{12}\left(-6a_n\Delta t + 6a_{n+1}^p\Delta t - 5\dot{a}_n\Delta t^2 - \dot{a}_{n+1}^p\Delta t^2\right) \tag{3a}$$

$$r_{n+1}^c = r_{n+1}^p + \frac{1}{60}\left(-9a_n\Delta t^2 + 9a_{n+1}^p\Delta t^2 - 7\dot{a}_n\Delta t^3 - 2\dot{a}_{n+1}^p\Delta t^3\right). \tag{3b}$$

A modified version of the Hermite integrator, known as the iterated Hermite integrator can be achieved by first calculating the corrected velocity and using it to calculated the predicted position, creating a set of equations with the following form:

$$v_{n+1}^c = v_n + \frac{1}{2}\left(a_{n+1}^p + a_n\right)\Delta t + \frac{1}{12}\left(\dot{a}_{n+1}^p - \dot{a}_n\right)\Delta t^2 \tag{4a}$$

$$r_{n+1}^c = r_n + \frac{1}{2}\left(v_{n+1}^c + v_n\right)\Delta t + \frac{1}{12}\left(a_{n+1}^p - a_n\right)\Delta t^2 \tag{4b}$$