Worksheet 1: Integrators

Patrick Kreissl und Sebastian Weber

November 14, 2012
University of Stuttgart

Contents

1 Cannonball

A program was written to simulate the flight of a cannonball thrown with the velocity $\vec{v} = (50, 50)^T \frac{m}{s}$ in two dimensions [1]. Therefore the Euler integration scheme was used.

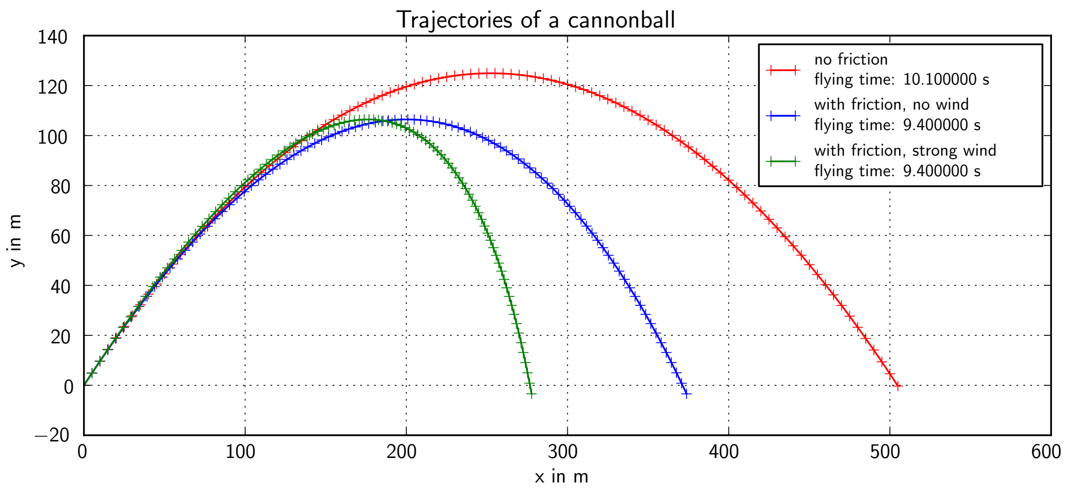

Figure 1: Trajectories

As a result we get a parabolic trajectory (see figure 1) if we neglect fricition. This meets one's expectations. The question what happens with friction is much more interesting because of the lack of a analytic solution. It's also possible to examine the influence of wind.

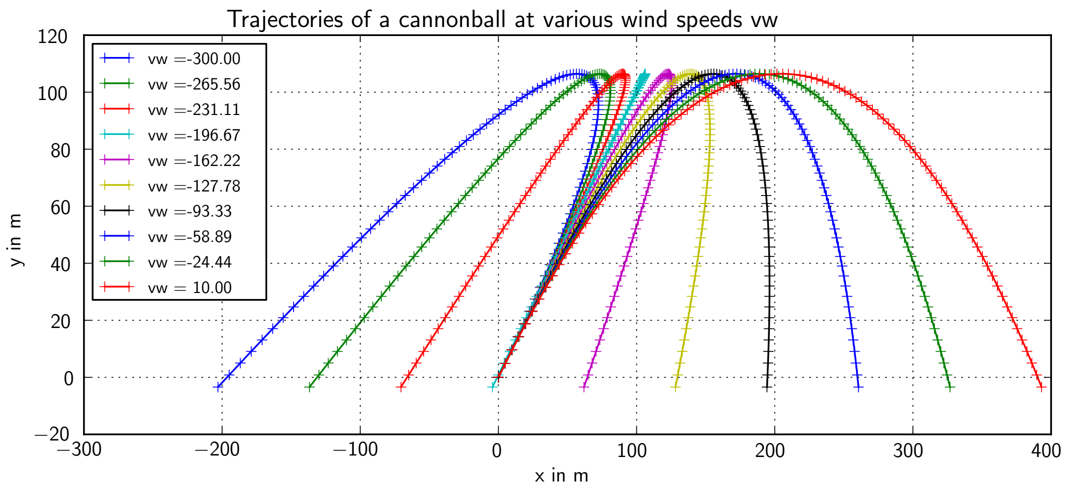

Figure 2: Influence of different wind speeds

As you can see in figure 2 the cannonball comes back close to his initial position if the wind speed is about $\vec{v}_W = (0, -196.67)^T \frac{m}{s}$.

[1]for the code please take a look at the appendix

2 Theoretical part

2.1 Derivation of the Velocity Verlet alogorithm

Let's start with a Taylor expansion of $x(t+\Delta t)$ and $v(t+\Delta t)$.

$$x(t+\Delta t) = x(t) + \dot{x}(t)\Delta t + \frac{1}{2}\ddot{x}(t)\Delta t^2 + \mathcal{O}(\Delta t^3) \tag{1}$$

$$= x(t) + v(t)\Delta t + \frac{1}{2}a(t)\Delta t^2 + \mathcal{O}(\Delta t^3) \tag{2}$$

$$v(t+\Delta t) = v(t) + a(t)\Delta t + \frac{1}{2}\dot{a}(t)\Delta t^2 + \mathcal{O}(\Delta t^3) \tag{3}$$

We get rid of the derivative of $a(t)$ by the use of Taylor's formular once more.

$$\dot{a}(t)\Delta t = a(t+\Delta t) - a(t) + O(\Delta t^2) \tag{4}$$

Considering $\Delta t \cdot O(\Delta t^2) = O(\Delta t^3)$ and putting everything together we obtain the Velocity Verlet alogorithm.

$$x(t+\Delta t) = x(t) + v(t)\Delta t + \frac{a(t)}{2}\Delta t^2 + \mathcal{O}(\Delta t^3) \tag{5}$$

$$v(t+\Delta t) = v(t) + \frac{a(t) + a(t+\Delta t)}{2}\Delta t + \mathcal{O}(\Delta t^3) \tag{6}$$

2.2 Velocity Verlet is equivalent to the standard Verlet algorithm

We can express x(t) by rearranging equation 5.

$$x(t) = x(t+\Delta t) - v(t)\Delta t - \frac{a(t)}{2}\Delta t^2 + \mathcal{O}(\Delta t^3) \tag{7}$$

Also with equation 5, we can calculate $x(t+2\Delta t)$ by substitution of t with $t+\Delta t$. Then you have to add the received equation to equation 7.

$$x(t+2\Delta t) + x(t) = 2x(t+\Delta t)\left(v(t+\Delta t) - v(t)\right)\Delta t + \frac{a(t+\Delta t) - a(t)}{2}\Delta t^2 + \mathcal{O}(\Delta t^3) \tag{8}$$

We use equation 6 to get an expression for $\left(v(t+\Delta t) - v(t)\right)\Delta t$.

$$\left(v(t+\Delta t) - v(t)\right)\Delta t = \frac{a(t) + a(t+\Delta t)}{2}\Delta t^2 + \mathcal{O}(\Delta t^4) \tag{9}$$

Plugging in this expression leads to the standard Verlet algorithm.

$$x(t+\Delta t) = 2 \cdot x(t) - x(t-\Delta t) + a(t) \cdot \Delta t^2 + \mathcal{O}(\Delta t^3) \tag{10}$$

2.3 Difficulties with the standard Verlet algorithm

The standard Verlet algorithm needs the two predecessors $x(t - \Delta t)$ and $x(t)$ to calculate $x(t + \Delta t)$. However the initial conditions give us only one starting value for x. Therefore we have to calculate the missing predecessors via Taylor's formula at the very first step. This results in a slightly bigger error than $\mathcal{O}(\Delta t^4)$.

3 Advanced integrators: Solar system

3.1 Simulating the solar system with the Euler scheme

A program was written to simulate a part of the solar system with given initial position `x_init` and velocity `v_init` by only taking account of Newton's law of universal gravitation. For this simulation again the Euler integration scheme was used.

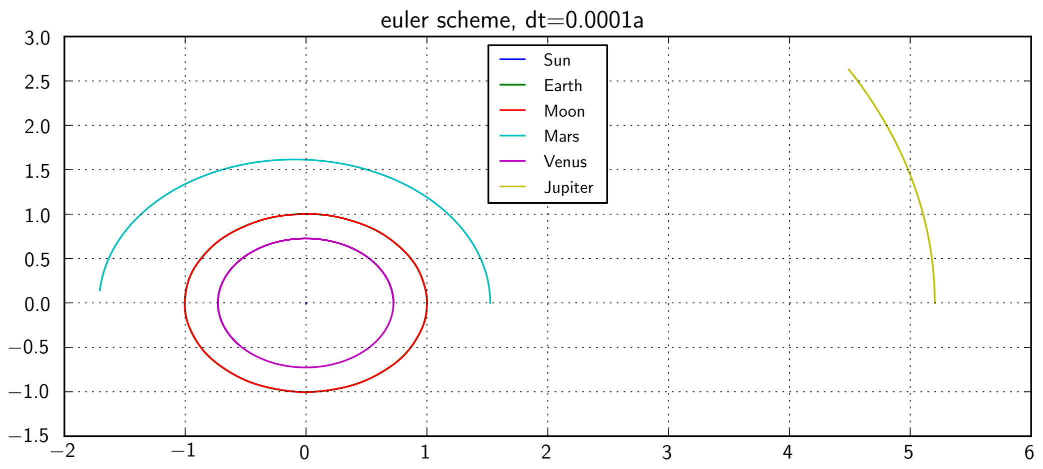

Figure 3: Simulation of a part of the solar system using the Euler integration scheme

The simulated trajectories are as expected nearly ellical (see figure 3). Now we take a closer look on the trajectory of the moon in the rest frame of the earth for different time steps `dt`.Therefor we run the simulation of the solar system for the time steps `dt`$= 0.0001$, `dt`$= 0.001$ and `dt`$= 0.01$. Then for all simulations the difference of the location of the moon an the location of the earth are calculated in order to get the trajectory of the moon relative to the earth. The resulting plot is shown in figure 4.

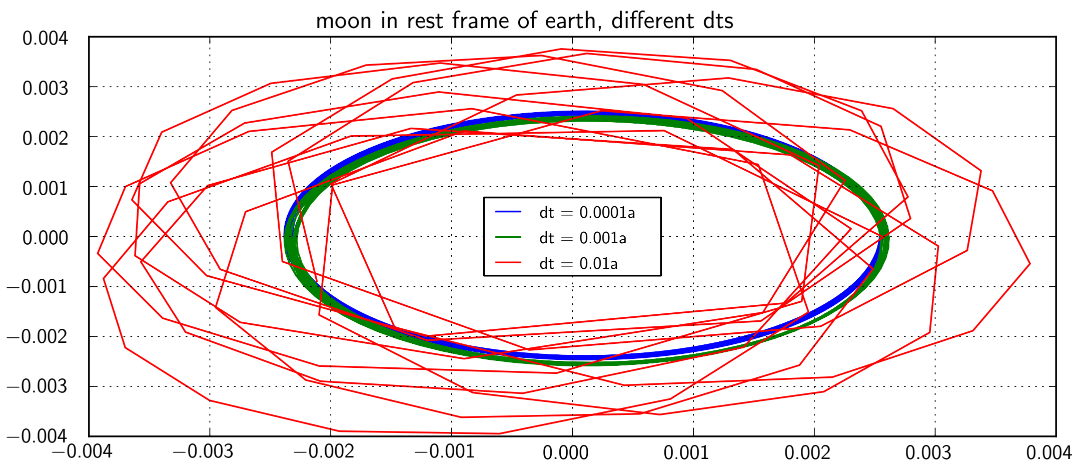

Figure 4: Simulated trajectory of the moon in the rest frame of the earth using different time steps

As expected it seems that the tinier the chosen time step is the more precise the simulated trajectory gets. For the time steps `dt1`= 0.0001 and `dt`= 0.001 the moons simulated trajectory relative to the earth is a ellipse as also observed in reality and therefore satisfactory.
In this small simulation of only six particles for each one of them the forces of all the others have to be computed every time step. (In this simulation that makes 15 forces per time step.) Therefor this part of the whole simulation is the most time-consuming.
For modern simulations with up to a few billion particles computing the forces takes an immense amount of the whole computing time.

3.2 Integrators

The simulation of the moons trajectory used the Euler integration scheme. Now this scheme is compared to the symplectic Euler scheme and the Velocity Verlet algorithm (see part 2) by computing the solar system using the different algorithms and comparing the plots of the trajectory of the moon. The result is shown in figure 5.

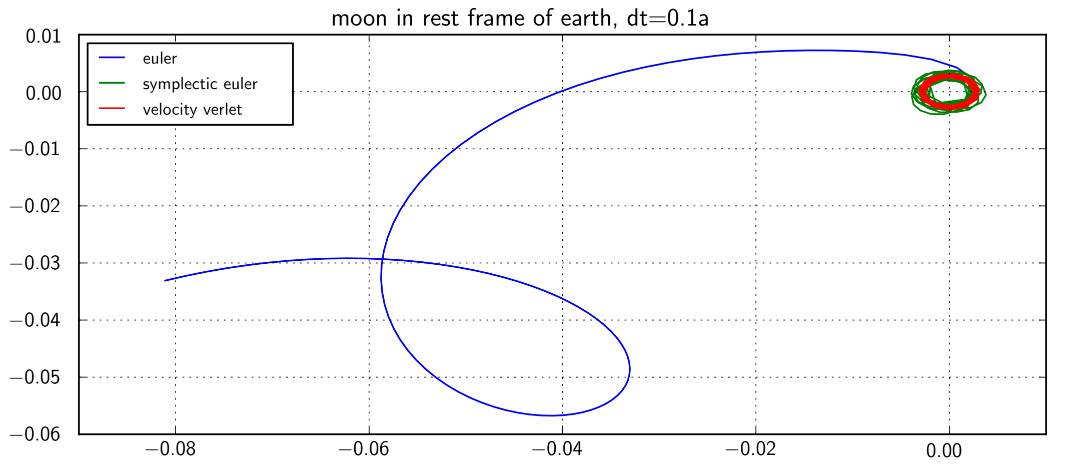

Figure 5: Simulated trajectory of the moon in the rest frame of the earth using different algorithms

Obviously the simple Euler integration scheme ist the poorest of the three compared algorithms. After a short time it diverges from the expected trajectory. In contrast the trajectories complusted by the symplectic Euler and the Velocity Verlet algorithm follow more or less the expected ellipse. For this simulation the Velocity Verlet seems to be the best of the three algorithms.

3.3 Long-term stability

This program is a simple modification of the one used in the part before. The only difference is that the simulated time is increased from one to ten years.

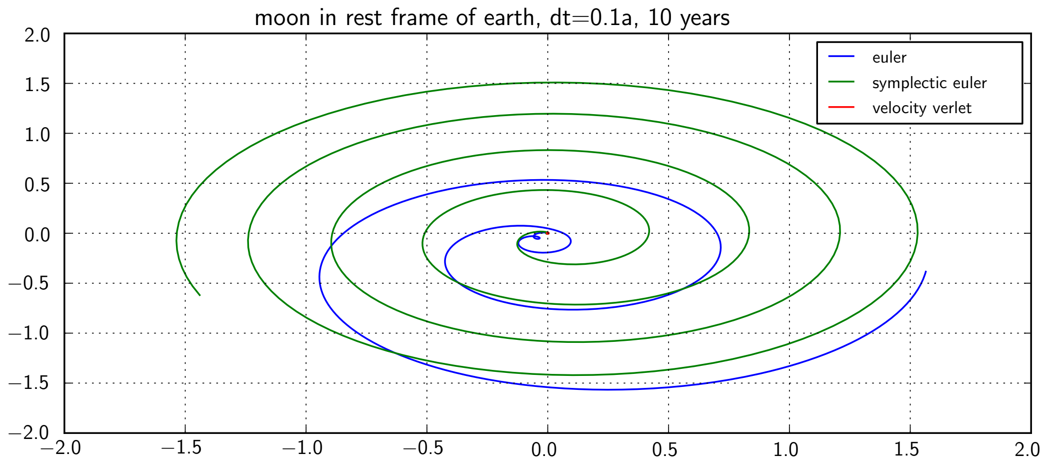

Figure 6: Long-term simulation of the trajectory of the moon using different algorithms

In the resulting plot (see figure 6) the Velocity Verlet algorithm computed trajektory can hardly be seen as litte red point in the center. This confirms the result of the previous simulation: Velocity Verlet seems not only to be short-term but also long-term the best of the compared algorithms.

Surprisingly in the long-term simulation the symplectic Euler algorithm diverges faster than the simple Euler integration algorithm.