Homework 4

Mats Thijssen

October 21, 2017

1 Introduction

In this exercise, a variety of different ODE-solvers (Ordinary Differential Equations) are programmed and applied to various orbiting bodies. Their efficiency and usefulness is compared in terms of error and energy conservation.

2 Theory

Ordinary Differential Equations are differential equations of one independent variable and it's derivative(s). For non-linear ODEs, exact solutions are rare, but there exists various schemes to solve them numerically. The equation of interest concerning orbits is given by Newtons law of gravitation, in the form of:

$$\frac{d^2r}{dt^2} = -\frac{GM}{r^2}$$

where r, the radius, is our independent variable. To solve this equation numerically, we convert into Cartesian coordinates, and for each coordinate convert it to 2 coupled 1st order equations:

$$\frac{dx}{dt} = v_x \tag{1}$$
$$\frac{dv_x}{dt} = -\frac{GMx}{r^3} \tag{2}$$
$$\frac{dy}{dt} = v_y \tag{3}$$
$$\frac{dv_y}{dt} = -\frac{GMy}{r^3} \tag{4}$$

These equations will be solved using 4 distinct methods: Forward-Euler, Rk2 (Runge Kutta, order 2), Rk4 and Verlet. It is specifically applied to the orbit of Mars, Halley's comet and S02 orbiting Sagittarius A*. Note that for Mars and Halley, a system of units will be used where distance is measured in units of perihelion distance and time in Earth years/Halley Orbits respectively.

2.1 Euler

Forward-Euler is the simplest of our schemes and is the result of Taylor expanding x(t) for an equation of form $\frac{dx}{dt} = f(x,t)$. This results in:

$$x(t+h) = x(t) + h\frac{dx}{dt} = x(t) + hf(x,t)$$

neglecting terms of 2nd order and higher. The time step is given by h, as it will throughout this report. This method is applied straight-forwardly to equation (1)-(4) above. Results are shown further on in this report. One important thing to note is that Forward-Euler is not time-symmetric for $h \to -h$, meaning this scheme does not conserve energy.

2.2 Rk2

2nd Order Runge-Kutta, also known as the Midpoint Method, is an extension of Euler's method. Here, instead of using the slope at the initial point, we use the slope of the point in the middle of the initial point and point after one time-step. This requires some extra computational effort in terms of finding the midpoint, but in return is accurate to 2nd Order. We can see this by Taylor expanding around t+h/2 and getting x(t) and x(t+h). Subtract one from another and end up with:

$$x(t+h) = x(t) + hf(x(t+h/2), t+h/2) + O(h^3)$$

The 2nd order in h term cancels, and we are left with a scheme accurate to second order. To find x after half a time-step, however, we must invoke Euler, invoking a total/global error of order h^2.

2.3 Rk4

Rk4 is, like Rk2, a method based on the slopes at the midpoint, but in a manner that is a bit more convoluted than Rk2. The slope is found at both start,end, and midpoint before a weighted average is taken to evolve in time. The scheme can be laid out as follows:

$$\begin{aligned} k_1 &= hf(x,t) \\ k_2 &= hf(x + \frac{k_1}{2}, t + \frac{h}{2}) \\ k_3 &= hf(x + \frac{k_2}{2}, t + \frac{h}{2}) \\ k_4 &= hf(x + k_3, t + h) \\ x(t+h) &= x(t) + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k4) \end{aligned}$$

As can be seen, greater weight is given to the slopes at the midpoints. If f is independent of x, then Rk4 is equivalent to Simpsons rule. Rk4 is a fourth-order scheme, with total/global error of order h^4.

2.4 Verlet

The Velocity Verlet algorithm we apply here is similar to the leapfrog method, where we calculate the position at each time step and velocity at the half-steps in between, using the other observable to get the next, as such:

$$\begin{aligned} x(t+h) &= x(t) + h * v(t+h/2) \\ v(t+3h/2) &= v(t+h/2) + h * f(x(t+h), t+h) \end{aligned}$$

To find the first velocity needed (at $t_0 + h/2$), Euler is used. We also find the velocities at the "position-points", as we wish to do an energy calculation. This is done by introducing the following two steps:

$$\begin{aligned} k &= h * f(x(t+h), t+h) \\ v(t+h) &= v(t+h/2) + k/2) \end{aligned}$$

Note that this algorithm is time-symmetric ($h \to -h$) and thus conserves Energy, unlike the other schemes mentioned above. The total error in both position and velocity is of order h^2.

2.5 Adaptive Step Size

For computational efficiency, it is not always wise to integrate over a fixed step size. In a region of large oscillations, we wish to use a small step size to capture as much of the behaviour as wished, while in fairly steady regions, we can make do with a much larger step size while retaining the same error. To accomplish this, we keep track of our error estimate and dynamically change the step size to keep the error below some specified value. An adaptive step size is implemented into the Rk4-method by the following scheme:
Two integrations are done, from t to t+2h in two different ways: 2 steps of h, and one step of 2h. For Rk4, the error scales like h^5, and is thus ch^5 for some constant c per step. We call the estimate

by 2 steps x_1, and x_2 for one step. These should both be separated from the true solution by some error and thus we can write:

$$x_1 + 2ch^5 = x_2 + c(2h)^5$$

Which gives error $\epsilon = (1/30)(x_1 - x_2)$ per step. If we took a different step size h', its error in terms of the old step size would be $\epsilon' = (1/30)(x_1 - x_2)(h'/h)^5$. We can define a target accuracy pr time, σ, such that we can require $\sigma h' = \epsilon'$. Solving this for h' gives:

$$h' = h\rho^(1/4)$$

where ρ is given by:

$$\rho = \frac{h\sigma}{(x_1 - x_2)/30} = \frac{\epsilon_{target}}{\epsilon_{actual}}$$

Now, this can easily be implemented for computationally. For each step, we compute x_1 and x_2 and calculate ρ. If ρ>1, we have less error than our boundary, and can keep the step. while increasing h to h' given above. If the error is too high, we repeat the step with the new h'.
This scheme is implemented and tested for Halley's Comet and compared to Rk4 and Verlet for computational effort spent getting similar errors.

2.6 Exact Solution

To compare this methods, an exact solution is needed. Since we are dealing with orbits, we can solve Keplers Equation:

$$M = E - e\sin E$$

where M is the mean anomaly, a measure of the angular distance from the perihelion the planet would've had, had it been in a circular orbit. It is formally defined as:

$$M = \frac{2\pi}{T}(t - \tau)$$

where T is the period, and tau the time of pericenter passage.
E is the eccentric anomaly we wished to solve for, i.e. how much does the orbit deviate from a circle. From there we can get the coordinates by:

$$x = a(\cos E - e)$$

$$y = b\sin E$$

To solve Keplers equation, we gather all terms on one side and use a root finding method to solve for E. To get the exact solution, we use Newton-Rapheson, which uses the derivative at the guess-position to track back and get a better guess. For the above equation it would look like:

$$E' = E - \frac{E - e\sin E - M}{1 - \cos E}$$

where E is our first guess, and E' our new guess. There are other methods for finding roots, such as bisection, secant and relaxation. All of these are implemented and compared to Newton-Rapheson.

3 Results: Mars

The four integration methods discussed above were applied to the orbit of mars. The planet was allowed to orbit for 5 earth-years, and the resulting orbits can be seen below.

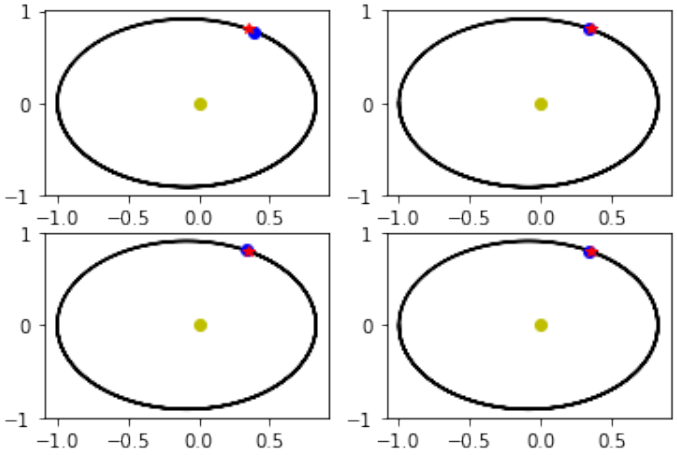

Figure 1: Orbit of Mars

The different schemes are, clockwise from top left; Euler, Rk2, Rk4, and Verlet. The blue dot represents the calculated position of Mars after 5 years, the red star the exact position. The yellow dot in the center represents the sun (not to scale). Note that the units are measured in perihelion distances.

3.1 Error

Since these are approximation methods to the ODEs, they all naturally have some associated error. To quantify this, the distance between the numerical and the analytical solution is calculated (after 5 years) for a varying amount of steps. Log-log plots of error vs. step size are shown below.

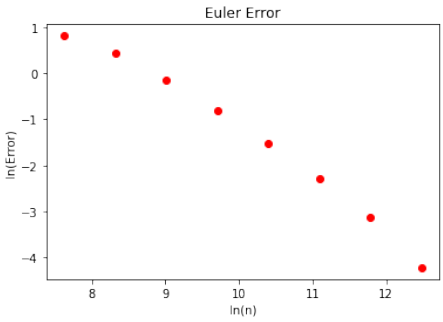

Figure 2: Error for Eulers Method

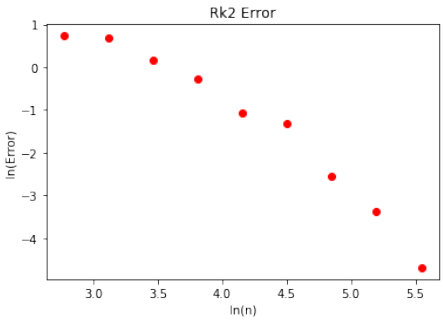

Figure 3: Error for Rk2

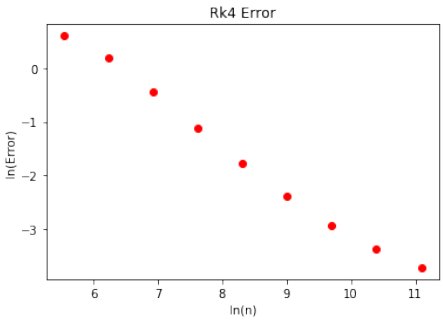

Figure 4: Error for Rk4

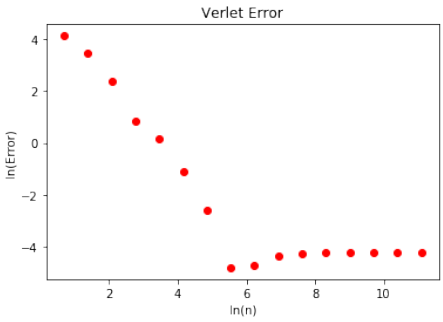

Figure 5: Error for Verlet

The slopes of the lines are, counterclockwise from top left, -0.9,-1.9,-0.8,-1.8. As expected, Euler converges at about order 1, and Rk2 at order 2. Rk4 inexplicably converges at a way slower rate than expected. There seems to be a bug in the program, but I have been unable to locate it. This

is surprising, given the nice-looking orbit seen in Fig.1. For Verlet, the error flattens out earlier than for the other schemes, seemingly hitting machine-precision. However, this seems surprisingly early, so I'm leaving the plot here. The slope, however, is only calculated for the part before the leveling. Note the different scalings for the number of steps for each method, as they all start approaching the correct answer for different steps. (i.e. order 9-10 for Euler, 4-5 Rk2, 7-8 Rk4, 3-4 Verlet)

3.2 Energy

Physically, an approximation method is not only measured by its error, but also how well it preserves physical laws. An important example in this application is energy conservation. Naturally, energy is conserved for an orbiting planet, but as mentioned in the theory section above, only one of the schemes implemented is time-symmetric (Verlet). Hence only it should show energy conservation. This is shown through a plot of the energy vs time for all 4 methods.

Note that the energy units are very complicated due to the nature of the system used.That is

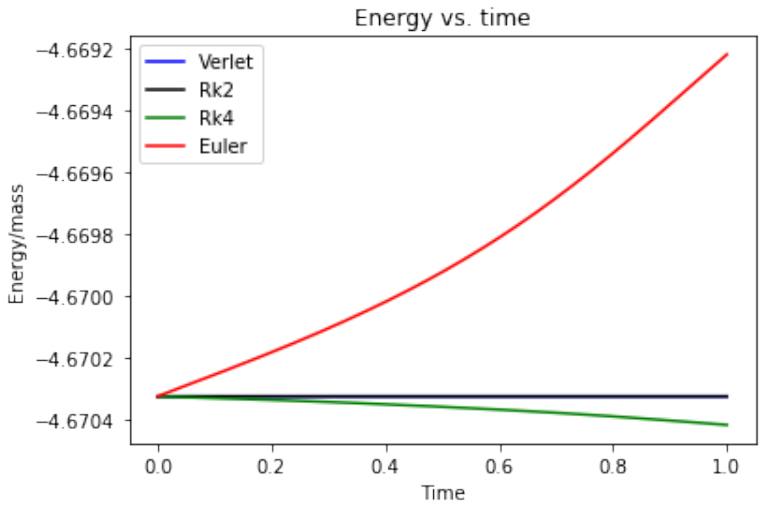

Figure 6: Not all methods conserve energy

not, however, the point of this plot.
Clearly, Euler and Rk4 do a bad job at conserving energy. Surprisingly, it seems like Rk2 conserves energy. This is surprising, as it is not a time symmetric formula. This could point to another unseen bug, or there could have been a miraculous coincidence that took place.

3.3 Root-Finding

As mentioned above, several other root-finding mechanisms were tried. They number of iterations they took to find the exact solution (within 1e-15) is shown in the table below, with an initial guess of E=M, where M is the mean anomaly.

Method	Number of Iterations
Newton-Rapheson	2
Relaxation	11
Bisection	48
Secant	4

As expected, NR is the best method, but the Secant method also works very well. Relaxation and Bisection are both quite a lot slower, though not in noticeable computational time for our purposes here.

4 Results: Halley's Comet and Adaptive Step Size

As with Mars, a plot of Halley's Orbit can be made:

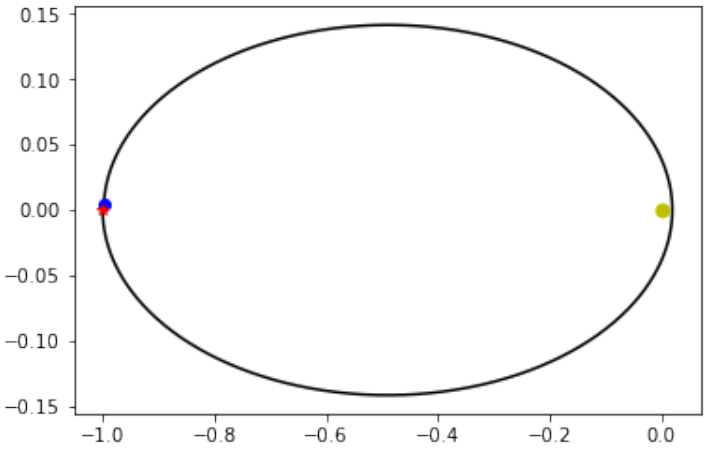

Figure 7: Halley, Adaptive Step Size

Now, we measure distance in units of Halley's perihelion and time in orbital time. The plot above is made using the adaptive step size scheme discussed in 2.5. The target error was set to 1e-7 (perihelions), and as can be seen it handles quite well measured against the exact solution (red star). It is expected that the adaptive step size, should significantly improve computational time, and so we check the running time of the adaptive step size vs Rk4 and Verlet for similar errors. See table below.

Method	Time (s)
Rk4-Adaptive	1
Rk4-Non-Adaptive	35
Verlet	5

Clearly the adaptive method is superior, as expected. The reason Rk4 is so slow is probably due to the same unknown bug mentioned above.

5 Results: S2

Finally, we explore the orbit of the star S2 around Sagittarius A*. To treat the relativistic effects, we replace the Newtonian potential with the Paczyński-Wiita potential, which is:

$$\phi = \frac{-GM}{r - r_s}$$

where

$$r_s = \frac{2GM}{c^2}$$

The Verlet method is used to compute several orbits (50) of the star. It looks as follows, this time in SI units:

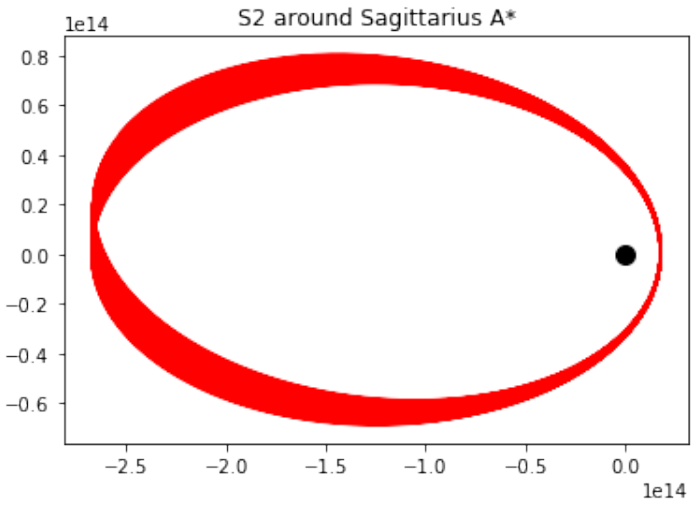

Figure 8: Precession of apsis of S2

Clearly, we can see the precession of the aphelion of S2. A quick calculation shows that the precession is about 0.192 radians after 50 orbits, equaling about 0.86 arcminutes/year.
For comparison, 50 orbits in a Newtonian potential would've looked like this:

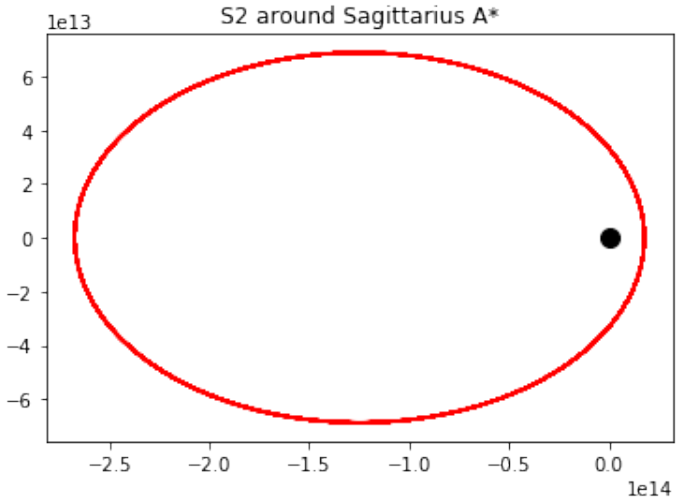

Figure 9: S2 in classical potential