Project: *Numerical solution of Newton's equations of motion*

Due: 16 July 2019, 12.00 noon

1 Introduction

The fundamental problem in Newton's dynamics is to solve the equation of motion of a particle, implied by the second law of dynamics with a given net force $\mathbf{F}$. Mathematically, we need to find the (unique) solution of the second-order ordinary differential equation (ODE)

$$\frac{\mathrm{d}^2\mathbf{r}}{\mathrm{d}t^2} = \frac{\mathbf{F}(\mathbf{v}, \mathbf{r}, t)}{m} \tag{1}$$

with the initial conditions[1] for the velocity $\mathbf{v}(0) = \mathbf{v}_0$ and the position $\mathbf{r}(0) = \mathbf{r}_0$. The problem is solved after we have found the position of our particle (and hence its velocity) at any instant of time t.

As we have mentioned in class, Newton's equations of motion not always can be solved analytically. That is, in some situations, the position of the particle cannot be expressed in terms of elementary functions in a closed form. In that case, numerical methods turn out to be particularly useful. The purpose of this project is to learn two basic numerical methods and apply them to solve Newton's equations of motion.

2 Basic numerical methods for solving equations of motion

Although the Newton's equation of motion (1) is a second-order ODE, it can be rewritten as a pair of coupled first-order ODEs by recalling the definitions of the velocity and the acceleration

$$\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = \frac{\mathbf{F}(\mathbf{v}(t), \mathbf{r}(t), t)}{m}, \qquad \frac{\mathrm{d}\mathbf{r}}{\mathrm{d}t} = \mathbf{v}(t). \tag{2}$$

Therefore the problem of solving a second-order ODE can be effectively reduced to the problem of solving two first-order ODEs of the form

$$\frac{\mathrm{d}f}{\mathrm{d}t} = G(f(t), t) \tag{3}$$

with a given initial condition $f(0) = f_0$.

Suppose that we want to solve the initial value problem (3) numerically, that is using a computer. We need to design a discrete algorithm that will provide us with a solution of the initial value problem. We need to keep in mind that this solution will be *approximate*, but by carefully choosing the algorithm and its parameters, or by introducing some refinements, usually we will be able to get quite a good approximation.

[1] Without loss of generality, we my assume that the initial instant of time is chosen as $t_0 = 0$.

2.1 Euler's method

Let us start with the Taylor expansion of f at the instant of time $t + \Delta t$ and neglect all terms of the order higher than one in Δt. Then

$$\begin{aligned} f(t+\Delta t) &\approx f(t) + \frac{\mathrm{d}f}{\mathrm{d}t}\Delta t \qquad (4)\\ &= f(t) + G(f(t),t)\Delta t, \qquad (5)\end{aligned}$$

where we have used Eq. (3) in the last step. Hence, if the value of the function f is known at time t, then its *approximate* value[2] $f^*(t+\Delta t)$ at a later instant $t+\Delta t$ can be found from Eq. (5).

In other words, Euler's method, starting at $t_0 = 0$ where $f(0) = f_0$ is known exactly, proceeds by using the slope k of the line tangent to the curve $f = f(t)$ at the left end the interval (t_0, t_1) to find the approximate value f^* of f at the other one (see Fig. 1 for the first step of the method). In subsequent steps, the approximate values of the function f obtained by the method are used successively.

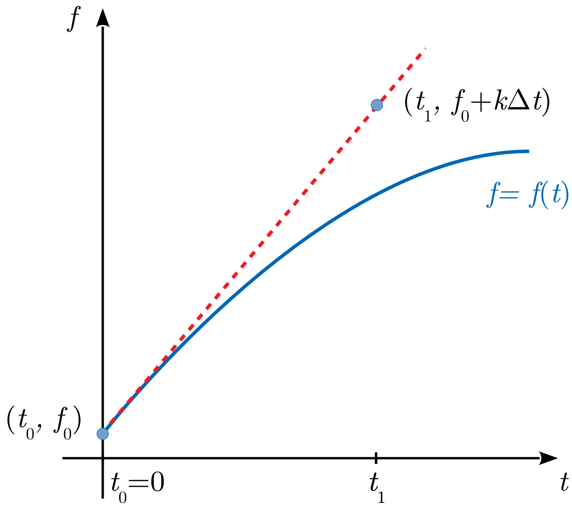

Figure 1: Illustration of Euler's method.

Since computers are discrete machines, we need to work on a discrete set of points representing the instants of time $t_i = i\Delta t$, with a discrete time steps Δt. Then, the algorithm based on Eq. (5), can summarized as follows

$$\begin{aligned} f_0^* &= f_0,\\ k &= G(f^*(t_i), t_i),\\ f^*(t_{i+1}) &= f^*(t_i) + k\,\Delta t.\end{aligned}$$

Or, simplifying the notation, as

$$\begin{aligned} f_0^* &= f_0,\\ k &= G(f_i^*, t_i),\\ f_{i+1}^* &= f_i^* + k\Delta t. \qquad (6)\end{aligned}$$

[2]In this project description the star symbol $*$ is used to denote the approximate values.

Given the initial condition $f(0) = f_0$, Eq. (6) can be iterated to find the (approximate) values of the function f in the required time interval, that is to numerically solve the first-order ODE (3) with that initial condition.

In particular, applying Euler's algorithm to the pair of ODEs (2), in the case of 1D motion, we get

$$
\begin{aligned}
v^*_{x,0} &= v_{x,0} && (7)\\
v^*_{x,i+1} &= v^*_{x,i} + \frac{F_x(v^*_i, x^*_i, t_i)}{m}\Delta t, && (8)\\
x^*_{x,0} &= x_0 && (9)\\
x^*_{i+1} &= x^*_i + v^*_{x,i}\Delta t. && (10)\\
t_{i+1} &= i\Delta t && (11)
\end{aligned}
$$

Now, starting with $i = 0$ and the initial values of the position and the velocity, and running the algorithm in a loop, we are able to find both the position and the velocity of the particle at any later instant of time.

2.2 Heun's method

Recall that Euler's algorithm, starting at $t_0 = 0$ where $f(0) = f_0$ is known exactly, uses the slope of the tangent line to $f = f(t)$ at the left end of the interval (t_0, t_1), to approximate the value of f at the right end.

We may refine this method by first evaluating the slope at the left end of the interval, which is $k_1 = G(f_0, t_0)$, and then at the right end. For the slope on the right end, Euler's method is used to predict the (unknown) value of f at t_1, that is the slope there is evaluated as $k_2 = G(f_1 + k_1\Delta t, t_1)$. Then, the average value of the slope at both ends is used to evaluate a corrected prediction f^*_1 of the value of f at $t = t_1$ (see Fig. 2, where the first step of the method is shown). From then on, the algorithm continues in the same way with the found approximate values of the function f. This scheme is known as Heun's method[3]

Implementing the described idea, the set of equations used to numerically solve this differential equation can be written in the following form

$$
\begin{aligned}
f^*_0 &= f_0, &&\\
k_1 &= G(f^*_i, t_i), && (12)\\
k_2 &= G(f^*_i + k_1\Delta t, t_i + \Delta t), && (13)\\
f^*_{i+1} &= f^*_i + \frac{1}{2}\left(k_1 + k_2\right)\Delta t. && (14)
\end{aligned}
$$

It allows us to find the approximate value of f on the required interval.

[3]Heun's method belongs to a class of methods for solving ODEs called Runge–Kutta methods, which are examples of *predictor–corrector* algorithms. Heun's method is also known as the 2nd order Runge–Kutta method

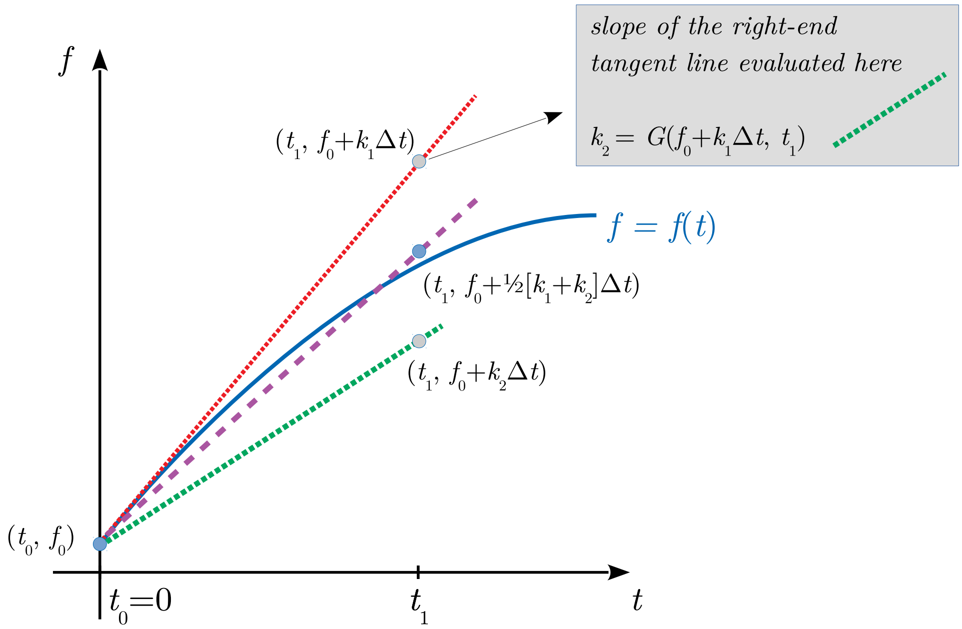

Figure 2: Illustration of Heun's method.

3 Problems and tasks

3.1 Projectile motion with air drag

First consider the problem of a 2D projectile of mass $m = 1$ kg moving with a linear drag, moving close to the Earth's surface. The equation of motion in this problem is

$$\ddot{\mathbf{r}} = -\mathbf{g} - \kappa\mathbf{v},$$

where $\kappa = k/m$ and k is the drag coefficient. We assume that the projectile starts out at the origin with velocity $\mathbf{v}_0 = 90$ m/s at an angle α to the horizontal.

1. Use Euler's method to write down two pairs of recursive equations for velocity and position of the projectile (both for x and y components).

2. For fixed initial conditions examine whether the numerical result is sensitive to the choice of the step Δt. To do so, choose three different values of Δt, plot the trajectory and compare with that obtained from the analytical formulas.

   Hint. You may visualize the difference by calculating $\mathbf{r}_{\text{analytical}} - \mathbf{r}_{\text{numerical}}$.

3. Based on the results from the previous point choose an appropriate value of the step. With the initial speed of the projectile fixed, solve the problem numerically for five different values of the angle α.

   (a) On one graph plot the trajectories and comment on their shape shape (give qualitative comments on the range and the maximum height).

   (b) On one graph plot the time dependence of the speed of the particle. Comment on the result.

4. With the angle α fixed, solve the problem numerically for five different values of the drag coefficient k.

   (a) On one graph, plot the trajectories and comment on their shape shape (give qualitative comments on the range and the maximum height).

   (b) On one graph, plot the time dependence of the speed of the particle. Comment on the result.

In the next part, suppose that the same particle is subject to a quadratic drag. That is, its equation of motion reads

$$\ddot{\mathbf{r}} = -\mathbf{g} - \beta|\mathbf{v}|\mathbf{v},$$

where $\beta = b/m$ and b is the drag coefficient. We assume again that the projectile starts out at the origin with velocity $\mathbf{v}_0 = 90$ m/s at an angle α to the horizontal.

5. Use Euler's method to write down the recursive equations for velocity and position of the projectile.

6. For fixed initial conditions examine whether the numerical result is sensitive to the choice of the step Δt. To do so, choose three different values of Δt and plot the trajectory. Comment on the result.

7. Based on the results from the previous point choose an appropriate value of the step. With the initial speed of the projectile fixed, solve the problem numerically for five different values of the angle α.

   (a) On one graph, plot the trajectories and comment on their shape shape (give qualitative comments on the range and the maximum height).

   (b) On one graph, plot the time dependence of the speed of the particle. Comment on the result.

8. With the angle α fixed, solve the problem numerically for five different values of the drag coefficient k.

   (a) On one graph, plot the trajectories and comment on their shape shape (give qualitative comments on the range and the maximum height).

   (b) On one graph, plot the time dependence of the speed of the particle. Comment on the result.

9. Choose one set of the initial conditions and on one graph plot the trajectories of a projectile moving (a) without air drag, (b) with a linear drag, and (c) with a quadratic drag. For the drag coefficient choose the values that give the same terminal speed.

3.2 Simple harmonic oscillator

In this part you are asked to numerically solve the equation of motion of the 1D simple harmonic oscillator $\ddot{x} = -\omega_0^2 x$, with the initial conditions $x(0) = 0.5$ m and $v_x(0) = 1$ m/s. Assume the natural angular frequency $\omega_0 = 1.5$ s^{-1}.

1. Use Euler's method to write down the recursive equations for velocity and position of the oscillating particle.

2. Examine whether the numerical result is sensitive to the choice of the step Δt. To do so, choose three different values of Δt, plot the trajectory in both configuration and phase space and compare with those obtained from the analytical formulas.

3. At every step calculate $E(t) = \frac{1}{2}m\omega_0^2 x^2 + \frac{1}{2}mv_x^2$ and plot the graph $E(t)$. Comment on the result. Does the effect you observe have anything to do with physics?

4. Write down the recursive Heun's equations (12)–(14) for v_i and x_i for the simple harmonic oscillator.

5. Choose the worst–performing step size Δt from point 2 and use Heun's method with this step size to solve the problem again.

   Plot the trajectory of the particle both in configuration and phase space. Plot Euler's, Heun's, and the analytical results together on one graph. Comment on the plot.

   At every step calculate $E(t) = \frac{1}{2}m\omega_0^2 x^2 + \frac{1}{2}mv_x^2$ for both Euler's and Heun's method, and plot the graph $E(t)$. Comment on the result.

   Note. In points 2, 3, and 5 make sure that the time interval covers several periods.

4 Remarks on the accuracy of numerical methods

Let us emphasize that the solutions we find by applying numerical methods are approximate. This is clearly visible in the case of the harmonic oscillator. There is a few factors limiting the accuracy of numerical methods. First of all, the algorithms are discrete, therefore approximations are inevitable. However, here we can control the accuracy to some extent, by constructing more sophisticated algorithms and properly choosing the time step. The accuracy of numerical methods is also limited by the precision of computer's floating-point arithmetic, although in our case it does not significantly affect the numerical solution.

Exercise *(optional, but you may include the solution in your report)*

Check that Heun's method discussed in this project is a second-order method, that is it is accurate to second order in Δt. What is the order of Euler's method?

5 Deliverables

After completing all tasks you need to write a self-contained report presenting your results. It should be starting with a short introduction, similar to the opening sections of this document. The report should be typed (LaTeX is recommended, but not compulsory) and submitted in a printed form, including graphs, with all pages stapled together.

You will need to implement the two numerical methods studied in this project. There are no restrictions imposed on the programming environment (C++, Fortran, Matlab, Mathematica, Octave, Maxima,... are all allowed). Please keep your code clear, simple, and concise. The source code files and the report pdf file should be packed into a single ZIP file and uploaded to Canvas by the due date.

In this project, you will be working in groups of four. You need to make sure that workload is distributed uniformly among all group members. On the title page please include the following statement: *"We state that each of us has contributed equally to this project."* and sign it. The code should be also labelled with the names of group members. Each group should submit a single report and a single ZIP file.

6 Honor Code

The JI Honor Code applies to the project in the same way as it does to homework. In particular, according to the current version of the Honor Code[4]

> *Assignments involving collaboration within a group (e.g., lab reports, project reports, collaborative course work) require that all members of the group whose name appears on the assignment are jointly and fully responsible for the entirety of the submitted work. If any section of the submission is found to violate the Honor Code, all group members whose name appears on the submission are equally and jointly liable for the violation.*

[4] `http://umji.sjtu.edu.cn/academics/academic-integrity/honor-code/`, as of 21 June 2019