Analyse Numérique / Computational Physics (M1) - T. Roscilde, F. Caleca

General rule for the TD sessions: the TD sessions are fully *hands-on* – namely, in every TD session you are supposed to *write computer codes* to learn about the phenomenology and efficiency of important algorithms, or, more ambitiously, to learn the physics of simple, yet fundamental models. You should choose a programming platform (Python, Matlab, Mathematica, C, Fortran, etc.), and you should be able to plot your results in the form of two-dimensional functions $y = f(x)$ (using matplotlib in Python, the plotting utilities of Matlab and Mathematica, Gnuplot, etc.), or occasionally in a more complicated form. We assume that you have some familiarity with at least one programming platform; if this is not the case, you should be able to familiarize yourself rapidly *e.g.* by attending online tutorials.

TD4: Numerical solution of ODEs

In this exercise sheet, we shall study numerically the physics of oscillators – and in so doing experience a few methods of solution for systems of ordinary differential equations (ODEs).

1 The pendulum

The equation of motion for the angle θ formed by a pendulum of length l with respect to the vertical direction is

$$\ddot{\theta}(t) = -\frac{g}{l}\sin\theta(t) \ . \tag{1}$$

In the following we shall measure time in units of $\sqrt{l/g}$, so that the g/l disappears from the equation. We can turn this second-order differential equation into a system of first-order ones by introducing the variables $y_1(t) = \theta(t)$, $y_2(t) = \dot{\theta}(t)$, so that we get the system

$$\begin{cases} \dot{y}_1 = y_2 \\ \dot{y}_2 = -\sin y_1 \end{cases} \tag{2}$$

namely the system $\dot{\boldsymbol{y}} = \boldsymbol{f}(\boldsymbol{y})$ with $\boldsymbol{f}(\boldsymbol{y}) = (y_2, -\sin y_1)$. We shall explore its solution upon changing the initial conditions $\boldsymbol{y}(0) = \boldsymbol{y}^{(0)}$.

The energy of the pendulum (in units of mgl, with m the mass of the pendulum) is given by

$$E(\boldsymbol{y}) = \frac{y_2^2}{2} - \cos(y_1) \tag{3}$$

and it is conserved by the exact dynamics.

1.1 Euler method - phase portraits

1.1.1

Set up the Euler method

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\boldsymbol{f}(\boldsymbol{y}_n, t_n) \tag{4}$$

to solve the above system over a time window $[0, t_0]$. You can choose $t_0 = 10$ (but check whether this time interval is enough to see all the interesting physics of the problem), and a variable h, defining the number of steps $M = t_0/h$. You could start from $h = 0.1$, but you should check that your results do not depend too much on h by taking a smaller (or larger) step and see what happens.

1.1.2

We shall first reconstruct the behavior of the solution with initial conditions $y_1^{(0)} = \theta^{(0)}, y_2^{(0)} = \dot{\theta}^{(0)} = 0$. Convince yourself that $y_1^{(0)} = 0$ and $y_1^{(0)} = \pi$ are stationary points (the solution does not evolve), and that the Euler method would give you the exact solution in this case. Then monitor the evolution using the initial conditions $y_1^{(0)} = \pi/4, \pi/2, 3\pi/4$ (or even a finer grid in the interval $[0, \pi]$). For each of these initial conditions, record the sequence of points $(y_{1,n}, y_{2,n})$ in a vector of length $M + 1$, and make a plot of $y_{2,n}$ vs. $y_{1,n}$ – this is what is called a phase portrait of the dynamics. You should see that the phase portraits are limit cycles – something familiar, because we know that the pendulum oscillates! (On the other hand, if you choose too large a h, you should see that limit cycles are not properly reproduced).

1.1.3

Now we introduce a second class of initial conditions, namely $y_1^{(0)} = 0$ and a variable $y_2^{(0)}$ over the interval $[0, 3]$ (but you could look at larger values as well). Study the evolution for $y_2^{(0)} = 0.5, 1, 1.5, 2.5, 3$: you should observe that something very drastic happens to the phase portraits at some value of $y_2^{(0)}$: can you make sense of it? What kind of motion of the pendulum corresponds to the new phase portraits?

1.1.4

If you have recorded the phase portraits $(y_{1,n}, y_{2,n})$, you can check whether energy $E_n = E(y_{1,n}, y_{2,n})$ (plotted as a function of n) is conserved along them. You will observe that conservation of energy is far from perfect, but it improves when h is smaller.

1.2 Runge-Kutta 4

Now that you have the physics of the problem under control, we can look at some algorithmic refinements, and implement the celebrated *RK4 method*, which is a method of order $p = 4$:

$$
\begin{aligned}
\boldsymbol{k}_1 &= h\boldsymbol{f}(\boldsymbol{y}_n, t_n) \\
\boldsymbol{k}_2 &= h\boldsymbol{f}(\boldsymbol{y}_n + \boldsymbol{k}_1/2, t_n + h/2) \\
\boldsymbol{k}_3 &= h\boldsymbol{f}(\boldsymbol{y}_n + \boldsymbol{k}_2/2, t_n + h/2) \\
\boldsymbol{k}_4 &= h\boldsymbol{f}(\boldsymbol{y}_n + \boldsymbol{k}_3, t_n + h) \\
\boldsymbol{y}_{n+1} &= \boldsymbol{y}_n + \frac{1}{6}(\boldsymbol{k}_1 + 2\boldsymbol{k}_2 + 2\boldsymbol{k}_3 + \boldsymbol{k}_4)
\end{aligned}
\tag{5}
$$

You can use it for a couple of initial conditions (for instance $(y_1^{(0)}, y_2^{(0)}) = (3\pi/4, 0)$ and $(0, 2.5)$), and compare the results with those of the Euler scheme. In particular, for the same value of h (you could take $h = 0.5, 0.1, 0.05$), you can compare the energy curves E_n obtained with the two methods. What can you deduce?

1.3 Implicit midpoint method

An (only apparently) less laborious Runge-Kutta method is represented by the *implicit midpoint method*

$$
\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + h\boldsymbol{f}\left(\frac{\boldsymbol{y}_n + \boldsymbol{y}_{n+1}}{2}, t_n + \frac{h}{2}\right) \tag{6}
$$

which defines a set of two non-linear equations

$$
\begin{cases}
y_{1,n+1} = y_{1,n} + \frac{h}{2}(y_{2,n} + y_{2,n+1}) \\
y_{2,n+1} = y_{2,n} - h\sin\left(\frac{y_{1,n}+y_{1,n+1}}{2}\right)
\end{cases} . \tag{7}
$$

This method of order 2 has the advantage of being *symplectic* – namely it should guarantee a much better conservation of the energy.
To implement this method, you should be able to solve the above system of equations iteratively: namely you can start with the initial guess $(y^{(0)}_{1,n+1}, y^{(0)}_{2,n+1}) = (y_{1,n}, y_{2,n})$, and then inject it into the right-hand side of Eqs. (7) to get a new estimate $(y^{(1)}_{1,n+1}, y^{(1)}_{2,n+1})$, etc. until convergence is reached, namely until $|y^{(k+1)}_{1,n+1} - y^{(k)}_{1,n+1}|, |y^{(k+1)}_{2,n+1} - y^{(k)}_{2,n+1}| < \eta$ where η is some tolerance that you fix (for instance, you can take $\eta = 10^{-3}$).
For the same value of h and for the same initial conditions, you should compare the evolution of the energy that you obtain with the Euler method ($p = 1$), the RK4 method ($p = 4$) and the implicit midpoint method ($p = 2$). What can you deduce?

2 Van der Pol oscillator

The van der Pol (vdP) oscillator is a model of the dynamics of some special electrical circuits, and its equation of motion reads:

$$\ddot{x} - \mu(1 - x^2)\dot{x} + x = 0 \tag{8}$$

where μ is a parameter that will be varied. Clearly when $\mu = 0$ one recovers a simple harmonic oscillator; on the other hand, for $\mu > 0$ the system describes an oscillator subject to a position-dependent viscous force.
The first-order ODE system corresponding to the vdP oscillator reads

$$\begin{cases} \dot{y}_1 = \dot{x} = y_2 \\ \dot{y}_2 = \ddot{x} = \mu(1 - y_1^2)y_2 - y_1 \end{cases} \tag{9}$$

This system of equations represents a *stiff* one – with stiffness controlled by the parameter μ – which may require a very small time step h to be properly studied.

2.1 Euler method and phase portraits

Starting from the initial condition $y_1^{(0)} = 1, y_2^{(0)} = 0$, and, using the Euler method, reconstruct the phase portraits (plots of $y_{2,n}$ vs. $y_{1,n}$) for $\mu = 0, 2, 5, 10, \ldots$. Choosing a given h (e.g. $h = 0.1$), you should observe that for a sufficiently small μ a limit cycle (with an asymmetric shape) appears in the evolution; while for a larger μ the limit cycle is no longer well described. Given a value of μ for which you do not observe a well defined limit cycle for $h = 0.1$, reduce h until you find the limit cycle.

2.2 Trapezoidal scheme

To take care of the stiff nature of the system, you can use an implicit scheme, namely the *trapezoidal rule*

$$\boldsymbol{y}_{n+1} = \boldsymbol{y}_n + \frac{h}{2}\left[\boldsymbol{f}(\boldsymbol{y}_n, t_n) + \boldsymbol{f}(\boldsymbol{y}_{n+1}, t_n + h)\right] \; . \tag{10}$$

Write down the system of equations, which you may be able to solve analytically for $y_{1,n+1}, y_{2,n+1}$ – or you can resort to an iterative solution for implicit schemes, as described in the previous exercise.
Choosing the same time step $h = 0.1$, can you observe the correct limit cycle for values of μ at which the Euler scheme failed to give you the correct result?