Assignment #9

Due Monday, 2 December 2024, at the start of class

This Assignment is based on Chapter 11 of our textbook.[1] Please read all of sections 11.1 and 11.2; the latter is very substantial! You can skip section 11.3, but please read section 11.4 through page 289; you can skip the rest (pages 290–294).

These expectations always apply to homework:

1. Please put the problems in the order they appear below.
2. When you use MATLAB/etc, show the commands and the results.
3. Keep a clear distinction between codes, input commands, computed results, and figures.
4. Other than the text you write, please minimize use of paper.

Do these problems:

P13. There is a famous nonlinear ODE system which arose[2] when the programmers of the first numerical weather forecast models were trying to understand the dynamics of the atmosphere:

$$\begin{aligned} x'(t) &= \sigma(y - x) \\ y'(t) &= x(\rho - z) - y \\ z'(t) &= xy - \beta z \end{aligned}$$

Standard parameter choices for this *Lorenz system* are $\sigma = 10, \rho = 28, \beta = 8/3$. Use `ode45()` to numerically solve this problem from initial conditions $x(0) = 1$, $y(0) = 1$, and $z(0) = 1$, until time $T = 100$. Use `plot3()` to get a rotatable figure. Describe in a few words what you see.

(Extra Credit) The Lorenz system exhibits a clear form of *sensitive dependence on initial conditions*, in a bounded dynamical system, which is one aspect of *chaos*. By making slight changes to the initial condition, e.g. one part in 10^8 or less, show this sensitive dependence. In particular, use color to distinguish two solutions which start very close in this sense, but which are on different wings of the butterfly after a not-very-long time. To show this effect you should use tight tolerance settings for `ode45()`.

[1] Greenbaum & Chartier, *Numerical Methods: Design, Analysis, and Computer Implementation of Algorithms*, Princeton University Press 2012).

[2] It is far from obvious how this ODE system is related to the atmosphere, though the version here is now standard in textbooks. Reading Lorenz's original paper is challenging, but it shows how he reduced the equations to just three.

P14. *This problem simplifies/clarifies Exercise 14 in Chapter 11. Please read that Exercise for more about the physical/astronomical context.*

This problem concerns the motion of a satellite around the sun. The satellite could be a planet, a comet, or an unpowered spacecraft. We assume the satellite mass is small compared to the sun, and so the sun is fixed at the origin. The coordinates of the satellite, in the plane of its orbit, are $x(t)$ and $y(t)$. The equations for the satellite motion are Newton's laws. In simplified units the ODE system is:

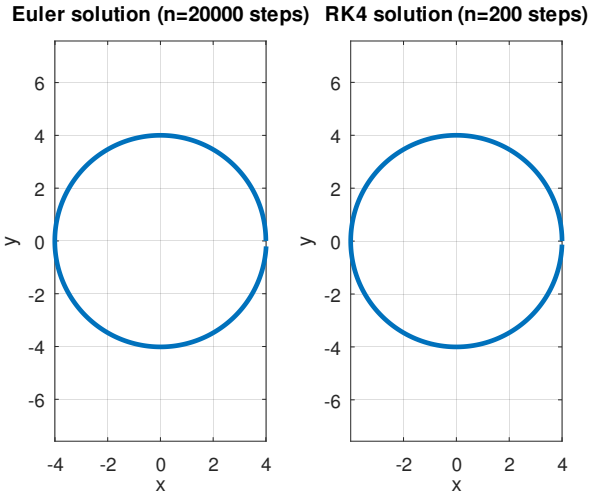
$$x'' = \frac{-x}{(x^2+y^2)^{3/2}}, \qquad y'' = \frac{-y}{(x^2+y^2)^{3/2}}$$

(a) Re-write these two second-order ODEs into a system of 4 first-order ODEs. Two equations in the new system define the velocities: $z(t) = x'(t)$ and $w(t) = y'(t)$. The resulting equations can be written

$$\mathbf{u}'(t) = \mathbf{f}(\mathbf{u}(t))$$

for a (column) vector $\mathbf{u}(t) = [x(t), y(t), z(t), w(t)]^\top$; you are asked to precisely state the function $\mathbf{f}$.

(b) Solve the ODE system from **(a)** using Euler's method and the RK4 method. In particular, use initial conditions $x(0) = 4, y(0) = 0, z(0) = 0, w(0) = 0.5$, and assume the duration is $T = 50$. For Euler's method use $h = 0.0025$ and for RK4 use $h = 0.25$. Plot the position solution[3] $x(t), y(t)$ for each method, and use `axis equal`. You should see a circular orbit, with very similar results, from each method. My code used `subplot()` to produce the Euler and RK4 results side-by-side (below). If your code does not produce equally circular orbits then it has a bug.



[3]The full solution $\mathbf{u}(t)$ is in four dimensions, and not easily visualizable.

(c) Fixing the other parameters, make the time step sizes h substantially bigger and observe that the orbits are no longer accurate circles. Make the steps sizes smaller and observe that you get the same circular result; results have converged at screen resolution. Observe that Euler is terribly inefficient here.

(d) Duplicate your previous code to make a new one which only uses the RK4 method. Repeat the experiment in **(b)**, but this time make a loop around the solver which runs these 6 cases and produces a plot for each:

case	$w(0)$	h	T
1	0.5	0.25	50
2	0.66	0.5	500
3	0.8	0.5	200
4	0.4	0.25	35
5	0.2	0.25	30
6	0.2	0.05	30

All other parameters have the same values as in **(b)**. Cases 5 and 6 differ only by the step size, so they correspond to the same continuum problem, but you will see very different results. What is going on? Which one is closer to correct, and why?

(Extra Credit) Use `ode45()` to re-do cases 5 and 6, which will be the same because you don't specify h. Adjust the relative and absolute tolerance to tighter values than the default. Show that the adaptive time-stepping in `ode45()` is producing very short time steps when the satellite is near the sun. Explain why this is happening, in physical terms.