Exercise # 1. Numerical methods for ODES.

Luca Bergamaschi – Andrea Franceschini

October 26, 2021

1. Consider the 2-step Simpson's method

$$y_{n+2} = y_n + \frac{h}{3}\left(f_n + 4f_{n+1} + f_{n+2}\right)$$

   (a) Implement the method to solve the test equation $y' = -5y,\ y(0) = 1$ with $h = 0.02, T \equiv t_N = 10$.

   (b) Compute the second initial value y_1 (a) with the forward Euler method and (b) by a 4th order Runge-Kutta method.

   (c) In both cases plot the error as the difference between the exact and the approximate solution at the grid points.

   (d) Comment the different behavior observed by the numerical method.

   (e) **Difficult (optional)** Consider now $y' = -5y,\ y(0) = \alpha$. There is a value of α for which the method is stable (in exact arithmetics). Find this value.

   (**Hint**: the solution to the difference equation is $y_n = c_1 z_1^n + c_2 z_2^n$, with $|z_1| > 1, |z_2| < 1$ so $y_n \to 0 \implies c_1 = 0, \ldots$)

2. Use the 4-stage Runge-Kutta method to solve the equation

$$\begin{aligned} y'(t) &= -10y^2(t) \\ y(0) &= 1 \end{aligned}$$

   $t \in (0, 2)$ with $h = 2^{-k},\ k = 5, \ldots, 10$. For all (six) values of h solve the equation and compute the error at the final time $T = 2$. Provide a `loglog` plot of this error as a function of the number of steps. Produce also a table with the values of h and of the error in each row. Check if the error reduces (progressively halving h) in accordance with the theory.

3. A BDF formula is obtained by first writing the ODE on a point t_{n+k} as $y'(t_{n+k}) = f(t_{n+k}, y_{n+k})$. Then the unknown function y is interpolated at the left hand side of $y' = f(t, y)$ using the nodes $t_n, \ldots, t_{n+k}$ and then differentiated. Obtain the BDF2 formula ($k = 2$) and find its local truncation error (Hint: recall that the interpolation error is

$$E(t) = \frac{(t - t_{n+k}) \ldots (t - t_n) f^{(k+1)}(\eta(t))}{(k+1)!}.$$

To obtain the local truncation error you should find the maximum of the derivative of $|E'(t)|$, which usually is taken at the endpoints of the interval).

Prove that BDF2 is absolutely stable in the real interval $\bar{h} \in (-\infty, 0)$. (Hint: the characteristic polynomial for this method applied to the test equation is

$$t^2(3 - 2\bar{h}) - 4t + 1 = 0$$

with roots

$$t_{12} = \frac{2 \pm \sqrt{1 + 2\bar{h}}}{3 - 2\bar{h}}$$

if $-\frac{1}{2} \leq \bar{h} < 0$ then t_{12} are both real and ... If, otherwise, $\bar{h} < \frac{1}{2}$ then t_{12} is complex and its modulus is ...)

4. We want to solve the linear system of ODEs

$$\begin{aligned} \boldsymbol{y}'(t) &= -A\boldsymbol{y}(t) \\ \boldsymbol{y}(0) &= [1,\ 1,\ \ldots, 1]^T \end{aligned}$$

with A[1] generated through the following Matlab commands

```
nx = 100;
G = numgrid('S',nx);
A = delsq(G)*(nx-1)^2,
```

and $t \in (0, 0.1]$. The *exact* solution to this problem at the final time is

$$\hat{\boldsymbol{y}} = \exp(-0.1A)\boldsymbol{y}_0,$$

which is available on the Moodle page as a file (`exact_solution`)[2]

Denote by λ the eigenvalue of A with largest modulus (in Matlab this can be computed using the command `lambda = -eigs(A,1,'lm')` which delivers minus the largest eigenvalue of A.)

(a) Compute the interval of absolute stability of the 4th order Runge-Kutta method for this problem. Then solve the problem by `ode45` with $t_0 = 0$, $T = 0.1$. Compute the final error (the infinity norm of the difference between the exact and the approximate solution), and the CPU time elapsed (use e.g. `tic` and `toc` commands).

(b) Solve the given problem with the Crank Nicolson (CN) method with $h \in \{h^{-3}, h^{-4}, h^{-5}\}$, recording for each case the final error obtained and the CPU time.

(c) Solve the given problem with the BDF3 method using $h \in \{h^{-3}, h^{-4}, h^{-5}\}$, recording for each case the final error obtained and the CPU time.

Remark. With methods CN and BDF3 a linear system must be solved at each timestep. You can either solve it with the \ of Matlab or (better) use the Conjugate Gradient method (no preconditioning) with tolerance $\texttt{tol} = h^3$.

Provide a table reporting, for each of the 7 runs, the method, the number of steps employed, the error obtained, and the CPU time elapsed. Are the error for CN and BDF3 consistent with the theory? What is the best method for a given accuracy?

[1] $-A$ is the 5-points Finite Difference discretization of the 2D Laplacian

[2] If you like you can compute it as `y = expm(-0.1*A) * ones(n,1)`. You need much memory for this!

5. The Lotka–Volterra equations, also known as the predator–prey equations, are a pair of first-order nonlinear differential equations, frequently used to describe the dynamics of biological systems in which two species interact, one as a predator and the other as prey. The populations change through time according to the pair of equations:

$$\begin{cases} x'(t) &= x(t)\,(\alpha - \beta y(t)) \\ y'(t) &= y(t)\,(\delta - \gamma x(t)) \\ x(0) &= x_0 \\ y(0) &= y_0, \end{cases} \tag{1}$$

where

- x is the number of prey (for example, rabbits);
- y is the number of some predator (for example, foxes);
- $x'(t)$ and $y'(t)$ represent the instantaneous growth rates of the two populations;
- t represents time;
- $\alpha, \beta, \gamma, \delta$ are positive real parameters describing the interaction of the two species.

Exercise.

(a) Solve the Lotka-Volterra equation with parameters $\alpha = 0.2, \beta = 0.01, \gamma = 0.07, \delta = 0.004$ by a 4th order RK method using as initial conditions $x_0 = 19, y_0 = 22$, $t_0 = 0, T = 300$ and a stepsize $h = 10^{-3}$.

(b) Produce a picture with the numerical solution of the prey ($x(t)$) and the predator ($y(t)$) in the interval $[0, 300]$.

Maximum grade: 12 points. The sum of the grades of the three mandatory exercises will average the grade of the oral exam.

Deadline for delivering the Homework: **November 25, 2021**. The deadline is not mandatory. (However, deliver after the deadline implies a reduction of the grade to Max: 10 points).

Upload on the Moodle page an archive with all the source files (e.g. M-files in Matlab) and a report containing the answer to theoretical questions, a brief description of the methods, some comments to the results and all the outputs. Regarding the report, I suggest it to be written in LaTeX, a document preparation system very suitable for mathematical formula rendering. LaTeXis publicly available for all operating systems; User's manuals are also free to download.