ESS101 Mod & Sim
SP1 2019
Assignment 3 - Explicit Integrators
Due date: 2019-10-18

In this assignment we will investigate some widely-used explicit integration techniques, and their behaviour for the simulation of various dynamic systems.

Instructions

The assignments comprise an important part of the examination in this course. Hence, it is important to comply with the following rules and instructions:

- The assignment is pursued in groups of two students. For group formation procedures, see Canvas.
- The findings from each assignment are described in a short report, written by each group independently.
- The report should provide clear answers to the questions, including your motivations, explanations, observations from simulations etc. Figures included in the report should have legends, and axes should be labelled.
- Since the assignments are part of the examination in the course, plagiarism is of course not allowed. If we observe that this happens anyway, it will be reported.
- The report should be uploaded to Canvas *before the deadline*. Matlab code is uploaded as separate files. If the deadline is not met, we cannot guarantee that you will be able to complete the assignment during the course.
- The written report is graded PASS, REVISE or RESUBMIT. With a decision REVISE, you will get a new due date to provide a revised report with changes highlighted. With a decision RESUBMIT, you will need to submit a completely rewritten report.

Please be environment-friendly: avoid printing these pages!

1. **General Runge-Kutta 2 methods** For a dynamic model:

$$\dot{\mathrm{x}} = \mathrm{f}(\mathrm{x}, \mathrm{u}, t) \tag{1}$$

where for the sake of simplicity we assume the input $\mathrm{u} = \mathrm{u}_k$ to be constant on the time interval $[t_k, t_{k+1}]$, RK2 methods are based on 2 evaluations of the function f, the first of which occurs at $\mathrm{x}(t_k), \mathrm{u}_k$. They can generally be written as:

$$\mathrm{k}_1 = \mathrm{f}(\mathrm{x}(t_k), \mathrm{u}(t_k), t_k) \tag{2a}$$
$$\mathrm{k}_2 = \mathrm{f}(\mathrm{x}(t_k) + a \cdot \Delta t \cdot \mathrm{k}_1, \mathrm{u}(t_k + c\Delta t), t_k + c\Delta t) \tag{2b}$$
$$\mathrm{x}_{k+1} = \mathrm{x}(t_k) + \Delta t \sum_{i=1}^{2} b_i \cdot \mathrm{k}_i \tag{2c}$$

and have the Butcher tableau (zeros in the "a" part are traditionally omitted)

$$\begin{array}{c|cc} 0 & & \\ c & a & \\ \hline & b_1 & b_2 \end{array}$$

The RK2 method will have a numerical error $e_k = \mathrm{x}_{k+1} - \mathrm{x}(t_{k+1})$ (where $\mathrm{x}(t)$ is the true trajectory of the ODE).

(a) For the case of simplicity, consider the case where f is time-independent (t does not enter as an argument) and $\mathrm{u}(t) = \mathrm{u}_k$ is constant over the time interval $[t_k, t_{k+1}]$. Provide conditions on a, b such that the error e_k of the method is of order 3.
Hint: observe that:

$$\mathrm{x}(t_{k+1}) = \mathrm{x}(t_k) + \Delta t \cdot \mathrm{f}(\mathrm{x}(t_k), \mathrm{u}_k) + \frac{\Delta t^2}{2} \cdot \dot{\mathrm{f}}(\mathrm{x}(t_k), \mathrm{u}_k) + \mathscr{O}\left(\Delta t^3\right) \tag{3}$$

(b) RK2 methods are at best of order 2. How does that relate to $e_k = \mathscr{O}\left(\Delta t^3\right)$?

(c) Consider a system having a trajectory given by a polynomial of order n, i.e.:

$$\mathrm{x}(t) = \mathrm{x}(t_k) + \sum_{i=1}^{n} \frac{\alpha_i}{i!} (t - t_k)^i, \quad t \in [t_k, t_{k+1}] \tag{4}$$

For what values of n, b, c, the RK2 method is exact on (4)? What do you observe from your computations?

2. **Accuracy & stability** We will start with investigating the accuracy/computational cost of some explicit integration schemes. We will consider the explicit Euler scheme and a Runge-Kutta scheme of order 2 and 4. We will test them on the classic test system:

$$\dot{x} = \lambda x \tag{14}$$

where $\lambda < 0$. The Butcher tables of these three schemes are provided hereafter.

(a) Code an explicit Euler scheme (RK1), and an RK2 and RK4 scheme for a generic function $f(x)$ (*declare your dynamics in a symbolic way, and generate a Matlab function that evaluates it, using "matlabFunction". The most comfortable way of coding this is also to have a code that "reads" the Butcher Tableau and deploys from it the adequate steps*). Test your codes for $\lambda = -2$ and $\Delta t = 10^{-1}$, simulating for $t_\mathrm{f} = 2$. Start with the initial condition $x = 1$.

(b) Investigate the evolution of the accuracy of your integrators as a function of Δt against the true solution of (14). Report your results and compare them to the theoretical order of accuracy of the various schemes.

(c) For what value of $\lambda < 0$ will the different schemes become unstable?

Table 1: Explicit Euler

$$\begin{array}{c|c} 0 & \\ \hline & 1 \end{array}$$

Table 2: Runge-Kutta 2

$$\begin{array}{c|cc} 0 & & \\ \frac{1}{2} & \frac{1}{2} & \\ \hline & 0 & 1 \end{array}$$

Table 3: Runge-Kutta 4

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & & \frac{1}{2} & & \\ 1 & & & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}$$

3. **Van-der-Pol oscillator** Consider the nonlinear dynamics:

$$\dot{x} = y \tag{15a}$$
$$\dot{y} = u\left(1 - x^2\right) y - x \tag{15b}$$

for $u = 5$, a final time of $t_\mathrm{f} = 25$ and $\Delta t = 10^{-2}$.

(a) Simulate the dynamics (15) using the Matlab integrator ode45 (this is an adaptive Runge-Kutta scheme of 5 with alternate Butcher tableau of order 4. You can use the default options). Plot the discrete times selected/reported by ode45. What do you observe?

(b) Deploy your own RK4 scheme on the same dynamics, compare the solutions. What Δt do you need in order for you solution to accurately match the one of the function ode45? Compare your discrete time grid to the one of ode45.