ESS101 Mod & Sim
SP1 2019
Assignment - Implicit integrators
Due date: 2019-10-25

In this assignment we will investigate some widely-used implicit 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. **Implicit RK1** Consider the implicit ODE:

$$F(\dot{x}, x, u) = 0 \tag{1}$$

and an integration approach based on

Algorithm: Integration of implicit ODE

Input: x_0, $u(t_0), \ldots, u(t_{N-1})$ and Δt
for $k = 0 : N-1$ **do**
  Solve

$$F\left(\frac{x_{k+1} - x_k}{\Delta t}, x_k, u(t_k)\right) = 0 \tag{2}$$

  for x_{k+1}
return $x_{0,\ldots N}$

   (a) Show that this approach is in fact an Explicit Euler scheme.

   (b) Very small modifications are required in the algorithm above to make it an implicit Euler scheme. Specify what has to be done?

2. **Gauss-Legendre IRK** We will now deploy a Gauss-Legendre collocation method. This is an IRK scheme with $s = 2$ stages and of order $2s = 4$, having the Butcher Tableau provided below.

Table 1: IRK4, Gauss-Legendre Collocation

$$\begin{array}{c|cc} \frac{1}{2}-\frac{\sqrt{3}}{6} & \frac{1}{4} & \frac{1}{4}-\frac{\sqrt{3}}{6} \\ \frac{1}{2}+\frac{\sqrt{3}}{6} & \frac{1}{4}+\frac{\sqrt{3}}{6} & \frac{1}{4} \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

(a) Write a Matlab code that deploys the IRK scheme on dynamics of the form:

$$\dot{\mathrm{x}} = \mathrm{f}(\mathrm{x}, t) \tag{6}$$

i.e. we will omit inputs for the sake of simplicity.

Hint: this task can be fairly simple. Start with writing a code that formulates $\mathrm{r}(\mathrm{K}, \mathrm{x}_k)$ *from symbolic data* A, b, c *(where A is a matrix, and b and c are vectors). Then export a Matlab function that evaluates* r *and* $\frac{\partial \mathrm{r}}{\partial \mathrm{K}}$*. In order to make your life easy, it is arguably best to declare your* $\mathrm{K}_{1,\ldots,s}$ *as a matrix (in* $\mathbb{R}^{n\times s}$*), and use the "reshape" command to cast them as a vector. Your main code will then simply have a Newton iteration (to solve* $\mathrm{r} = 0$ *for* $\mathrm{K}_{1,\ldots,s}$ *at each integrator step) nested in a for loop that runs the simulation.*

(b) Test your code vs. the explicit RK4 scheme you have developed in the previous assignment on the test system

$$\dot{x} = \lambda x \tag{7}$$

(c) Deploy your IRK4 and explicit RK4 on the Van Der Pol dynamics:

$$\dot{x} = y \tag{8a}$$

$$\dot{y} = u\left(1 - x^2\right) y - x \tag{8b}$$

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

3. **DAE integration** in this task we will convert your IRK4 integrator into an integrator for fully-implicit DAEs:

$$\mathrm{F}(\dot{\mathrm{x}}, \mathrm{x}, t) = 0 \tag{9}$$

for the sake of simplicity we will consider that the dynamics have no input. You will need to modify your code slightly to accommodate the algebraic variable z and the implicit dynamics.

(a) Test your code on the 3D pendulum with the constraint

$$C(\mathrm{q}) = \frac{1}{2}\left(\mathrm{p}^\top \mathrm{p} - L^2\right) = 0 \tag{10}$$

in the constrained Lagrange formalism. The model equations read as:

$$\dot{\mathrm{p}} = \mathrm{v} \tag{11a}$$

$$m\dot{\mathrm{v}} = -mg\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} - z\mathrm{p} \tag{11b}$$

$$0 = \mathrm{p}^\top \dot{\mathrm{v}} + \mathrm{v}^\top \mathrm{v} \tag{11c}$$

Put the dynamics in a state-space form first, using e.g.

$$\mathrm{x} = \begin{bmatrix} \mathrm{p} \\ \mathrm{v} \end{bmatrix} \tag{12}$$

and deploy your IRK4 scheme for DAEs on the problem. Try e.g. the consistent initial conditions:

$$x = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \in \mathbb{R}^6 \tag{13}$$

and e.g. $t_\mathrm{f} = 2$. Try different Δt. What do you observe? Monitor the value of the DAE constraint (10) in your simulation. What happens? Why? How can we fix it?

(b) Test your code on the 3D pendulum model one obtains directly from Lagrange, i.e.

$$\dot{\mathrm{p}} = \mathrm{v} \tag{14a}$$

$$m\dot{\mathrm{v}} = -mg \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} - z\mathrm{p} \tag{14b}$$

$$0 = \frac{1}{2}\left(\mathrm{p}^\top \mathrm{p} - L^2\right) \tag{14c}$$

What happens? Why?