# Project A: Numerical scheme for time discretization

Morteza Namvar

October 14, 2021

## 1 Lobatto quadrature

## 2 Computation of three-dimensional streamlines

There are many different methods that can be used to approximate solutions to a differential equation and in fact whole classes can be taught just dealing with the various methods. In this assignment some numerical scheme for time discretization are studied including: Forward Euler, second order Runge-Kutta, fourth order Runge-Kutta, and two-step Adams-Bashforth scheme. After reviewing these methods in this assignment, they are implemented and the implementation is verified by solving an equation with given analytical solution.

### 2.1 Forward Euler scheme

Euler scheme is one of the oldest and easiest to use to approximate solutions to a differential equation. This method was originally devised by Euler. Suppose there is a deferential equation with the initial condition as shown in Eq. 1:

$$\frac{dy}{dt} = f\left(t, y\right) \qquad y\left(t_0\right) = y_0 \tag{1}$$

where $f(t, y)$ is a known function and the initial condition are also known values.

Generally, if we have $t_n$ and the approximation to the solution at this point, $y_n$, and we want to find the approximation at $t_{n+1}$ it is enough just to use the following equation:

$$y_{n+1} = y_n + f\left(t_n, y_n\right) \cdot \left(t_{n+1} - t_n\right) \tag{2}$$

Because of the simplicity in the implementation level, in most cases the discretization in time is done by using an equal time interval. So, $t_0$, $t_1$, $t_2$,... are of a uniform size of $h$. In other words, it is assumed that

$$t_{n+1} - t_n = h \tag{3}$$

Hence, if we do the formula for the next approximation becomes:

$$y_{n+1} = y_n + h\, f_n \tag{4}$$

### 2.2 Second Order Runge-Kutta scheme

They were first studied by Carle Runge and Martin Kutta around 1900 [1]. Modern developments are mostly due to John Butcher in the 1960s [2]. Runge-Kutta methods are single-step methods, however, with multiple stages per step.

The formulation of two step Runge-Kutta scheme is as follows [2]:

$$\begin{cases} y_{n+1} = y_n + h(\frac{1}{2}k_1 + \frac{1}{2}k_2) \\ k_1 = f(t_n, y_n) \\ k_2 = f(t_n + h, y_n + hk_1) \end{cases} \tag{5}$$

### 2.3 Fourth Order Runge-Kutta scheme

The formulation of four step Runge-Kutta method is as following:

$$\begin{cases} k_1 = hf(t_0, y_0) \\ k_2 = hf(t_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_1) \\ k_3 = hf(t_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_2) \\ k_4 = hf(t_0 + h, y_0 + k_3) \\ y_{n+1} = y_0 + (\frac{1}{6}k_1 + \frac{1}{3}k_2 + \frac{1}{3}k_3 + \frac{1}{6}k_4) \end{cases} \tag{6}$$

### 2.4 Two-step Adams-Bashforth

A simple multistep method is the two-step Adams–Bashforth method. Based on Eq. 7, this method needs two values of $y_{n+1}$ and $y_n$ to compute the next value of $y_{n+2}$:

$$y_{n+2} = y_{n+1} + \frac{3}{2}hf(t_{n+1}, y_{n+1}) - \frac{1}{2}hf(t_n, y_n) \tag{7}$$

The initial condition provides only one value of $y_n$. But, the value of $y_{n+1}$ is unknown also. A possibility to resolve this issue is to use the Euler's method to compute the $y_{n+1}$. So, the Adams-Bashforth scheme is split in two steps. So, in the first step instead of using $h$ as the time step, the value of $\frac{h}{2}$ is used.

*First step*: Using forward Euler scheme with time step of length $\frac{h}{2}$. At this step the values at $t_n$ is given as $y_n$. So, it is possible to calculate the $f(t_n, y_n)$. Then this value is used as the initial condition of Euler method to calculate the values at $t_{n+1}$. The values obtained will be used as "predictor" in the second step.

*Second step*: In the second step, a time steps of length $h$ is using to perform the Adams-Bashforth method. At this step all the right-hand side values of Eq. 7 are known.

### 2.5 The case study of this exercise

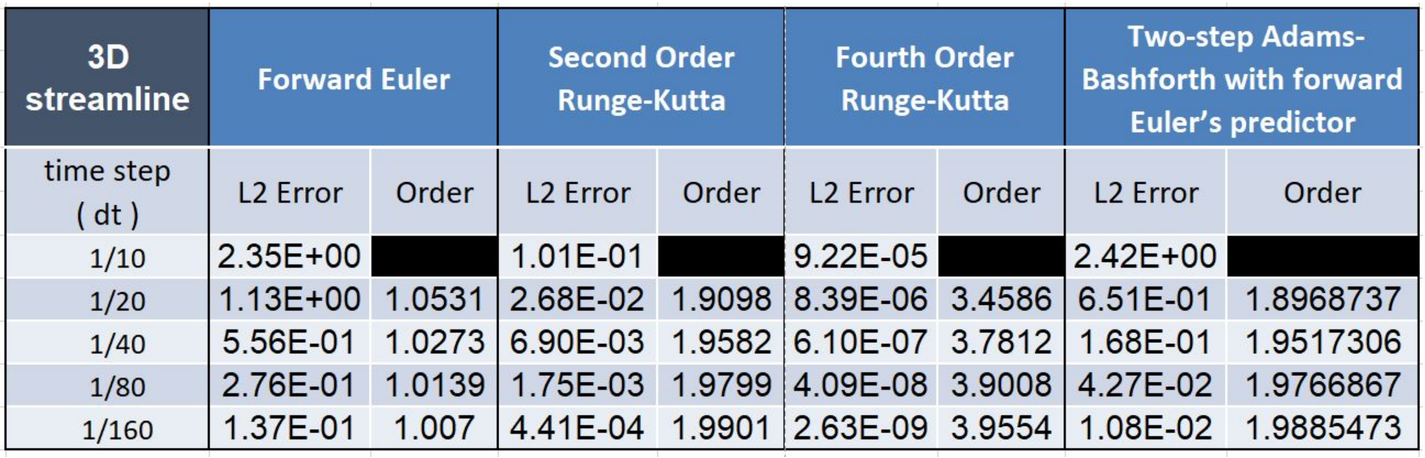
This exercise consists in programming different ODE solvers and verifying their implementations. For simplicity, the analytical velocity field, $u(x, t) = [u_x(X, t), u_y(X, t), u_z(X, t)]$, is taken as follows:

$$\begin{cases} u_x(X, t) = xsin(\omega_x t)exp(-t) \\ u_y(X, t) = ysin(\omega_y t)exp(-t) \\ u_z(X, t) = zsin(\omega_z t)exp(-t) \end{cases} \tag{8}$$

where $\omega_x = \pi/3$, $\omega_y = 2\pi/3$, and $\omega_z = \pi$.

To verify the implementation of the above numerical solvers, the results are illustrated in Fig. 1. The steady state numerical positions $X^{sn}$ that were computed in the main time marching loop are compared to the steady state analytical positions $X^{sa}$. The L2 norm is used to evaluate the error:

$$\text{L2error} \equiv ||X^{sn} - X^{sa}||_{L^2} = \sqrt{\sum_i \left(x_i^{sn} - x_i^{sa}\right)^2 + \left(y_i^{sn} - y_i^{sa}\right)^2 + \left(z_i^{sn} - z_i^{sa}\right)^2} \tag{9}$$



| 3D streamline | Forward Euler | | Second Order Runge-Kutta | | Fourth Order Runge-Kutta | | Two-step Adams-Bashforth with forward Euler's predictor | |
|---|---|---|---|---|---|---|---|---|
| time step ( dt ) | L2 Error | Order | L2 Error | Order | L2 Error | Order | L2 Error | Order |
| 1/10 | 2.35E+00 | | 1.01E-01 | | 9.22E-05 | | 2.42E+00 | |
| 1/20 | 1.13E+00 | 1.0531 | 2.68E-02 | 1.9098 | 8.39E-06 | 3.4586 | 6.51E-01 | 1.8968737 |
| 1/40 | 5.56E-01 | 1.0273 | 6.90E-03 | 1.9582 | 6.10E-07 | 3.7812 | 1.68E-01 | 1.9517306 |
| 1/80 | 2.76E-01 | 1.0139 | 1.75E-03 | 1.9799 | 4.09E-08 | 3.9008 | 4.27E-02 | 1.9766867 |
| 1/160 | 1.37E-01 | 1.007 | 4.41E-04 | 1.9901 | 2.63E-09 | 3.9554 | 1.08E-02 | 1.9885473 |

Figure 1: Order of accuracy for different numerical schemes.

To study the order of accuracy, it is required to half the time steps for steady state solution and then if the accuracy increased by one order of magnitude the method is called first order. If the accuracy increased by two order of magnitude that scheme is called second order. As it is expected the accuracy of forward Euler is 1. So, by decreasing the time step values by half, the accuracy of results is increased by one order which proof the implementation.

For the Runge-Kutta scheme, the order of accuracy is related to the number of steps taking to solve the equation for one time step. Hence, the two step Runge-Kutta is second order and four step Runge-Kutta is fourth order. Based on the results shown in Fig. 1 it proofs this expectation.

## 3 Shooting Method

## References

[1] I. F. S. N. Hussain, Kasim, Runge-kutta type methods for directly solving special fourth-order ordinary differential equations, Mathematical Problems in Engineering 14 (3) (2015) 342–351.

[2] atozmath.com, https://atozmath.com/example/conm/rungekutta.aspx?he=e&q=rk4.