Roll No: EE18B067 Name:Abhishek Sekar

Date: December 12, 2021

1. [Numerical O.D.E]
Consider the non-linear ordinary differential equation

$$y^{'} + y(1-y) = 0$$

with y(0) = $\frac{1}{2}$.

- Write a code to solve the above to obtain y(1) using
  - Euler Explicit
  - Euler Implicit
  - Trapezoidal Rule
  - RK2
  - RK4

  For each of the above, identify for what time stepping the scheme is stable, and the overall global accuracy.

Solution:
The true value:
On solving the differential equation,

$$\begin{aligned}
&\frac{dy}{dt} = y(y-1)\\
&\frac{dy}{y(y-1)} = dt\\
&\int \frac{dy}{y(y-1)} = \int dt\\
&\int \frac{1}{y-1} - \frac{1}{y} dy = t + c\\
&\log\left|\frac{y-1}{y}\right| = t + c\\
&\log\frac{1-y}{y} = t + c && \text{(As y < 1)}
\end{aligned}$$

On substituting the initial conditions given, we find that c = 0 and y = $\frac{1}{e^t+1}$.
Therefore, the true value of y(1) is just $\frac{1}{1+e}$.
The procedure:
The various schemes were derived using $f(y,t) = -y(1-y)$.
For the sake of computing y(1), a time step dt of 0.01 was used for all the schemes.
The schemes are described below,
Euler Explicit:

$$y_e[i+1] = y_e[i] + dt \cdot (y_e[i] - 1) \cdot (y_e[i]) \tag{1}$$

Euler Implicit:
Here, we have

$$y_i[i+1] = y_i[i] + dt \cdot (y_e[i+1] - 1) \cdot (y_e[i+1])$$

On solving this quadratic for $y_i[i+1]$, we take the smaller root as $y_i[i+1] \approx y_i[i]$ for small dt.
Thus, we have,

$$y_i[i+1] = \frac{(1+dt) - \sqrt{((1+dt)^2 - 4(y_i[i])dt))}}{(2dt)} \tag{2}$$

Treading in a similar fashion, we have
Trapezoidal Rule:

$$y_t[i+1] = \frac{(1+(\frac{dt}{2})) - \sqrt{((1+(\frac{dt}{2}))^2 - 4(\frac{dt}{2})((\frac{dt}{2})y_t[i]^2 + (1-(\frac{dt}{2}))y_t[i])))}}{dt} \tag{3}$$

RK-2
Here we consider the variant with $\alpha = 0.5$

$$k_1 = dt(y_{rk2}[i]-1)(y_{rk2}[i]) \quad \text{(first intermediate point)} \tag{4}$$
$$k_2 = dt(y_{rk2}[i]-1+0.5k_1)(y_{rk2}[i]+0.5k_1) \quad \text{(second intermediate point)} \tag{5}$$
$$y_{rk2}[i+1] = y_{rk2}[i] + k_2 \tag{6}$$

RK-4

$$k_1 = dt(y_{rk4}[i]-1)(y_{rk4}[i]) \quad \text{(first intermediate point)} \tag{7}$$
$$k_2 = dt(y_{rk4}[i]-1+0.5k_1)(y_{rk4}[i]+0.5k_1) \quad \text{(second intermediate point)} \tag{8}$$
$$k_3 = dt(y_{rk4}[i]-1+0.5k_2)(y_{rk4}[i]+0.5k_2) \quad \text{(third intermediate point)} \tag{9}$$
$$k_4 = dt(y_{rk4}[i]-1+k_3)(y_{rk4}[i]+k_3) \quad \text{(fourth intermediate point)} \tag{10}$$
$$y_rk4[i+1] = y_{rk4}[i] + \frac{1}{6}k_1 + \frac{1}{3}\cdot(k_2+k_3) + \frac{1}{6}k_4 \tag{11}$$

Stability:
The implicit schemes (Euler implicit and Trapezoidal Rule) are unconditionally stable whereas the explicit methods are conditionally stable.
We see that for the task of computing y(1), all the schemes are stable.
However, for finding the stability of the schemes, we try experimenting with $y^{'} = \lambda y$. Finding the stability for Euler explicit, we see that it gets unstable as dt exceeds 3 and for the RK methods its extremely non trivial to find the stability.However issues arise at about dt = 4.
Plots

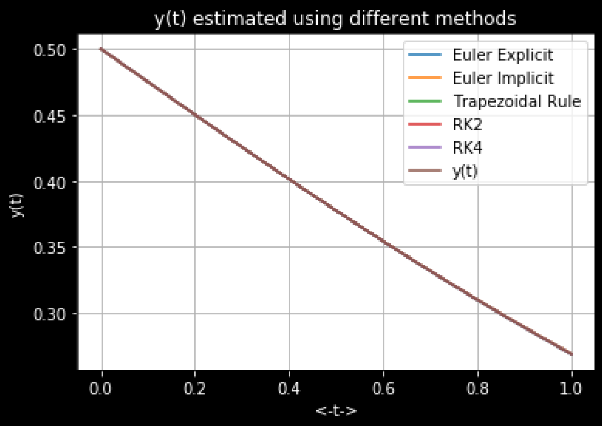

Figure 1: Actual plot of the function using the various schemes

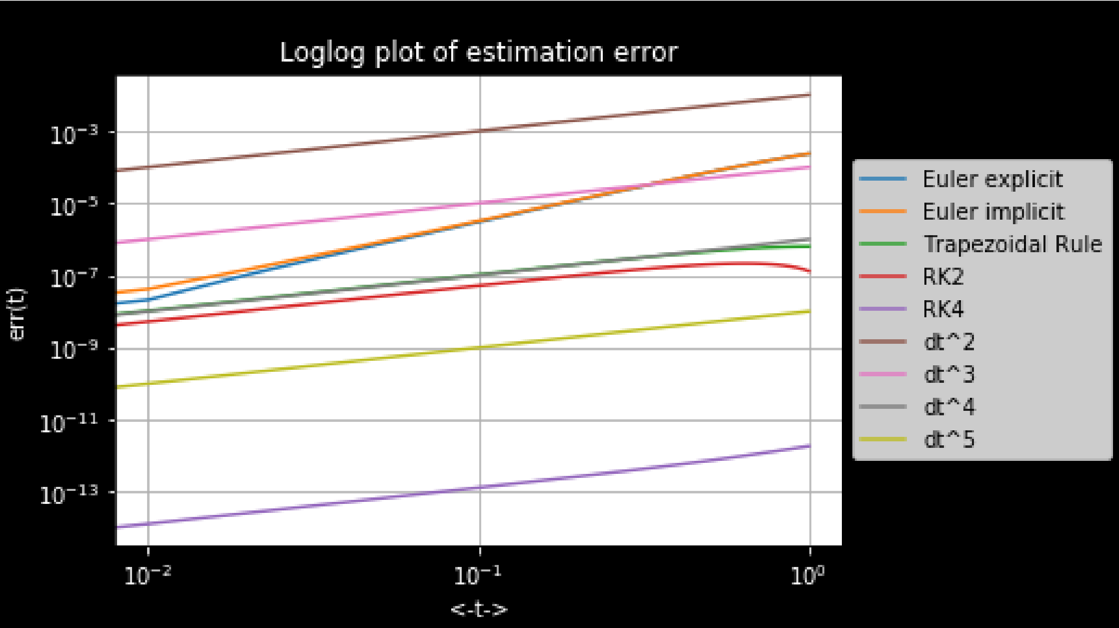

Figure 2: Plot of the error

Function values at y(1)

```
Actual value y(1) : 0.268941421369995
Euler explicit estimate y(1) : 0.2687054165811632
Euler implicit estimate y(1) : 0.26917773195118677
Trapezoidal Rule estimate y(1) : 0.2689420543813492
Second Order Runge Kutta estimate y(1) : 0.2689412904060045
Fourth Order Runge Kutta estimate y(1) : 0.26894142137184596
```

Observations

- We see that the order of accuracy of the schemes are RK4 > RK2 > Trapezoidal Rule > Euler explicit and implicit.
- From the plot of the errors we see that the local accuracy of the schemes agree with what we'd expect theoretically.
- For this particular time step we see that all the plots for y(t) overlap which talks about how good all the schemes are.

Global Accuracy
The overall global accuracy of the schemes are given below:

- Euler explicit and implicit :$\mathcal{O}(dt)$.
- Trapezoidal Rule : $\mathcal{O}((dt)^2)$.
- RK2 : $\mathcal{O}((dt)^2)$.
- RK4 : $\mathcal{O}((dt)^4)$.

2. [Shooting method]
Write a code that guarantees global fourth order accuracy to solve

$$y^{"} + 4y^{'} + 3y = \sin t^2$$

with y(0) = 0 and y(1) = 1 using the shooting method.

Solution:
We break the given second order ODE into two first order ones before employing the shooting method. We use an RK4 based approach for the shooting method as we require an overall 4^{th} order global accuracy.
The coupled system of differential equations representing the ODE is given below

$$u^{'} = v \tag{12}$$

$$v^{'} = \sin{(t^2)} - 3u - 4v \tag{13}$$

where u = y and v = $y^{'}$.
The shooting method tries approximating the initial value of v iteratively by making different guesses.
The solution obtained by the shooting method is shown below.
The exact solution was extremely non-trivial to obtain and hence can't be plotted accurately.
Plots

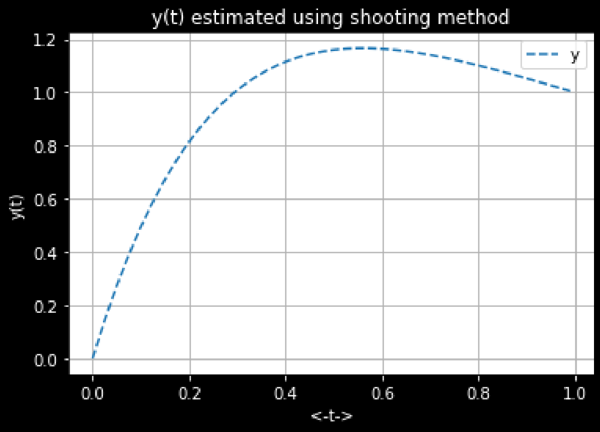

Figure 3: y(t)

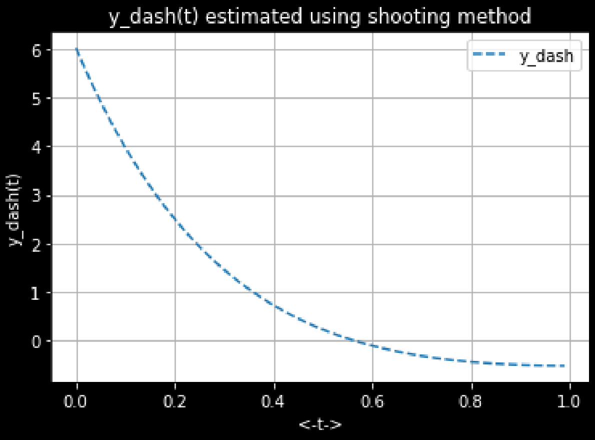

Figure 4: $y^{'}$(t)