# Runge-Kutta Methods to Solve ODEs

Instructor: Koel Das

Debayan Sarkar[1] Diptanuj Sarkar[2]

[1]Dept. of Physical Sciences
IISER Kolkata

[2]Dept. of Physical Sciences
IISER Kolkata

MA3105: Numerical Analysis Final Project

# Introduction

Ordinary Differential Equations (ODEs) are fundamental in modeling real-world phenomena across various fields of science and engineering. While analytical solutions are ideal, they are often unattainable for complex equations. Numerical methods, such as the Forward Euler Method and Runge-Kutta Methods, provide practical alternatives. This report focuses on the Runge-Kutta methods, specifically the second and fourth-order schemes, and compares them to the Forward Euler Method.

# Numerical Methods to Solve ODEs

In this project we have looked at three different ODE solvers. They have been listed below:

- Forward Euler Method
- Runge-Kutta Method of the Second Order (RK2)
- Runge-Kutta Method of the Fourth Order (RK4)

In the next slides we go over these methods in more details, beginning with the most basic one.

# Euler Method

This is the simplestway to solve ODEs numerically. This is a first order method. As in, the error in it grows as the square of the step size. The motivation for it has been shown below.

# Euler Method

Let $y(x)$ be a smooth function. Then, from its taylor expansion we have

$$y(x+h) = y(x) + y'(x)h + \sum_{n=2}^{\infty} y^{(n)}(x)h^n$$

$$\Rightarrow y(x+h) = y(x) + y'(x)h + \mathcal{O}(h^2)$$

$$\Rightarrow y(x+h) = y(x) + y'(x)h \qquad \text{(Ignoring the second order terms)}$$

This gives us the expression for iteratively calculating $y(x)$ using the forward Euler Method:

$$y(x+h) = y(x) + hf(x,y)$$

Denoting them in terms of $x_n$ and $y_n$ we get the following expression:

$$\boxed{y_{n+1} = y_n + hf(x_n, y_n)}$$

# Runge-Kutta Methods

Runge-Kutta methods are a family of iterative methods to solve ODEs which were first introduced in the 1900s. These methods improve upon the Forward Euler Method by considering intermediate slopes to achieve higher accuracy. The Runge-Kutta methods implemented here have been discussed in further detail below.

# Runge-Kutta Methods: Second Order (Midpoint Method)

The most commonly used second order Runge-Kutta method is the midpoint method, where we improve upon the Euler Method using the slope of the midpoint. The algorithm goes as follows. Given a step-size of $h > 0$, in each iteration we define the following:

$$k_1 = f(x_n, y_n)$$

$$k_2 = f\left(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_1\right)$$

Now using these values we calculate the next term in the iteration using the following expression:

$$y_{n+1} = y_n + hk_2$$

Substituting the value of $k_2$ in this expression we get:

$$\boxed{y_{n+1} = y_n + hf\left(x_n + \frac{h}{2}, y_n + \frac{h}{2}f(x_n, y_n)\right)}$$

# Runge-Kutta Methods: Second Order

However this is not the only second order Runge-Kutta Method for solving an IVP. Any second order Runge-Kutta Method can be parameterised by $\alpha$ and given by the formula:

$$\boxed{y_{n+1} = y_n + h\left[\left(1 - \frac{1}{2\alpha}\right) f(x_n, y_n) + \frac{1}{2\alpha} f(x_n + \alpha h, y_n + \alpha h f(x_n, y_n))\right]}$$

This formula has been derived in the next few slides.

# Runge-Kutta Methods: Second Order

First let us define the following terms:

$$
\begin{aligned}
k_1 &= hf(x_n, y_n) \\
k_2 &= hf(x_n + \alpha h, y_n + \beta k_1) \\
y_{n+1} &= y_n + ak_1 + bk_2
\end{aligned} \tag{1}
$$

Where, $a$, $b$, $\alpha$, $\beta$ are constants.

# Runge-Kutta Methods: Second Order

Now, consider the taylor series expansion of $y(x)$ upto the second order.

$$y(x+h) = y(x) + h\frac{\mathrm{d}y}{\mathrm{d}x} + \frac{h^2}{2}\frac{\mathrm{d}^2y}{\mathrm{d}x^2} + \mathcal{O}(h^3)$$

$$\Rightarrow y(x+h) = y(x) + hf(x,y) + \frac{h^2}{2}\left(\frac{\partial f}{\partial x} + f(x,y)\frac{\partial f}{\partial y}\right) + \mathcal{O}(h^3)$$

$$\Rightarrow y_{n+1} = y_n + hf(x_n,y_n) + \frac{h^2}{2}\left(\frac{\partial f}{\partial x} + f(x_n,y_n)\frac{\partial f}{\partial y}\right) + \mathcal{O}(h^3) \qquad (2)$$

# Runge-Kutta Methods: Second Order

We can also do a taylor series expansion for $k_2$ upto an error of $\mathcal{O}(h^3)$.

$$\begin{aligned} k_2 &= hf(x_n + \alpha h, y_n + \beta k_1) \\ &= h\left[f(x_n, y_n) + \frac{\partial f}{\partial x}\alpha h + \frac{\partial f}{\partial y}\beta k_1\right] + \mathcal{O}(h^3) \end{aligned}$$

# Runge-Kutta Methods: Second Order

We can also do a taylor series expansion for $k_2$ upto an error of $\mathcal{O}(h^3)$.

$$\begin{aligned} k_2 &= hf(x_n + \alpha h, y_n + \beta k_1) \\ &= h\left[f(x_n, y_n) + \frac{\partial f}{\partial x}\alpha h + \frac{\partial f}{\partial y}\beta k_1\right] + \mathcal{O}(h^3) \end{aligned}$$

Substituting this value of $k_2$ into (1) we get,

$$y_{n+1} = y_n + h(a+b)f(x_n, y_n) + bh^2\left[\alpha\frac{\partial f}{\partial x} + \beta f(x_n, y_n)\frac{\partial f}{\partial y}\right] + \mathcal{O}(h^3) \tag{3}$$

# Runge-Kutta Methods: Second Order

Comparing (2) and (3) and equating the coefficients we get the following equations.

$$\alpha b = \frac{1}{2} \Rightarrow b = \frac{1}{2\alpha}$$

$$\beta b = \frac{1}{2} \Rightarrow \beta = \alpha$$

$$a + b = 1 \Rightarrow a = 1 - \frac{1}{2\alpha}$$

# Runge-Kutta Methods: Second Order

Then (1) can be re-written parameterised only in terms of $\alpha$ as,

$$\boxed{y_{n+1} = y_n + h\left[\left(1 - \frac{1}{2\alpha}\right) f(x_n, y_n) + \frac{1}{2\alpha} f(x_n + \alpha h, y_n + \alpha h f(x_n, y_n))\right]}$$

As is clear from this derivation, this method is valid upto an error of the order of $\mathcal{O}(h^3)$.

# Runge-Kutta Methods: Fourth Order

The fourth order Runge-Kutta Method, also known as the classic Runge-Kutta Method is the most widely known member of the Runge-Kutta Family, and is known as 'RK4'. The algorithm goes as follows. Given a step-size of $h > 0$, in each iteration we define the following:

$$
\begin{aligned}
k_1 &= f(x_n, y_n) \\
k_2 &= f\left(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_1\right) \\
k_3 &= f\left(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_2\right) \\
k_4 &= f\left(x_n + h, y_n + hk_3\right)
\end{aligned}
$$

# Runge-Kutta Methods: Fourth Order

Now, using these intermediate slopes, we calculate the next term in the iteration using the following expression:

$$\boxed{y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)}$$

A derivation similar to that of RK2 can be done for this as well and it can be shown that RK4 is a fourth order method i.e. the error in the numerically obtained solution from the analytical solution is of the order of $\mathcal{O}(h^5)$.

# Given ODEs

We were asked to solve the following differential equations using the aforementioned ODE solvers.

$$\frac{dy}{dx} = y - x$$

$$\frac{dy}{dx} = y - x^2$$

The initial values provided for both these equations is $y(0) = \frac{2}{3}$

# Direction Fields

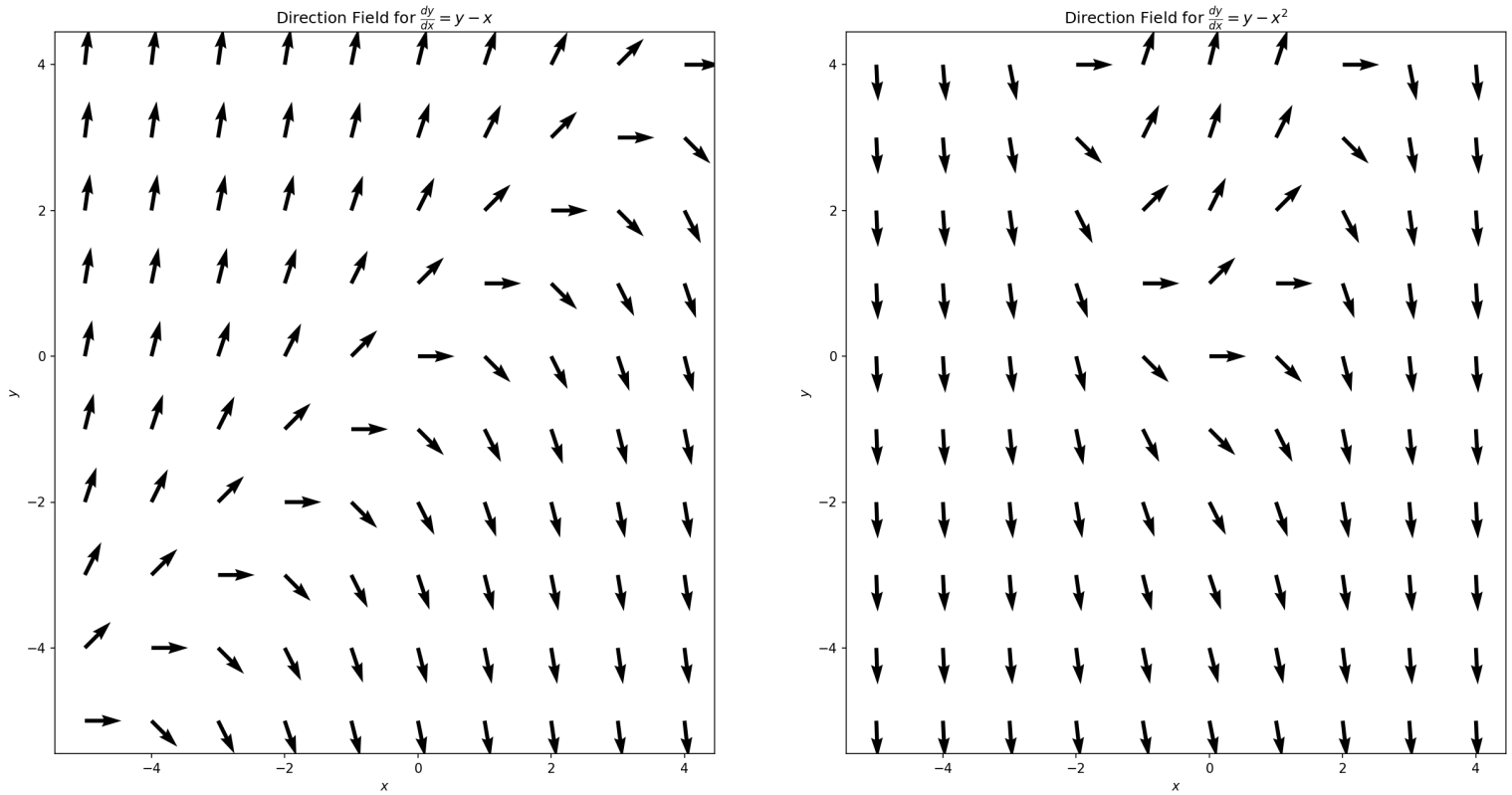


Figure: Direction Fields for the given ODEs

# Analytical Solutions

The analytical solutions to these equations can be computed by converting these into exact differentials using the integrating factor $\mu(x) = e^{-x}$. Solving these equations with the intial condition $y(0)$ gives us the solutions

$$\boxed{y_1(x) = 1 + x + (y(0) - 1)e^x}$$

$$\boxed{y_2(x) = 2 + 2x + x^2 + (y(0) - 2)e^x}$$

# Solving the ODEs

The given IVPs were solved using the priorly mentioned numerical methods, implemented in python. A plot of the solutions obtained on the interval $x \in [0, 5]$ has been included below.

# Solving the ODEs

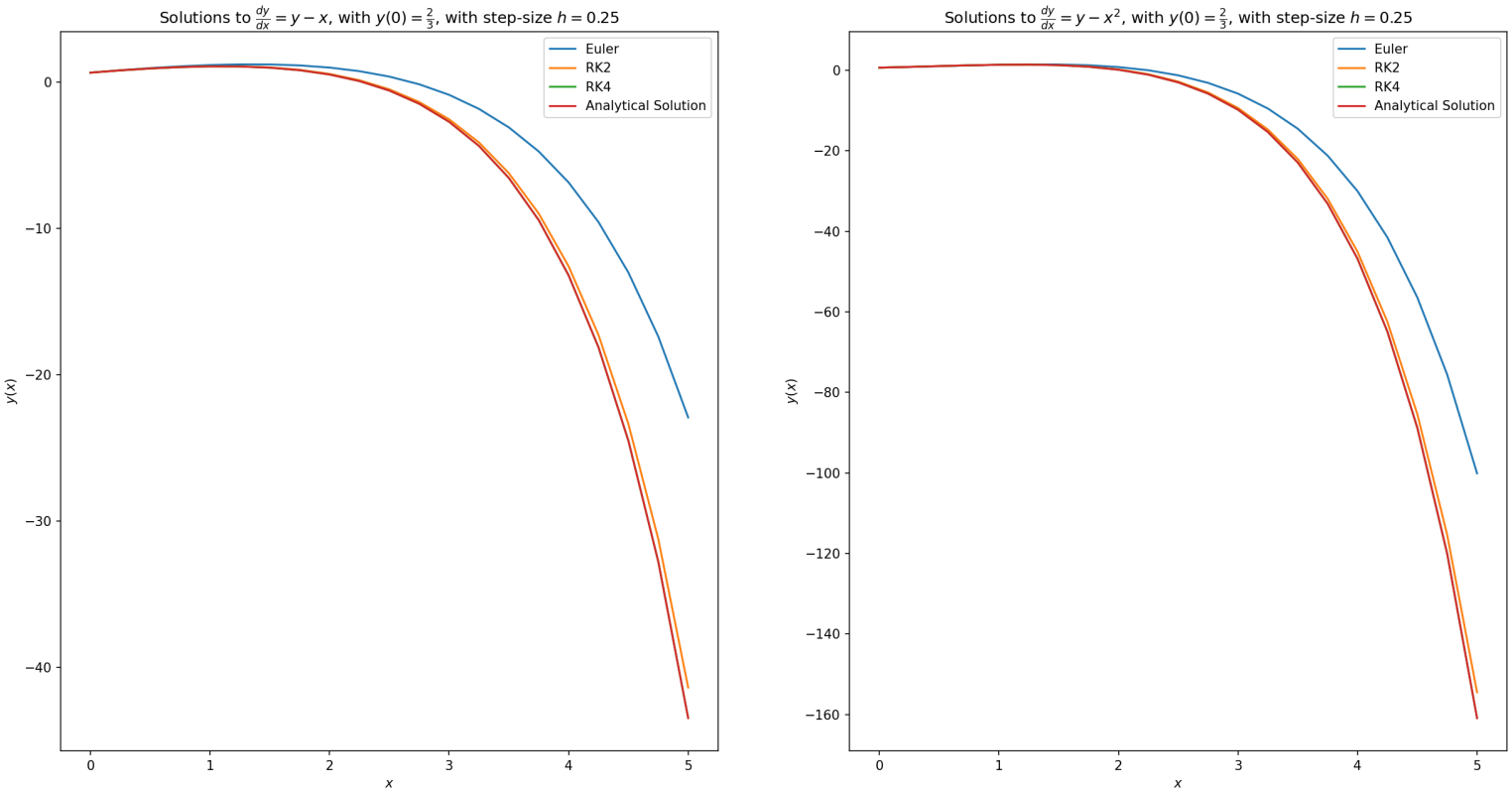


Figure: Plot of the solutions of the ODEs using the priorly mentioned numerical methods

# Solving the ODEs: Comparision and Analysis

Looking at the plots, we can compare the methods and conclude the following:

**Forward Euler Method:** Produced noticeable deviations from the exact solution, and as the value of $x$ grew, the deviation became larger.

**RK2 Method:** Significantly reduced errors compared to Forward Euler. The inclusion of intermediate slopes provided better stability and accuracy.

**RK4 Method:** Delivered solutions nearly indistinguishable from the exact solution. The higher accuracy of $\mathcal{O}(h^5)$ made RK4 the most reliable among the three methods for the given ODEs.

# Solving the ODEs: Varying the Initial Values

We varied the initial value $y(0)$ and plotted the analytical solution and RK4 solutions to see how they compare. The step-size taken for the solutions was $h = 0.025$.

# Solving the ODEs: Varying the Initial Values

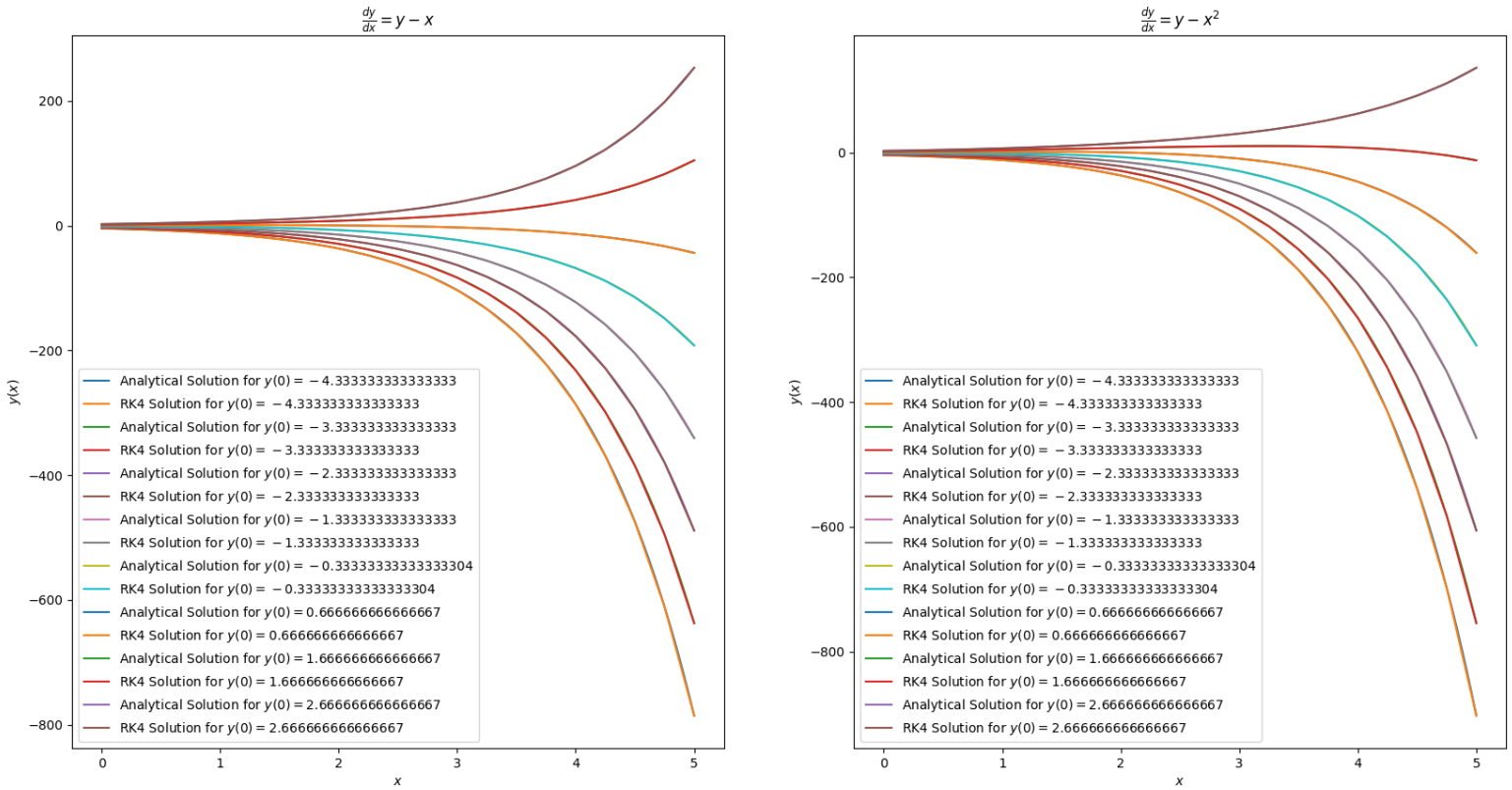


Figure: Plots of the Solutions of the given Differential Equations with varying values of $y(0)$