Runge Kutta Method

IRUNGU MUNGAI

March 2023

4th order RUNGE-KUTTA

- It finds an approximaation of y for a given x.
- It works by appoximating the solution of an ODE at discrete step using weighted average of function evaluations at multiple intermediate points between those time steps.
- The RK4 is the most common variant method due to its high accuracy and computational efficiency.

4th order RUNGE-KUTTA

- It finds an approximaation of y for a given x.
- It works by appoximating the solution of an ODE at discrete step using weighted average of function evaluations at multiple intermediate points between those time steps.
- The RK4 is the most common variant method due to its high accuracy and computational efficiency.

4th order RUNGE-KUTTA

- It finds an approximaation of y for a given x.
- It works by appoximating the solution of an ODE at discrete step using weighted average of function evaluations at multiple intermediate points between those time steps.
- The RK4 is the most common variant method due to its high accuracy and computational efficiency.

General Form

$$y_{n+1} = y_n + h * (k_1 + 2 * K_2 + 2 * k_3 + k_4)$$

- The values of K_i are calculated as follows:

The value of K_1

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

The value of K_2

$$k_2 = f(t_n + (h/2), y_n + (h/2) * k_1)$$

General Form

$$y_{n+1} = y_n + h * (k_1 + 2 * K_2 + 2 * k_3 + k_4)$$

- The values of K_i are calculated as follows:

The value of K_1

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

The value of K_2

$$k_2 = f(t_n + (h/2), y_n + (h/2) * k_1)$$

General Form

The value of K_3

$$k_3 = f(t_n + (h/2), y_n + (h/2) * k_1)$$

The value of K_4

$$k = f(t_n + (h/2), y_n + (h/2) * k_1)$$

- f() is the function that defines the ODE

General Form

The value of K_3

$$k_3 = f(t_n + (h/2), y_n + (h/2) * k_1)$$

The value of K_4

$$k = f(t_n + (h/2), y_n + (h/2) * k_1)$$

- f() is the function that defines the ODE

PseudoCode

- input initial values:t0,y0,h,T
- Set $t_n = t0 and y_n = y0$

PseudoCode

- input initial values:t0,y0,h,T
- Set $t_n = t0andy_n = y0$

PseudoCode

- While $t_n < T$:

Calculate K1:

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

Calculate K2:

$$k_2 = f(t_n + (h/2), y_n + (h/2) * k_1)$$

Calculate K3:

$$k_3 = f(t_n + (h/2), y_n + (h/2) * k_1)$$

Calculate K4:

$$k = f(t_n + (h/2), y_n + (h/2) * k_1)$$

PseudoCode

- While $t_n < T$:

Calculate K1:

$k_1 = f(t_n, y_n)$

Calculate K2:

$k_2 = f(t_n + (h/2), y_n + (h/2) * k_1)$

Calculate K3:

$k_3 = f(t_n + (h/2), y_n + (h/2) * k_1)$

Calculate K4:

$k = f(t_n + (h/2), y_n + (h/2) * k_1)$

PseudoCode

- While $t_n < T$:

Calculate K1:

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

Calculate K2:

$$k_2 = f(t_n + (h/2), y_n + (h/2) * k_1)$$

Calculate K3:

$$k_3 = f(t_n + (h/2), y_n + (h/2) * k_1)$$

Calculate K4:

$$k = f(t_n + (h/2), y_n + (h/2) * k_1)$$

PseudoCode

- While $t_n < T$:

Calculate K1:

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

Calculate K2:

$$k_2 = f(t_n + (h/2), y_n + (h/2) * k_1)$$

Calculate K3:

$$k_3 = f(t_n + (h/2), y_n + (h/2) * k_1)$$

Calculate K4:

$$k = f(t_n + (h/2), y_n + (h/2) * k_1)$$

PseudCode

Calculate: y_{n+1}

$$y_{n+1} = y_n + (h/6) * (k_1 + 2 * K_2 + 2 * k_3 + k_4)$$

Set new t_n

$$t_n = t_n + h$$

Output Final Value of y(t)

Output final value of y(t)

PseudCode

Calculate: y_{n+1}

$$y_{n+1} = y_n + (h/6) * (k_1 + 2 * K_2 + 2 * k_3 + k_4)$$

Set new t_n

$$t_n = t_n + h$$

Output Final Value of y(t)

Output final value of y(t)

PseudCode

Calculate: y_{n+1}

$$y_{n+1} = y_n + (h/6) * (k_1 + 2 * K_2 + 2 * k_3 + k_4)$$

Set new t_n

$$t_n = t_n + h$$

Output Final Value of y(t)

Output final value of y(t)