Runge-Kuttah Method Presentation

Using Beamer

EDWIN OWINO, ALLAN VINCENT, NELSON IRUNGU

JKUAT

April 1, 2023

Introduction

Runge-Kuttah Method is a series of methods used to solve differential equations with more accuracy without performing many calculations. It assumes the following solutions:

- 1st Order (Eulers Method)
- 2nd Order Runge-Kuttah Method
- 4th Order Runge-Kuttah Method (RK4)

1st Order Runge-Kuttah Method

It uses the following formular:

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

To construct the tangent at the point x and obtain the value of $y(x+h)$ whose slope is:

$$f(x,y) \ or \ simply, \frac{dy}{dx}$$

General formula: $y_{i+1} = y_i + hf(x_i, y_i)$

NB

This is the easiest of all the methods but has an error margin

2nd Order Runge-Kuttah Method

The Runge-Kutta method finds an approximate value of y for a given x. Only first-order ordinary differential equations can be solved by using the Runge Kutta 2nd order method.

Below is the formula used to compute next value: y_{n+1} from previous value
y_n: $y_{n+1} =$ value of y at $(x = n + 1)$
$y_n =$ value of y at $(x = n)$ where:
$0n(x - x_0)/h$; h is step height
$x_{n+1} = x_0 + h$

cont...

The essential formula to compute the value of $y(n+1)$:

$$K_1 = h * f(x_n, y_n)$$

$$\mathrm{K}_2 = h * f((x_n + \frac{h}{2}), (y_n + \frac{K_1 * h}{2}))$$
$$\mathrm{y}_{n+1} = y_n + K_2 + (h^3)$$

2nd order Continuation

- The formula basically computes the next value y_{n+1} using current yn plus the weighted average of two increments:
- K1 is the increment based on the slope at the beginning of the interval, using y.
- K2 is the increment based on the slope at the midpoint of the interval, using $(y + h * K1/2)$.
- The method is a second-order method, meaning that the local truncation error is on the order of $O(h3)$, while the total accumulated error is order $O(h4)$.

4th Order Runge-Kuttah Method

To be presented by Nelson

TIP

4th Order Runge-Kuttah method also called RK4 is the most important in the series

4th order RUNGE-KUTTA

- It finds an approximation of y for a given x.
- It works by aproximating 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 approximation of y for a given x.
- It works by aproximating 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 approximation of y for a given x.
- It works by aproximating 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 $t_n = t0andy_n = y0$

PseudoCode

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

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:

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

PseudoCode

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)

PseudoCode

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)

PseudoCode

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)