The 4th -order Runge-Kutta method for a single ODE

By Gilberto E. Urroz, Ph.D., P.E.
January 2010

Problem description

This worksheet provides two different versions of the 4th-order Runge-Kutta method for solving a first-order ordinary differential equation (ODE) of the form:

$$\frac{d}{dx} y = f(x, y)$$

subject to the initial condition: $y(xs) = ys$

The solution will be provided in the range: $xs \le x \le xe$ which will be divided into n subintervals to produce solution vectors "xsol" and "ysol", as detailed below.

The increment in the x solution vector, Δx,is calculated as:

$$\Delta x = \frac{xe - xs}{n}$$

Thus, the solution vector will be calculated using the uniformly-distributed values: [xs,xs+Δx,xs+2*Δx,...,xs+n*Δx].Using SMath Studio the x solution vector can be easily created using the command:

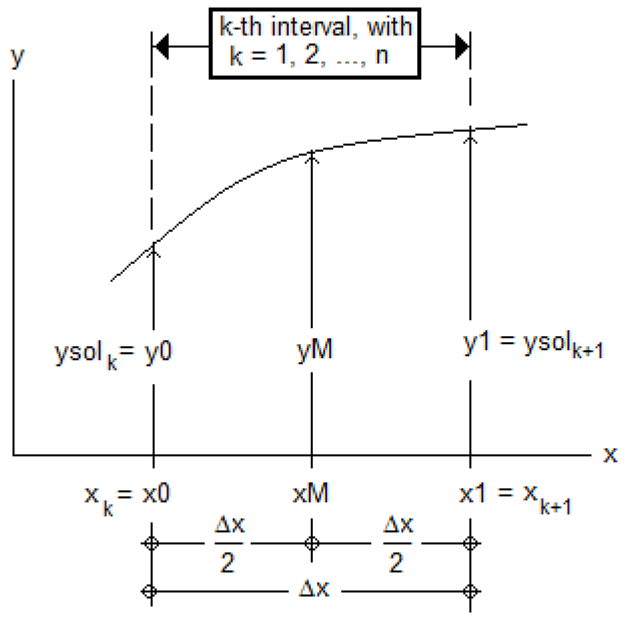
```
xsol: range(xs,xe,xs+Δx)
```

The y solution vector will be calculated using the Runge-Kutta algorithm, which is detailed below. The first value in the y solution will be the initial condition ys, i.e.,

$$ysol_1 = ys$$

Iterative process for solution - version 1

Next, we start an iterative procedure with the index k varying between 1 and n, i.e., k = 1, 2, ..., n. To illustrate the calculations involved, we illustrate the k-th interval in the following figure:



In this figure the lower x limit of the interval is x[k], which is represented by x0 in the calculations. On the other hand, the upper x limit of the interval is x[k+1], which is represented by x1. The corresponding values of the solution are y0 = y(x0) and y1 = y(x1). The Runge-Kutta algorithm uses the mid-point (xM,yM) illustrated also in the figure.

Runge-Kutta calculations - version 1

The calculations involved in the 4th-order Runge-Kutta algorithm to calculate y1 = ysol[k+1] are listed below:

$$xM = x0 + \frac{1}{2}\cdot\Delta x \qquad K1 = \Delta x\cdot f(x0, y0)$$

$$yM = y0 + \frac{1}{2}\cdot K1 \qquad K2 = \Delta x\cdot f(xM, yM)$$

$$yM = y0 + \frac{1}{2}\cdot K2 \qquad K3 = \Delta x\cdot f(xM, yM)$$

$$y1 = y0 + K3 \qquad K4 = \Delta x\cdot f(x1, y1)$$

$$ysol_{k+1} = y0 + \frac{1}{6}\cdot(K1 + 2\cdot K2 + 2\cdot K3 + K4)$$

Solution summary and graph

Once the iterative process has been completed, the solution will be contained in the column vectors "xsol" and "ysol". At this point, the user may decide to show the solution as vectors, or put them together into a matrix, say,

$$M = \text{augment}(xsol, ysol)$$

This matrix can be stored in a file, or plotted in a 2D graph to show the solution graphically.

Example of 4th-order Runge-Kutta solution -version 1

Solve the ODE: $\frac{dy}{dx} = \sin(x)$, subject to the initial condition: $y(0) = 1$ in the interval: $0 \le x \le 20$ using 100 intervals in the solution.

Solution:

First, define the function f(x,y): $f(x, y) := \sin(x) + \cos(y)$

The initial conditions are: $xs := 0$ $ys := 1$

The end of the solution interval is: $xe := 20$

Use 100 intervals: $n := 100$

Calculate the increment size, Δx:

$\Delta x := \text{eval}\left(\frac{xe - xs}{n}\right)$ $\Delta x = 0.2$

Create the x solution vector: $xsol := \text{eval}(xs, xs + \Delta x .. xe)$

Create the y solution vector as a zero matrix of n rows and 1 column, and initialize the y solution vector with the initial condition ys:

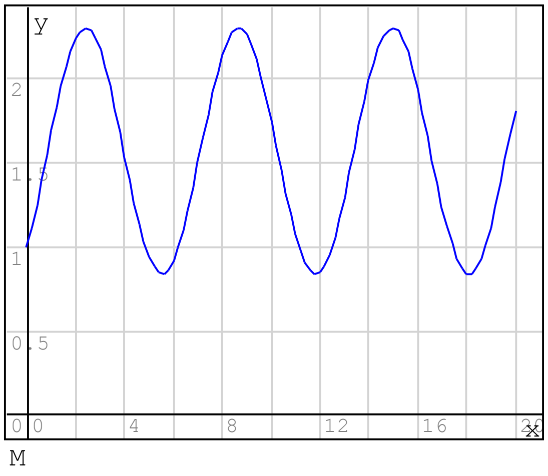
$ysol := \text{matrix}(n, 1)$ $ysol_1 := ys$

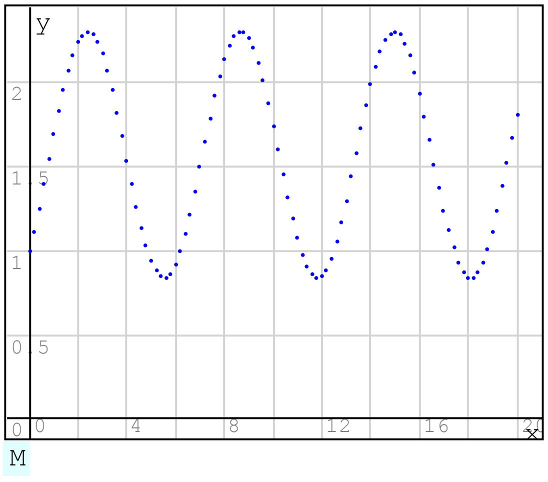
The following "for" loop calculates the Runge-Kutta algorithm (version 1) to produce the solution:

```
for $k \in 1 .. n$
    $x0 := \text{eval}(xsol_k)$
    $y0 := \text{eval}(ysol_k)$
    $xM := \text{eval}\left(x0 + \frac{1}{2} \cdot \Delta x\right)$
    $K1 := \text{eval}(\Delta x \cdot f(x0, y0))$
    $yM := \text{eval}\left(y0 + \frac{1}{2} \cdot K1\right)$
    $K2 := \text{eval}(\Delta x \cdot f(xM, yM))$
    $yM := \text{eval}\left(y0 + \frac{1}{2} \cdot K2\right)$
    $K3 := \text{eval}(\Delta x \cdot f(xM, yM))$
    $y1 := \text{eval}(y0 + K3)$
    $x1 := \text{eval}(xsol_{k+1})$
    $K4 := \text{eval}(\Delta x \cdot f(x1, y1))$
    $ysol_{k+1} := \text{eval}\left(y0 + \frac{1}{6} \cdot (K1 + 2 \cdot K2 + 2 \cdot K3 + K4)\right)$
```

The solution is summarized into matrix M and shown as a x-y plot:

$M := \text{augment}(xsol, ysol)$

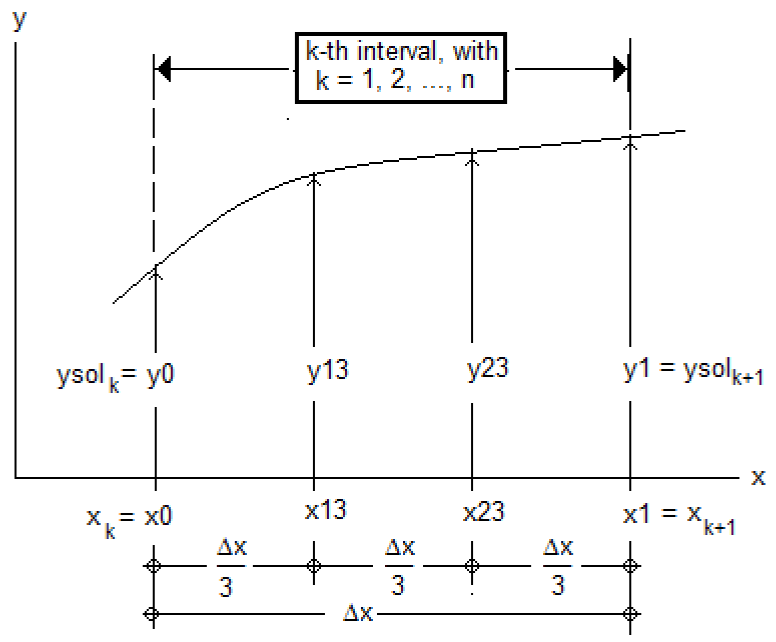




The plot to the left uses the "Graph by lines" option in the "Plot" palette, while the plot to the right uses the "Graph by points" option in the "Plot" palette.

Iterative process for solution - version 2

The second version of the Runge-Kutta solution divides the k-th interval into three equal parts are illustrated in the figure below:



In this figure the lower x limit of the interval is x[k], which is represented by x0 in the calculations. On the other hand, the upper x limit of the interval is x[k+1], which is represented by x1. The corresponding values of the solution are y0 = y(x0) and y1 = y(x1). The Runge-Kutta algorithm uses two mid-points (x13,y13) and (x23, y23) illustrated also in the figure.

Runge-Kutta calculations - version 2

The calculations involved in the 4th-order Runge-Kutta algorithm to calculate y1 = ysol[k+1] are listed below:

$x13 = x0 + \frac{1}{3}\cdot \Delta x \quad x23 = x0 + \frac{2}{3}\cdot \Delta x \quad K1 = \Delta x\cdot f(x0, y0)$

$y13 = y0 + \frac{1}{3}\cdot K1 \quad K2 = \Delta x\cdot f(x13, y13)$

$y23 = y13 + \frac{1}{3}\cdot K2 \quad K3 = \Delta x\cdot f(x23, y23)$

$y1 = y0 + K1 - K2 + K3 \quad K4 = \Delta x\cdot f(x1, y1)$

$$ysol_{k+1} = y0 + \frac{1}{8}\cdot(K1 + 3\cdot K2 + 3\cdot K3 + K4)$$

Example of 4th-order Runge-Kutta solution -version 2

Solve the ODE: $\frac{dy}{dx} = \sin(x)$, subject to the initial condition: $y(0) = 1$ in the interval: $0 \le x \le 20$ using 100 intervals in the solution.

Solution:

First, define the function f(x,y): $f(x, y) := \sin(x) + \cos(y)$

The initial conditions are: $xs := 0$ $ys := 1$

The end of the solution interval is: $xe := 20$

Use 100 intervals: $n := 100$

Calculate the increment size, Δx:

$\Delta x := \text{eval}\left(\frac{xe - xs}{n}\right)$ $\Delta x = 0.2$

Create the x solution vector: $xsol := \text{eval}(xs, xs + \Delta x .. xe)$

Create the y solution vector as a zero matrix of n rows and 1 column, and initialize the y solution vector with the initial condition ys:

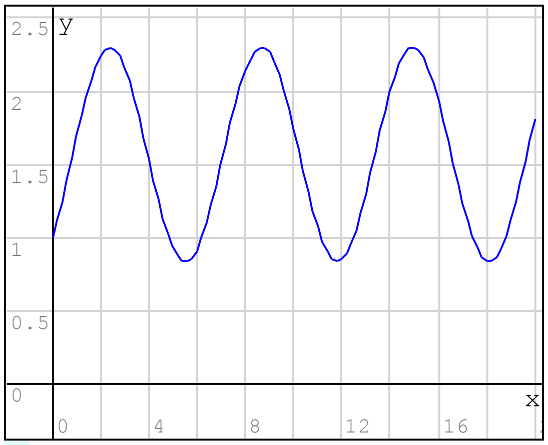
$ysol := \text{matrix}(n, 1)$ $ysol_1 := ys$

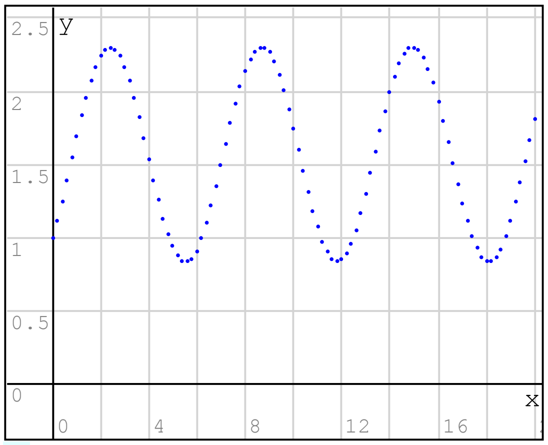
The following "for" loop calculates the Runge-Kutta algorithm (version 2) to produce the solution:

```
for k ∈ 1 .. n
    $x0 := \mathrm{eval}(xsol_k)$
    $y0 := \mathrm{eval}(ysol_k)$
    $x13 := \mathrm{eval}\left(x0 + \frac{1}{3}\cdot\Delta x\right)$
    $x23 := \mathrm{eval}\left(x0 + \frac{2}{3}\cdot\Delta x\right)$
    $K1 := \mathrm{eval}(\Delta x\cdot f(x0, y0))$
    $y13 := \mathrm{eval}\left(y0 + \frac{1}{3}\cdot K1\right)$
    $K2 := \mathrm{eval}(\Delta x\cdot f(x13, y13))$
    $y23 := \mathrm{eval}\left(y13 + \frac{1}{3}\cdot K2\right)$
    $K3 := \mathrm{eval}(\Delta x\cdot f(x23, y23))$
    $y1 := \mathrm{eval}(y0 + K1 - K2 + K3)$
    $x1 := \mathrm{eval}(xsol_{k+1})$
    $K4 := \mathrm{eval}(\Delta x\cdot f(x1, y1))$
    $ysol_{k+1} := \mathrm{eval}\left(y0 + \frac{1}{8}\cdot(K1 + 3\cdot K2 + 3\cdot K3 + K4)\right)$
```

The solution is summarized into matrix N and shown as a x-y plot:

$N := \mathrm{augment}(xsol, ysol)$





The plot to the left uses the "Graph by lines" option in the "Plot" palette, while the plot to the right uses the "Graph by points" option in the "Plot" palette.