

MEMORIAL UNIVERSITY OF NEWFOUNDLAND
FACULTY OF ENGINEERING AND APPLIED SCIENCE

Engineering: 5434 — Applied Mathematical Analysis

NUMERICAL SOLUTION OF ORDINARY DIFFERENTIAL EQUATIONS WITH INITIAL VALUES

In the following, w is a dummy name for the dependent variable y . It is used to underscore the fact that the numerical solution is never going to yield the actual values of y . The solution will give only an approximation to y which is designated by a different variable such as w .

Algorithm for Euler's Method

Given: $\frac{dy}{dx} = f(x, y) \quad x_0 \leq x \leq x_n \quad y(x_0) = y_0$

Required: Approximations w for values of y at $n+1$ equally spaced numbers in the interval $x = (x_0, x_n)$

Step 1 Set $h = (x_n - x_0) / n$; $x = x_0$; $w = y_0$

Print (x, w)

Step 2 For $i = 1, 2, 3, \dots, n$ do Steps 3 to 5.

Step 3 Set $K = hf(x, w)$

Step 4 Set $w = w + K$
 $x = x_0 + ih$

Step 5 Print (x, w)

Step 6 STOP.

The Euler's method is direct and simple but very inefficient since in most cases, an appropriate step size is prohibitively small. The modified Euler's method is an improvement since it uses a predicted slope at the required point and applies a correction to it based on the predicted value.

Algorithm for Modified Euler's (Huen) Method

Given: $\frac{dy}{dx} = f(x, y) \quad x_0 \leq x \leq x_n \quad y(x_0) = y_0$

Required: Approximations w for values of y at $n+1$ equally spaced numbers in the interval $x = (x_0, x_n)$

Step 1 Set $h = (x_n - x_0) / n$; $x = x_0$; $w = y_0$

Print (x, w)

Step 2 For $i = 1, 2, 3, \dots, n$ do Steps 3-5.

Step 3 Set $K_1 = hf(x, w)$; $K_2 = hf(x + h, w + K_1)$ *Predict*

Step 4 Set $w = w + (K_1 + K_2) / 2$ *Correct and compute w_i*
 $x = x_0 + ih$ *Compute x_i*

Step 5 Print (x, w)

Step 6 STOP.

For improved accuracy, the predicted slope K_2 can be re-corrected by setting $K_2 = hf(x+h, w+K_2)$ and can be re-re-corrected again if desired. However, it is not computationally very efficient to do so. Improved solution techniques are the Runge-Kutta methods of various orders. Modified Euler's method can be shown to be a special case of Runge-Kutta method of order two.

Algorithm for Runge-Kutta Method (Order Four)

Given: $\frac{dy}{dx} = f(x, y)$ $x_o \leq x \leq x_n$ $y(x_o) = y_o$

Required: Approximations w for values of y at $n+1$ equally spaced numbers in the interval $x = (x_o, x_n)$

Step 1 Set $h = (x_n - x_o) / n$; $x = x_o$; $w = y_o$

Print (x, w)

Step 2 For $i = 1, 2, 3, \dots, n$ do Steps 3 to 5.

Step 3 Set $K_1 = hf(x, w)$; $K_2 = hf(x+h/2, w+K_1/2)$
 $K_3 = hf(x+h/2, w+K_2/2)$; $K_4 = hf(x+h, w+K_3)$

Step 4 Set $w = w + (K_1 + 2K_2 + 2K_3 + K_4) / 6$ *Compute w_i*
 $x = x_o + ih$ *Compute x_i*

Step 5 Print (x, w)

Step 6 STOP.

The Runge-Kutta methods are efficient in several cases if an appropriate step size is chosen. One way to estimate if an appropriate step size has been chosen is to reduce the step size by 50% and recompute the solution at a given point. If the difference between the solutions for the two step sizes is within tolerable limits, the solution is acceptable at that point. This procedure can be repeated for successive steps. However, using this procedure in practice is computationally costly. Runge-Kutta-Fehlberg Method is adaptive to the step size requirements and is more efficient. It adjusts the step size at every point such that the solution adapts itself to the function.

Algorithm for Runge-Kutta-Fehlberg Method

Given: $\frac{dy}{dx} = f(x, y)$ $x_o \leq x \leq x_n$ $y(x_o) = y_o$

Tolerance TOL , Max. step size h_{max} , Min. step size h_{min} .

Required: Approximations w for values of y at several points in the interval $x = (x_o, x_n)$ where the difference in x -coordinates between successive points falls within the interval $h = (h_{min}, h_{max})$

Step 1 Set $h = h_{max}$; $x = x_o$; $w = y_o$; $FLAG = 1$

Print (x, w)

Step 2 While $FLAG = 1$ do steps 3 to 11.

- Step 3 Set $K_1 = hf(x, w)$; $K_2 = hf(x + h/4, w + K_1/4)$
 $K_3 = hf(x + 3h/8, w + 3K_1/32 + 9K_2/32)$
 $K_4 = hf(x + 12h/13, w + 1932K_1/2197 - 7200K_2/2197 + 7296K_3/2197)$
 $K_5 = hf(x + h, w + 439K_1/216 - 8K_2 + 3680K_3/513 - 845K_4/4104)$
 $K_6 = hf\left(x + h/2, w - 8K_1/27 + 2K_2 - 3544K_3/2565 + 1859K_4/4104 - \frac{11K_5}{40}\right)$
- Step 4 Set $R = \frac{|w + (K_1/360 - 128K_3/4275 - 2197K_4/75240 + K_5/50 + 2K_6/55)|}{h}$
- Step 5 Set $\delta = 0.84(TOL/R)^{1/4}$
- Step 6 If $R \leq TOL$ then do steps 7 and 8
- Step 7 Set $x = x + h$ *Acceptable step size - proceed*
 $w = (w + 25K_1/216 + 1408K_3/2565 + 2197K_4/4104 - K_5/5)$
- Step 8 Print (x, w, h)
- Step 9 If $\delta \leq 0.1$ then Set $h = 0.1h$
 else if $\delta \geq 4$ then set $h = 4h$
 else set $h = \delta h$ *Compute new step size*
- Step 10 If $h > h_{\max}$ then set $h = h_{\max}$
- Step 11 If $x \geq x_n$ then set $FLAG = 0$
 else if $x + h > x_n$ then set $h = x_n - x$
 else if $h < h_{\min}$ then set $FLAG = 0$ and
 Print 'The value of "hmin" is insufficient'
(Procedure terminates unsuccessfully since tolerance is violated) .
- Step 12 STOP. *(If STOP is reached safely, the procedure has been successful)*

All the above methods are single step methods, i.e., they use the current values to advance the solution by one step. A better estimate of the function can often be obtained if we use the results of several previous steps including the current step to advance to the next step. Procedures utilizing this idea are called multi-step methods. In such methods, the solution in the initial few steps is computed using any single step method of choice. After the first few steps, the multi-step method proceeds by utilizing the results computed thus far and advances the solution by one more step. One of the numerically stable multi-step methods is due to Adams-Bashforth-Moulton. In the following algorithm, the solution for step $i+1$ depends on the solution for the current step and three previous steps.

Algorithm for Adams Method (Using Runge-Kutta Fourth-Order)

Given: $\frac{dy}{dx} = f(x, y)$ $x_0 \leq x \leq x_n$ $y(x_0) = y_0$

Required: Approximations w for values of y at $n+1$ equally spaced numbers in the interval $x = (x_0, x_n)$

Step 1 Set $h = (x_n - x_0) / n$; $w_0 = y_0$

Print (x_o, w_o)

Step 2 For $i = 1, 2,$ and 3 do Steps 3 to 5. *Compute the starting values using Runge-Kutta*

Step 3 Set $K_1 = hf(x_{i-1}, w_{i-1})$; $K_2 = hf(x_{i-1} + h/2, w_{i-1} + K_1/2)$

$$K_3 = hf(x_{i-1} + h/2, w_{i-1} + K_2/2)$$

$$K_4 = hf(x_{i-1} + h, w_{i-1} + K_3)$$

Step 4 Set $w_i = w_{i-1} + (K_1 + 2K_2 + 2K_3 + K_4) / 6$

$$x_i = x_o + ih$$

Step 5 Print (x_i, w_i)

Step 6 For $i = 4, 5, \dots, n$ do steps 7 and 8.

Step 7 Set $x_i = x_o + ih$

$$w_i = w_{i-1} + h \left\{ \frac{55f(x_{i-1}, w_{i-1}) - 59f(x_{i-2}, w_{i-2}) + 37f(x_{i-3}, w_{i-3}) - 9f(x_{i-4}, w_{i-4})}{24} \right\} \quad \text{Predict } w_i$$

$$w_i = w_{i-1} + h \left\{ \frac{9f(x_i, w_i) + 19f(x_{i-1}, w_{i-1}) - 5f(x_{i-2}, w_{i-2}) + f(x_{i-3}, w_{i-3})}{24} \right\} \quad \text{Correct } w_i$$

Step 8 Print (x_i, w_i)

Step 9 STOP.

The above procedure stores all the values of x and w at all the intervals. It can easily be modified to limit the storage to 5 quantities each at any given point during the solution process