



GLOBAL JOURNAL OF RESEARCHES IN ENGINEERING: J
GENERAL ENGINEERING
Volume 14 Issue 7 Version 1.0 Year 2014
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 2249-4596 & Print ISSN: 0975-5861

An Algorithm for Integration, Differentiation and Finding Root Numerically

By N. Rahman, Aminul Islam, Pulakesh Gain, B. K. Datta & R. C. Bhowmik

Pabna University of Science & Technology (PUST), Bangladesh

Abstract- Numerical analysis concerns the development of algorithms for solving various types of problems of mathematics; it is a vast-ranging field having deep interaction with computer science, mathematics, engineering, and the sciences. Numerical analysis mainly consists of Numerical Integration, Numerical Differentiation and finding Roots numerically. In this paper we develop an algorithm combination of Numerical Integration (Trapezoidal rule, Simpson's **1/3** rule, Simpson's **3/8** rule and Weddle's rule.), Numerical Differentiation (Euler, modified Euler and Runge- Kutta second and fourth order) and finding Roots (Bisection method and False position method) numerically.

GJRE-J Classification : FOR Code: 091307, 010399



Strictly as per the compliance and regulations of:



An Algorithm for Integration, Differentiation and Finding Root Numerically

N. Rahman^α, Aminul Islam^σ, Pulakesh Gain^ρ, B. K. Datta^ω & R. C. Bhowmik[¥]

Abstract- Numerical analysis concerns the development of algorithms for solving various types of problems of mathematics; it is a vast-ranging field having deep interaction with computer science, mathematics, engineering, and the sciences. Numerical analysis mainly consists of Numerical Integration, Numerical Differentiation and finding Roots numerically. In this paper we develop an algorithm combination of Numerical Integration (Trapezoidal rule, Simpson's 1/3 rule, Simpson's 3/8 rule and Weddle's rule), Numerical Differentiation (Euler, modified Euler and Runge-Kutta second and fourth order) and finding Roots (Bisection method and False position method) numerically.

I. INTRODUCTION

Numerical analysis is the area of mathematics and computer science that creates, analyzes, and implements algorithms for solving numerically the problems of continuous mathematics. Such problems originate generally from real-world applications of algebra, geometry, and calculus, and they involve variables which vary continuously. The formal academic area of numerical analysis varies from highly theoretical mathematical studies to computer

science issues involving the effects of computer hardware and software on the implementation of specific algorithms^[5].

An algorithm is a procedure or formula for solving a problem. The word derives from the name of the mathematician, Mohammed ibn-Musa al-Khwarizmi.

Given a set of data of points $(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$ of a function $y = f(x)$, where $f(x)$ is not known explicitly, it is required to compute the value of the definite integral

$$I = \int_a^b y \, dx \quad (1)$$

We derive a general formula for numerical integration using Newton's forward difference formula. Let the interval $[a, b]$ be divided into n equal subintervals such that

$$a = x_0 < x_1 < \dots < x_n = b$$

Clearly, $x_n = x_0 + n$. Hence the integral becomes $I = \int_{x_0}^{x_n} y \, dx$

Approximating y by Newton's forward difference formula, we obtain. ^[4]

$$I = \int_{x_0}^{x_n} [y_0 + p\Delta y_0 + \left(\frac{p(p-1)}{2!}\right)\Delta^2 y_0 + \left(\frac{p(p-1)(p-2)}{3!}\right)\Delta^3 y_0 + \dots] dx$$

Since $x = x_0 + ph$, $dx = h \, dp$ and hence the above integral becomes

$$\begin{aligned} I &= h \int_0^n [y_0 + p\Delta y_0 + \left(\frac{p(p-1)}{2!}\right)\Delta^2 y_0 + \left(\frac{p(p-1)(p-2)}{3!}\right)\Delta^3 y_0 + \dots] dp \\ &\Rightarrow \int_{x_0}^{x_n} y \, dx = nh[y_0 + \left(\frac{n}{2}\right)\Delta y_0 + \left(\frac{n(2n-3)}{12}\right)\Delta^2 y_0 + \dots] dp \end{aligned} \quad (2)$$

From this general formula, we can obtain different integration formula by putting $n = 1, 2, 3, \dots$ etc.

Programming language C is very flexible and powerful. It originally designed in the early 1970s ^[3]. It allows us to maximum control with minimum command. It is recognized worldwide and used in a multitude of applications especially in Numerical Analysis. Along with other numerous benefits, we have used programming language C in this paper.

Already B. K. Datta et al in [1] and [2] have given the algorithms of numerical differentiation and numerical integration. In this paper we have developed a combined algorithm of numerical differentiation, numerical integration and finding roots numerically.

The outline of this paper is as follows: Section 2 contains the brief description of the existing methods with methodology. In Section 3, we develop an algorithm, using the programming language C, which gives us the solution of a problem simultaneously regarding four popular existing numerical integration methods namely Trapezoidal rule, Simpson's 1/3 rule, Simpson's 3/8 rule and Weddle's rule or the solution of an ordinary differential equation simultaneously regarding four popular existing methods namely Euler, modified Euler, Runge-Kutta second and fourth order or

Author α ρ ω ¥ : Department of Mathematics, Pabna University of Science & Technology (PUST). e-mails: nl.nizhum@gmail.com, pust.pulok@gmail.com, bimaldu@gmail.com, rajpust09@gmail.com
Author σ : Department of EEE, University of Information Technology & Sciences (UITS). e-mail: pr.aminul@gmail.com

the solution of a problem simultaneously regarding two existing numerical methods namely Bisection method and False position method. Moreover, the simulator identifies the method that gives the best solution comparing with possible exact solution of the problem in each case. Conclusions are given at the end at Section 4.

II. EXISTING METHODS

We give a brief description of the existing methods of Numerical Integration like Trapezoidal rule,

$$\int_{x_0}^{x_1} y dx = h \left[y_0 + \frac{1}{2} \Delta y_0 \right] = h \left[y_0 + \frac{1}{2} (y_1 - y_0) \right] = \frac{h}{2} [y_0 + y_1]$$

For the next interval $[x_1, x_2]$ and others we have the similar expression as $\int_{x_1}^{x_2} y dx = \frac{h}{2} [y_1 + y_2]$ and so on. And for the last interval $[x_{n-1}, x_n]$, we have $\int_{x_{n-1}}^{x_n} y dx = \frac{h}{2} [y_{n-1} + y_n]$.

Combining all these expressions, we obtain the rule

$$\int_{x_0}^{x_n} y dx = \frac{h}{2} [y_0 + 2(y_1 + y_2 + \dots + y_{n-1}) + y_n]$$

This is known as Trapezoidal rule.

$$\int_{x_0}^{x_n} y dx = \frac{h}{3} [y_0 + 4(y_1 + y_3 + \dots + y_{n-1}) + 2(y_2 + y_4 + \dots + y_{n-2}) + y_n]$$

This is known as Simson's 1/3 rule.

c) *Simson's 3/8 Rule*

Putting $n = 3$ in (2) all differences higher than the first will become zero and we obtain

$$\int_{x_0}^{x_3} y dx = 3h \left[y_0 + \frac{3}{2} \Delta y_0 + \frac{9}{12} \Delta^2 y_0 + \frac{3}{24} \Delta^3 y_0 \right] = \frac{3}{8} h [y_0 + 3y_1 + 3y_2 + y_3]$$

Similarly, $\int_{x_3}^{x_6} y dx = \frac{3}{8} h [y_3 + 3y_4 + 3y_5 + y_6]$ and so on.

Summing up all these, we obtain

$$\int_{x_0}^{x_n} y dx = \frac{3}{8} h [y_0 + 3y_1 + 3y_2 + 2y_3 + \dots + 2y_{n-3} + 3y_{n-2} + 3y_{n-1} + y_n] \quad [13]$$

This rule is known as Simson's 3/8 rule.

d) *Weddle's Rule*

Putting $n = 6$ in (2) all differences higher than the first will become zero and we obtain

$$\int_{x_0}^{x_6} y dx = \frac{3}{10} h [y_0 + 5y_1 + y_2 + 6y_3 + y_4 + 5y_5 + y_6]$$

Similarly, we obtain the general form

$$\int_{x_0}^{x_n} y dx = \frac{3}{10} h [y_0 + 2(y_6 + y_{12} + \dots + y_{n-6}) + 5(y_1 + y_7 + \dots + y_{n-5}) + 5(y_5 + y_{11} + \dots + y_{n-1}) + 6(y_3 + y_9 + \dots + y_{n-3}) + y_2 + y_4 + \dots + y_n]$$

This is known as Weddle's rule.

Simpson's 1/3 rule, Simpson's 3/8 rule and Weddle's rule, methods of numerical differential equations like Euler, modified Euler, Runge-Kutta second and fourth order and numerical methods namely Bisection method and False position method in this section with their methodology^[2].

a) *Trapezoidal Rule*

Putting $n = 1$ in (2) all differences higher than the first will become zero and we obtain

b) *Simson's 1/3 Rule*

Putting $n = 2$ in (2) all differences higher than the first will become zero and we obtain

$$\int_{x_0}^{x_2} y dx = 2h \left[y_0 + \Delta y_0 + \frac{1}{6} \Delta^2 y_0 \right] = \frac{h}{3} [y_2 + 4y_1 + y_0]$$

Similarly, $\int_{x_2}^{x_4} y dx = \frac{h}{3} [y_2 + 4y_3 + y_4]$ and finally

$$\int_{x_{n-2}}^{x_n} y dx = \frac{h}{3} [y_{n-2} + 4y_{n-1} + y_n]$$

Combining all these expressions, we obtain

e) *Euler Method*

In mathematics and computational science, the Euler method is a first-order numerical procedure for solving ODEs with a given initial value. It is the most basic explicit method for numerical ODEs^[1].

i. *Procedure*

We consider the differential equation

$$y' = f(x, y) \quad (3)$$

with the initial condition $y(x_0) = y_0$ (4)

Suppose that we wish to solve the equation (3) with (4) for the value of y at $x = x_r = x_0 + rh$ ($r = 1, 2, \dots$)

Integrating (3) with y_0 to y_1 and x_0 to x_1 , we get
 $y_1 - y_0 = \int_{x_0}^{x_1} f(x, y) dx$

$$\text{Or, } y_1 = y_0 + \int_{x_0}^{x_1} f(x, y) dx \quad (5)$$

Assuming that $f(x, y) = f(x_0, y_0)$ in $x_0 \leq x \leq x_1$, this gives Euler's formula

$$y_1 \approx y_0 + h f(x_0, y_0) [\text{since } x_1 - x_0 = h] \quad (6)$$

Similarly for the range $x_1 \leq x \leq x_2$, we have
 $y_2 = y_1 + \int_{x_1}^{x_2} f(x, y) dx$

We thus obtain the iterative formula

$$y_1^{(n+1)} = y_0 + \left(\frac{h}{2}\right) [f(x_0, y_0) + f(x_1, y_1^n)] ; n = 0, 1, 2, \dots \quad (11)$$

Where y_1^n is the n th approximation to y_1 . The iterative formula (11) can be started by choosing y_1^0 from Euler's formula $y_1^0 = y_0 + h f(x_0, y_0)$

g) *Runge-Kutta method (Second order)*i. *Procedure*

We consider the differential equation $y' = f(x, y)$ (12)

With the initial condition $y(x_0) = y_0$ (13)

Suppose that we wish to solve the equation (12) with (13) for the value of y at

$$x = x_r = x_0 + rh \quad (r = 1, 2, \dots)$$

Integrating (12) with y_0 to y_1 and x_0 to x_1 , we get
 $y_1 - y_0 = \int_{x_0}^{x_1} f(x, y) dx$

$$\text{Or } y_1 = y_0 + \int_{x_0}^{x_1} f(x, y) dx \quad (14)$$

Now integrating (14) by means of trapezoidal rule to obtain

$$y_1 = y_0 + \left(\frac{h}{2}\right) [f(x_0, y_0) + f(x_1, y_1)] \quad (15)$$

Substitute $y_1 = y_0 + h f(x_0, y_0)$ on the right side of equation (15), we obtain

$$y_1 = y_0 + \left(\frac{h}{2}\right) [f_0 + f(x_0 + h, y_0 + h f_0)] \quad (16)$$

Substituting $f(x_1, y_1)$ for $f(x, y)$ where $x_1 \leq x \leq x_2$, we have

$$y_2 \approx y_1 + h f(x_1, y_1) [\text{since } x_2 - x_1 = h]$$

Proceeding in this way, we obtain the general formula

$$y_{n+1} = y_n + h f(x_n, y_n), \quad n = 0, 1, 2, \dots$$

f) *Modified Euler's method*i. *Procedure*

We consider the differential equation

$$y' = f(x, y) \quad (7)$$

With the initial condition $y(x_0) = y_0$ (8)

Suppose that we wish to solve the equation (7) with (8) for the value of y at

$$x = x_r = x_0 + rh \quad (r = 1, 2, \dots)$$

Integrating (7) with y_0 to y_1 and x_0 to x_1 , we get
 $y_1 - y_0 = \int_{x_0}^{x_1} f(x, y) dx$

$$\text{Or, } y_1 = y_0 + \int_{x_0}^{x_1} f(x, y) dx \quad (9)$$

Now integrating (9) by means of trapezoidal rule to obtain

$$y_1 = y_0 + \left(\frac{h}{2}\right) [f(x_0, y_0) + f(x_1, y_1)] \quad (10)$$

Where $f(x_0, y_0) = f_0, x_1 - x_0 = h$

Now set $k_1 = h f_0$ and $k_2 = h f(x_0 + h, y_0 + k_1)$. And hence equation (16) becomes

$$y_1 = y_0 + \left(\frac{1}{2}\right) [k_1 + k_2].$$

This is the Runge-Kutta second order formula.

h) *Runge-Kutta method (Fourth order)*i. *Procedure*

We mention the fourth order formulae defined by

$$y_1 = y_0 + W_1 k_1 + W_2 k_2 + W_3 k_3 + W_4 k_4 \quad (17)$$

Where

$$k_1 = h f(x_0, y_0)$$

$$k_2 = h f(x_0 + \alpha_0 h, y_0 + \beta_0 k_1)$$

$$k_3 = h f(x_0 + \alpha_1 h, y_0 + \beta_1 k_1 + v_1 k_2)$$

$$k_4 = h f(x_0 + \alpha_2 h, y_0 + \beta_2 k_1 + v_2 k_2 + \delta_1 k_3) \quad (18)$$

Where the parameters have to be determined by expanding both sides of (17) by Taylor's series and securing agreement of terms up to and including those containing h^4 . The choice of the parameters is, again arbitrary and we have therefore several fourth order Runge-kutta formulae. If for example we set

$$\alpha_0 = \beta_0 = \alpha_1 = \frac{1}{2}, \alpha_2 = 1$$

$$\beta_1 = \frac{1}{2}(\sqrt{2} - 1), \beta_2 = 0$$

$$v_1 = 1 - \frac{1}{\sqrt{2}}, v_2 = -\frac{1}{\sqrt{2}}, \delta_1 = 1 + \frac{1}{\sqrt{2}}$$

$$W_1 = W_4 = \frac{1}{6}, W_2 = \frac{1}{3}\left(1 - \frac{1}{\sqrt{2}}\right), W_3 = \frac{1}{3}\left(1 + \frac{1}{\sqrt{2}}\right)$$

We obtain the method of Gill, whereas the choice

$$\alpha_0 = \alpha_1 = \frac{1}{2}, \beta_0 = v_1 = \frac{1}{2}$$

$$\beta_1 = \beta_2 = v_2 = 0, \alpha_2 = \delta_1 = 1$$

$$W_1 = W_4 = \frac{1}{6}, W_2 = W_3 = \frac{2}{6}$$

Leads to the fourth order Runge-Kutta formulae, whereas

$$k_1 = hf(x_0, y_0)$$

$$k_2 = hf\left(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_1\right)$$

$$k_3 = hf\left(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_2\right)$$

$$k_4 = hf(x_0 + h, y_0 + k_3)$$

Then

$$y_1 = y_0 + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

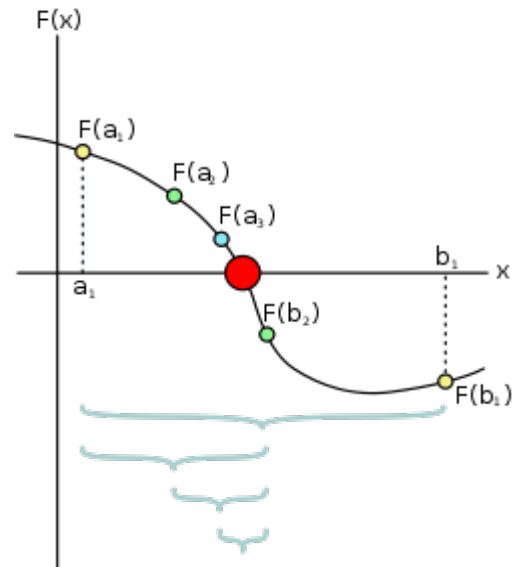
i) Bisection method

The bisection method in mathematics is a root-finding method which repeatedly bisects an interval and then selects a subinterval in which a root must lie for further processing. It is a very simple and robust method, but it is also relatively slow. Because of this, it is often used to obtain a rough approximation to a solution which is then used as a starting point for more rapidly converging methods. The method is also called the binary search method or the dichotomy method^[4].

i. Procedure

The method is applicable when we wish to solve the equation $f(x) = 0$ for interval $[a, b]$ and $f(a)$ the real variable x , where f is a continuous function defined on an and $f(b)$ have opposite signs. In this case a and b are said to bracket a root since, by the intermediate value theorem, the f must have at least one root in the interval (a, b) . At each step the method divides the interval in two by computing the midpoint $c = (a + b)/2$ of the interval and the value of the function $f(c)$ at that point. Unless c is itself a root (which is very unlikely, but possible) there are now two possibilities: either $f(a)$ and $f(c)$ have opposite signs and bracket a root, or $f(c)$ and $f(b)$ have opposite signs and bracket a root. The method selects the subinterval that is a bracket as a new interval to be used

in the next step. In this way the interval that contains a zero of f is reduced in width by 50% at each step. The process is continued until the interval is sufficiently small.



Explicitly, if $f(a)$ and $f(c)$ are opposite signs, then the method sets c as the new value for b , and if $f(b)$ and $f(c)$ are opposite signs then the method sets c as the new a . (If $f(c) = 0$ then c may be taken as the solution and the process stops.) In both cases, the new $f(a)$ and $f(b)$ have opposite signs, so the method is applicable to this smaller interval.

j) False Position Method

In problems involving arithmetic or algebra, the false position method or regula falsi is used to refer to basic trial and error methods of solving problems by substituting test values for the unknown quantities.

i. Procedure

The poor convergence of the bisection method as well as its poor adaptability to higher dimensions (i.e., systems of two or more non-linear equations) motivate the use of better techniques^[6]. One such method is the Method of False Position. Here, we start with an initial interval $[x_1, x_2]$ and we assume that the function changes sign only once in this interval. Now we find an x_3 in this interval, which is given by the intersection of the x axis and the straight line passing through $(x_1, f(x_1))$ and $(x_2, f(x_2))$. It is easy to verify that x_3 is given by

$$x_3 = x_1 - \frac{(x_2 - x_1)f(x_1)}{f(x_2) - f(x_1)}$$

Now, we choose the new interval from the two choices $[x_1, x_3]$ or $[x_3, x_2]$ depending on in which interval the function changes sign.

The false position method differs from the bisection method only in the choice it makes for subdividing the interval at each iteration. It converges faster to the root because it is an algorithm which uses appropriate weighting of the initial end points x_1 and x_2 using the information about the function, or the data

of the problem. In other words, finding x_3 is a *static* procedure in the case of the bisection method since for a given x_1 and x_2 , it gives *identical* x_3 , no matter what the function we wish to solve. On the other hand, the false position method uses the information about the function to arrive at x_3 .

III. ALGORITHM

INPUT: Type your choice C.

If C==1

{

INPUT: function f , limits x_0 and x_n , number of division n , direct result r .

Step-1: Compute $h = \frac{x_n - x_0}{n}$

Step-2: Set $i = 0$

Step-3: While $i \leq n$, repeat Step- 4

Step-4: Set $y[i] = f(x_0 + ih)$

Step-5: Set $i = 1$

Step-6: While $i < n$, repeat Step-7

Step-7: Set $sum_1 = sum_1 + 2y[i]$

 If $i \% 2 = 0$

 Set $sum_2 = sum_2 + 2y[i]$

 Else

 Set $sum_2 = sum_2 + 4y[i]$

 If $i \% 3 = 0$

 Set $sum_3 = sum_3 + 2y[i]$

 Else

 Set $sum_3 = sum_3 + 3y[i]$

$t_sum_1 = (h/2) * (y[0] + y[n] + sum_1)$

$t_sum_2 = (h/3) * (y[0] + y[n] + sum_2)$

$t_sum_3 = (3h/8) * (y[0] + y[n] + sum_3)$

 If $i \% 6 = 0$

 Set $sum_4 = sum_4 + 2y[i]$

 Else if $i \% 3 = 0$

 Set $sum_4 = sum_4 + 6y[i]$

 Else

 Set $sum_4 = sum_4 + y[i]$

 Else If $(i \% 6 == 1 || i \% 6 == 5)$

$Sum_4 = sum_4 + 5y[i]$

 Else

$Sum_4 = sum_4 + y[i];$

$t_sum_4 = \left(\frac{3h}{8}\right) * (y[0] + y[n] + sum_4)$

Step-8: Set $a = |t_sum_1 - r|, b = |t_sum_2 - r|, c = |t_sum_3 - r|, d = |t_smu_4 - r|$

 If $(a < b \text{ and } a < c)$

 If $(a < d)$

t_sum_1

```

Else
  t_sum_4
If(b < a and b < c)
  If(b < d)
    t_sum_2
  Else
    t_sum_4
  If(c < a and c < b)
    If(c < d)
      t_sum_3
    Else
      t_sum_4

```

48

OUTPUT: $t_sum_1, t_sum_2, t_sum_3$ and t_sum_4 with message which sum is most accurate.
STOP.

```

}^{[2]}

```

```

Else If C==2

```

```

{

```

INPUT: function $f(x, y)$, initial condition (x_0, y_0) , interval h , value of x , direct result r .

Step-1: Set $n = (x - x_0)/h$, $y_{00} = y_0, y_{0e} = y_0, y_{02} = y_0$

$$y_{04} = y_0, y_1 = y_{00} + hf(x_0, y_{00})$$

Step-2: Set $i = 1$

Step-3: While $i \leq n$, repeat Step-4 to step-7

Step-4: Set $j = 1$

Step-5: While $j \leq i$ repeat step-10

Step-6: Set $x_1 = x_0 + h, y_{10} = y_{00} + \frac{1}{2}hf(x_0, y_0) + f(x_1, y_1), y_1 = y_{10}$

Step-7: Set $y_{1e} = y_{0e} + hf(x_0, y_{0e}), k_{11} = hf(x_0, y_{04}), k_{22} = hf(x_0 + \frac{1}{2}h, y_{04} + \frac{1}{2}k_{11})$

$$k_{33} = hf\left(x_0 + \frac{1}{2}h, y_{04} + \frac{1}{2}k_{22}\right), k_{44} = hf(x_0 + h, y_{04} + k_{33})$$

$$r_4 = y_{04} + (k_{11} + 2k_{22} + 2k_{33} + k_{44})/6$$

$$k_1 = hf(x_0, y_{02}), k_2 = hf(x_0 + h, y_{02} + k_1)$$

$$y_1r_2 = y_{02} + \frac{1}{2}(k_1 + k_2), y_{0e} = y_{1e}, y_{00} = y_{10}$$

$$y_{04} = y_1r_4, y_{02} = y_1r_2, x_0 = x_n$$

Step-8: Set $a = |y_{1e} - r|, b = |y_{10} - r|, c = |y_1r_2 - r|, d = |y_1r_4 - r|$

```

  If(a < b and a < c)

```

```

    If(a < d)

```

```

      y_{1e}

```

```

    Else

```

```

      y_1r_4

```

```

  If(b < a and b < c)

```

```

    If(b < d)

```

```

      y_{10}

```

```

    Else

```

```

      y_1r_4

```

```

  If(c < a and c < b)

```

If($c < d$)

$y_1 r_2$

Else

$y_1 r_4$

Output: $y_{1e}, y_{10}, y_1 r_2$ and $y_1 r_4$ with message which method gives best solution.

}^[1]

Else If $C=3$

{

INPUT: function f end points a, b ; initial approximations p_0, p_1 ; tolerance TOL, maximum number of iterations N direct result r .

Step-1: Set $i = 2$;

$$q_0 = f(p_0);$$

$$q_1 = f(p_1).$$

Step-2: While $i \leq N$ do Steps 3-7.

Step-3: Set $p = p_1 - q_1(p_1 - p_0)/(q_1 - q_0)$

Step-4: If $|p - p_1| \leq \text{TOL}$ then

OUTPUT(p);

STOP.

Step-5: Set $i = i + 1$;

$$q = f(p).$$

Step-6: If $q * q_1 < 0$ then set

$$p_0 = p_1;$$

$$q_0 = q_1.$$

Step-7: Set $p_1 = p$;

$$q_1 = q.$$

Step-8: OUTPUT (failure)

Step-9: Set $i = 1$;

$$FA = f(a).$$

Step-10: While $i \leq N$ do steps 11-14

Step-11: Set $bi = a + (b - a) / 2$;

$$FB = f(bi).$$

Step-12: If $FB=0$ or $\frac{b-a}{2} < \text{TOL}$ then

OUTPUT(bi)

STOP.

Step-13: $i = i + 1$.

Step-14: If $FA * FB > 0$ then set $a = bi$;

FA=FB

Else set $b = bi$

Step-15: OUTPUT(failure)

Step-16: Set $c = |p - r|, d = |bi - r|$

If $(c < d)$ then

P

Else

bi

OUTPUT: A message which root is most accurate between Bisection and False position method.

STOP.

}

Else

STOP.

IV. CONCLUSION

In this paper, we develop an algorithm incorporated with Numerical Integration (Trapezoidal rule, Simpson's 1/3 rule, Simpson's 3/8 rule and Weddle's rule.), Numerical Differentiation (Euler, modified Euler and Runge-Kutta second and fourth order) and finding Roots (Bisection method and False position method) numerically. We observed that the result obtained according to our procedure is completely identical with the hand calculation and save our time and labour. Moreover, Weddle's rule gives the best solution, the Runge-Kutta fourth order gives the best solution and the False position method gives the best solution in Numerical Integration, Numerical Differentiation and finding Roots numerically respectively.

REFERENCES RÉFÉRENCES REFERENCIAS

1. B. K. Datta, N. Rahman, & R. C. Bhowmik, A Numerical Simulator for Solving Ordinary Differential Equations, IJSET, V-3,I-3,pp-250-252, 2014.
2. B. K. Datta, N. Rahman, R. C. Bhowmik, U. Roy, M.R.Kabir & S.Paul, A Numerical Simulator for Solving Numerical Integration, IJSET, V-3, I-4, pp-342-345, 2014.
3. Wiki Answers, wiki.answer.com.5
4. M. Goyal, Computer-based Numerical & Statistical Techniques, Infinity Science Press LLC, New Delhi, India, 2007.3.
5. www.Scholarpedia.org.
6. A. Kaw, E.E. Kalu, Numerical Methods with Applications, Lalu.com, 2008.