Numerical Solution of Ordinary Differential Equations Using Continuous Runge-Kutta Methods (Feldberg of Order Four and Five)

Madeha yousif
University of Technology, Iraq, 100049@utechnology.edu.iq

Follow this and additional works at: https://scholarworks.uaeu.ac.ae/ejer

Part of the Ordinary Differential Equations and Applied Dynamics Commons

Recommended Citation
Available at: https://scholarworks.uaeu.ac.ae/ejer/vol25/iss3/3

This Article is brought to you for free and open access by Scholarworks@UAEU. It has been accepted for inclusion in Emirates Journal for Engineering Research by an authorized editor of Scholarworks@UAEU. For more information, please contact EJER@uae.ac.ae.
NUMERICAL SOLUTION OF ORDINARY DIFFERENTIAL EQUATIONS USING CONTINUOUS RUNGE-KUTTA METHODS (FELDBERG OF ORDER FOUR AND FIVE)

Madeha Sh.yousif
University of Technology- Applied Science Department
100049@utechnology.edu.iq

(Received 7th April 2020 and Accepted 26th June 2020)

Abstract
In this paper the continuous Runge-Kutta method (Runge-Kutta Feldberg method of order four and five) have been used to find the numerical solution of ordinary differential equation not only at the mesh points but also the all points between them. the results are computed using matlab program.

Key words: Ordinary Differential Equations, Continuous Runge-Kutta Methods

1-Introduction
The initial value problem can be defined as
\[ y' = f(x, y(x)) \quad s \geq x_0 \quad \ldots(1) \]
\[ y(x_0) = y_0 \]
Which can be solved by the following s-stage Runge-Kutta method, and s denotes the number of stages of methods [1].
The coefficients \{a_0\}, \{c_i\} and \{b_j\} fully characterize Runge-Kutta method and we shall assume that the following condition holds.
\[ y_{n+1} = y_n + h \sum_{i=1}^{s} b_i k_i \quad \ldots(2) \]
where
\[ k_i = f(x_n + c_i \cdot h, y_n + h \sum_{j=1}^{s} a_{ij} k_j), \quad i = 1, 2, \ldots, s \]
\[ c_i = \sum_{j=1}^{s} a_{ij}, \quad i = 1, \ldots, s \]
conjunction with this defining algorithm, the analysis for these defining RK algorithms is local in nature, requiring no assertions about previous approximate solution values.

Using equation (2) to find the solution of (1)

$$y(x_0 + h) = y_1 = y_0 + h \sum_{i=0}^{s} b_i k_i$$  \(\text{...}(3)\)

where

$$k_0 = f(x_0, y_0)$$  \(\text{...}(4)\)

and

$$k_i = f(x_0 + c_i h, y_0 + h \sum_{j=0}^{i-1} a_{ij} k_j)$$  \(\text{...}(5)\)

For i=1, 2,...,s, where s is defined as the number of stages of the algorithm and where the a, b and c coefficients have been selected so that the algorithmic solution, y1 is equivalent to a Taylor sum of order p.

Specifically, subtracting a Taylor series expansion in h of y1 from an expansion of y(x0+h) results in certain truncation error coefficients TEC i,j, as coefficients of the local truncation error, where TEC i,j term are expressions in a, b and c for the order m, with i=1,2,...,\lambda_m where \lambda_m increases with increasing order. For an algorithm to be of order p, all TEC i,j terms must vanish for m=1,…,p, i=1,..., \lambda_m. The a_0 may be determined from the relations:

$$\sum_{j=0}^{i-1} a_{ij} = c_i$$  \(\text{...}(6)\)

The form of the continuous algorithm parallels that of the defining algorithm since the efficiency of the new method requires the use of derivative evaluations (4,5) as the core of the new system. One assumes that the solution has been advanced from x_0 to x_1=x_0+h, i.e., that k_1,k_2,…,k_s have been determined and that the solution is required at some intermediate point, x_1=x_0+h with

$$h^* = \delta h$$  \(\text{...}(7)\)

where generally the scaling factor \(\delta \in (0, 1)\).

The continuous algorithm assumes the basic form of the defining algorithm

$$y(x_0 + h^*) = y_1^* = y_0 + h^* \sum_{i=0}^{s} b_i k_i^*$$  \(\text{...}(8)\)

where

$$k^*_0 = f(x_0, y_0)$$  \(\text{...}(9)\)

and

$$k^*_i = f(x_0 + c_i h^*, y_0 + h^* \sum_{j=0}^{i-1} a_{ij} k^*_j)$$  \(\text{...}(10)\)


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

$$k_2 = hf(x_0 + \frac{1}{4}h, y_0 + \frac{1}{4}k_1)$$

$$k_3 = hf(x_0 + \frac{3}{8}h, y_0 + \frac{3}{32}k_1 + \frac{9}{32}k_2)$$

$$k_4 = hf(x_0 + h, y_0 + k_1)$$

Then an approximation to the solution of the initial value problem using a Runge-Kutta method of order four:

$$y_{n+1} = y_n + \frac{25}{216} k_1 + \frac{1408}{2565} k_2 + \frac{2197}{4014} k_3 - \frac{1}{5} k_4$$

where the four function values k_1 , k_3, k_4 and k_5 are used k_2.

better value for the solution is determined using a Runge-Kutta method of order five:-

$$Z_{n+1} = y_n + \frac{16}{135} k_1 + \frac{6656}{12825} k_2 + \frac{28561}{56430} k_3 - \frac{9}{50} k_4 + \frac{2}{55} k_5$$

The optimal step size (sh) can be determined by multiplying the scalar s times the current step size h, the scalar s is

$$s = \left( \frac{tol h}{2|z_{n+1} - y_{n+1}|} \right)^{\frac{1}{4}} \approx 0.84 \left( \frac{tol h}{|z_{n+1} - y_{n+1}|} \right)^{\frac{1}{4}}$$

where tol is the specified error control tolerance[5].

2-Construction of Continuous Runge-Kutta Methods[1,2,3,4].

We can extend the Explicit Runge-Kutta methods into Continuous Runge-Kutta method which determines the solution of the IVP at any point x*=x_0+\delta h, once the solution has been evaluated at x_1=x_0+h, where the scaling factor, \(\delta \), generally ranges between 0 and 1.

By using the derivative evaluations generated in obtaining y(x_1), this new continuous solution may be determined at a cost of only a few additional derivative evaluations. Thus, for a given Runge-Kutta algorithm, a family of continuous algorithms may be developed, which, when used in
The method is available with the advantage that once additional derivative evaluation has been made, the solution may be determined at any number of points within the given step for a few arithmetic operations, while still maintaining the accuracy of the defining algorithm.

When the RKF(4)5 coefficients RKF(4)5 are used and \( c_0 = 1, a_n = a_6 = 0 \), are chosen with \( b_1 = 0 \), the continuous coefficients become:

\[
\begin{align*}
b_0 &= 1 - \delta (301/120 + \delta (269/108 + \delta (311/360))) \\
b_3 &= \delta (3/2 + \delta (199927/22572 + \delta (311/360))) \\
b_1 &= \delta (57/50 + \delta (3/2 + \delta (42/25))) \\
b_4 &= \delta (-96/55 + \delta (3907/2056 + \delta (4096/513))) \\
b_2 &= \delta (-102/55 + \delta (4096/513 + \delta (311/360))) \\
b_5 &= \delta (3/2 + \delta (4 + \delta (5/2)))
\end{align*}
\]

**Remark [2 ][3]**

We can construct a CRK method of high order by following the same approach.

**4-Numerical Examples**

**Example(1):**

Consider the following differential equation of the first order:

\[
y' = y^2 \sin x \quad y(0) = 1 \quad 0 \leq x \leq 1
\]

The exact solution has the form

\[
y(x) = \sec x
\]

Table (1) presents the numerical results for the global error. RKF(4)th and 5th and CRK are used with differential value of \( \delta \).

<table>
<thead>
<tr>
<th>( h )</th>
<th>( x )</th>
<th>RKF(4)</th>
<th>RKF(5)</th>
<th>CRK</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.1</td>
<td>4.7404e-003</td>
<td>4.7404e-003</td>
<td>( \delta = 1 )</td>
</tr>
<tr>
<td>0.1</td>
<td>0.2</td>
<td>4.7404e-003</td>
<td>4.7404e-003</td>
<td>1.0696e-003</td>
</tr>
<tr>
<td>0.5</td>
<td>0.5</td>
<td>1.0404e-001</td>
<td>1.0404e-001</td>
<td>1.0404e-001</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>5.4800e-001</td>
<td>5.4800e-001</td>
<td>5.4800e-001</td>
</tr>
<tr>
<td>0.05</td>
<td>0.1</td>
<td>4.7404e-003</td>
<td>-6.6912e-011</td>
<td>4.7404e-003</td>
</tr>
<tr>
<td>0.05</td>
<td>0.2</td>
<td>4.7404e-003</td>
<td>-6.6912e-011</td>
<td>4.7404e-003</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>5.4800e-001</td>
<td>-3.2211e-008</td>
<td>5.4800e-001</td>
</tr>
</tbody>
</table>

For \( i = 1,2,\ldots,s \), where \( s \geq s \). If \( c^\ast \) and \( a^\ast \) are chosen so that

\[
k^\ast_i = k_i \quad \ldots(11)
\]

for \( i = 1, \ldots, s \) then the major expense of applying the algorithm has been avoided since \( k_1, k_2, \ldots, k_s \) have already been determined during the evaluation of \( y(x) \). Thus, only \( k_{s+1}, \ldots, k_s \) need to be evaluated.

The proposed strategy requires that the \( c^\ast_i \) and \( a^\ast_j \) parameters of the continuous algorithm be related to their respective \( c_i \) and \( a_n \) coefficients in the defining algorithm, for \( i = 1, \ldots, s \). The additional parameters \( c^\ast_i \) and \( a^\ast_j \), \( i = s+1, \ldots, s^\ast \) are then used to generate a continuous algorithm at accuracy comparable to that of the defining algorithm for a particular value of \( \delta \).

Comparing (4.5) with (9,10) one finds that if:

\[
c^\ast_i = \frac{c_i h}{h^\ast} = \frac{c_i}{\delta} \quad \ldots(12)
\]

and

\[
a^\ast_j = \frac{a_{ij} h}{h^\ast} = \frac{a_{ij}}{\delta} \quad \ldots(13)
\]

for \( i = 1,2,\ldots,s \), \( j = 0,1,\ldots,i-1 \), then \( k^\ast_i = k_i \) for \( i = 1, \ldots, s \).

Thus for \( i \leq s \), the \( c^\ast_i \) and \( a^\ast_j \) parameters are already determined, being related to their corresponding coefficients in the defining algorithm by a factor of \( \frac{1}{\delta} \). These \( k^\ast_i \) evaluations will have already been computed during the application of the defining algorithm.

**3-A Continuous Fourth-order Runge-Kutta Method [5,7]**

Using the coefficients of the defining algorithm of RKF(4)5 continuous fourth-order numerical results RKF(4)th and 5th and CRK.

Table (1)
Consider the following differential equation of the first order:

\[ y' = y + 3x - x^2 \quad y(0) = 1 \quad 0 \leq x \leq 1 \]

The exact solution has the form:

\[ y(x) = 2e^x + x^2 - x - 1 \]

Table (2) presents the numerical results for the global error, RKF(4th and 5th) and CRK are used with differential value of \( \delta \).

<table>
<thead>
<tr>
<th>( h )</th>
<th>( x )</th>
<th>RKF(4)</th>
<th>RKF(5)</th>
<th>CRK</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>1.4373e+000</td>
<td>1.4373e+000</td>
<td>1.4373e+000</td>
<td>-5.9260e+001</td>
</tr>
<tr>
<td>0.05</td>
<td>1.4373e+000</td>
<td>1.4373e+000</td>
<td>1.4373e+000</td>
<td>-5.9260e+001</td>
</tr>
</tbody>
</table>

Consider the following differential equation of the first order:

\[ y(x) = \left( \frac{x^3 + 3}{3} \right)^{1/3} \]

Table (3) presents the numerical results for the global error, RKF(4th and 5th) and CRK are used with differential value of \( \delta \).

<table>
<thead>
<tr>
<th>( h )</th>
<th>( x )</th>
<th>RKF(4)</th>
<th>RKF(5)</th>
<th>CRK</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.1</td>
<td>-3.3389e-003</td>
<td>-3.3389e-003</td>
<td>-3.3389e-003</td>
</tr>
<tr>
<td>0.5</td>
<td>8.6736e-002</td>
<td>8.6736e-002</td>
<td>8.6736e-002</td>
<td>-4.2287e-001</td>
</tr>
<tr>
<td>1</td>
<td>-3.490e-001</td>
<td>-3.490e-001</td>
<td>-3.490e-001</td>
<td>-2.1890e+000</td>
</tr>
<tr>
<td>0.01</td>
<td>0.1</td>
<td>-3.3389e-003</td>
<td>-3.3389e-003</td>
<td>-3.3389e-003</td>
</tr>
<tr>
<td>0.5</td>
<td>8.6736e-002</td>
<td>8.6736e-002</td>
<td>8.6736e-002</td>
<td>-4.9189e-001</td>
</tr>
</tbody>
</table>

The obtained results show that the CRK method is better than RKF method of the same order.

If \( 0 \leq \delta < 1 \) takes different values, we obtain the solution between the discrete points. The error may be increased because we need rounding error.

6-References

5-Conclusions

1. The obtained results show that the CRK method is better than RKF method of the same order.
2. If \( \delta = 1 \) in CRK method the results are approximately equal to the results of RKF(4th)5th method.


