On the implementation of Implicit Methods to Solve Initial Values Problems for Ordinary Differential Equations

Badreldin O. S. Elgabbani[1]
Department of Engineering and Applied Sciences
Umm Al-Qura University
Makkah, Saudi Arabia
boelgabbani@uqu.edu.sa

Othman A. Alrusaini[2]
Department of Engineering and Applied Sciences
Umm Al-Qura University
Makkah, Saudi Arabia
oarusaini@uqu.edu.sa

Emad A. Shafie[3]
Department of Engineering and Applied Sciences:
Umm Al-Qura University
Makkah Saudi Arabia
eashafie@uqu.edu.sa

Abstract
The study presents a new method for solving initial value problems (IVPs) for ordinary differential equations (ODEs). The study successfully satisfied Butcher conditions and got a Jacobian matrix: a_{11}, a_{12}, a_{13}, a_{21}, λ ,a_{23}, a_{31}, a_{32}, λ and then applying modified Newton, to find out the unknown vector λ_i from iterative equation $y_i = y_i + \partial_i$ .The diagonal-implicit Range-Kutta, method submitted by this study solves the problem arising from application of an implicit multistage for linear or nonlinear algebra problems.The results achieved by using these parameters via Visual studio C++, compared to the ordinary differential equations that have an exact solution are more than impressive and get a shouted result, for example while comparing the exact solution of an ordinary differential equation with our numerical solution for the same problem, we get an approximation error 9.9873D-05. Therefore, with great confidence we can use this method with problems that need numerical solution.

Keywords:

Ordinary differential equation; Butcher; Runge-Kutta; implicit;, Array

1. Introduction

It seems clearly that many areas of engineering and scientific analysis require methods for solving set of initial value problem of ordinary differential equations (ODEs). At first scalar (ODEs) need to be considered $\frac{dy}{dx} = \acute{y}(x) = f(x,y)$ with initial condition given by $y(x_0) = y_0$ where $f(x,y)$ indicates any explicit functionality between the dependent variable y and independent variable x, x_0 represent specific value of x and y mentioned initial value of y(x) at $x = x_0$

This in general called initial value problem (IVP) for ordinary differential equations (ODEs)[16] Implicit methods for solving stiff problems, force researcher to solve nonlinear system[17] of algebraic equations.

Some of them have tried to improve the implementing of I.R.K method to take advantage of their good stability; others have tried to improve the stability properties of Linear Multistep Methods [1] by present a survey of implicit Rung-Kutta methods [2].

Recently some researcher investigate the implementation of multistep multistage [3] methods on parallel computer [4].

The study presented in this paper satisfied Bucher array condition and get our own parameters, the study got a shouted results comparing to other methods here by we come across some definitions and theorems needed in this study

- Continuity

A vector function $\underline{\phi}(\underline{x})$ is said to be continuous at a point $\underline{x}_0$ if, given any positive number,we can choose a positive number δsuch that$\|\ \phi(\underline{x}) - \phi(\underline{x}_o)\ \| \le \varepsilon$ whenever$\|\ \underline{x} - \underline{x}_o\ \| \le \delta$ (that is whenever $\underline{x}$ is an element of open neighborhood ($\underline{x}_o$, δ) .

- Convergence

Consider the numerical solution of initial value problems
[7, 8] given by $\sum \propto\ y_{n+1} = h\,\emptyset\, f(\ y_{n+k}\ , y_{n+k-1}\ , \ldots\ldots, y_n,\ x_n; h)$
$\acute{Y} = F(x,Y)\ \ Y(a) = \varkappa$Where $\varkappa = [\eta_1\ , \eta_2\ \ , \ldots.., \eta_q\ ]^T$
$y_\mu = (h)\ \ \mu = 0,1,2, \ldots\ldots k-1$
is said to be convergence if for all initial value problems
Satisfying the hypotheses of the mean value theorem where
$\text{MAX}\|\,(y(x_n) - y)\| \to 0$ as $h \to 0$,$0 \le n_0 \le N$

- Mean Value Theorem

Where J is a Jacobin matrix of F with respect to Z and the bar ,indicate that each row of J is evaluated at different mean value that is F(Z) -F(Z*) = J(ζ)(Z - Z*). Where J is a Jacobin matrix of F with respect to Z and the

Manuscript received March 5, 2022
Manuscript revised March 20, 2022
https://doi.org/10.22937/IJCSNS.2022.22.3.66

bar indicate that each row of J is evaluated at different mean value is:

$$J(\xi)=\begin{bmatrix} F_{11}(\xi_1) & F_{11}(\xi_1)\ldots\ldots & F_{11}(\xi_1)\\ F_{21}(\xi_{m1}) & F_{22}(\xi_2) & F_{2m}(\xi_2)\\ \ldots\ldots\ldots & \ldots\ldots\ldots & \ldots\ldots\\ F_{m1}(\xi_m) & F_{m2}(\xi_m) & F_{mm}(\xi_m)\end{bmatrix}$$

- Stability

Let us now apply the Runge-Kutta method with R=3 to the test equation $y'=\lambda y$ then

K 1 = h* f(x ,y) =λy

K2 = h *f(x +c 2h,y + c 2 k 1) = λ (y+h c 2 λy)= λ y(1+hc$_2$λ)K 3 =h *f(x +c 3h, y+(c3 - a 32)k1+a$_{32}$ k 2)

=λ[(y +(c3 - a 32) hλy +a 32 hλy(1+ λhc 2)]

=λy(1+λhc 3+ c 2 a 32 hλ^2Y^2)

Hence ϕ(x i, y, h) = b 1K 1+b2K 2+ b 3K 3

=λy[(b 1+b2+ b 3)+ hλ(b 2C 2 +b 3C 3)+ b 2 c 2 a 32 $\lambda^2 h^2$]

Then we obtain

y_{n+1} -y_n=λh[(b1+b2+b3)+ hλ(b2 C2 +b3 C3)+ b2 c 2 a_{32} λ^2h^2] y_n

Let $h = h\lambda$

yn+1 /y_n=[1+(b 1+b2+b 3) h + h^2 (b 2 C 2+b 3C 3)+ b2 c2a32 h^3]

We can then define the three-stage Runge-Kutta method to be absolutely stable on the interval (a, b) if r_1 satisfies | r_1|< 1 whenever $h \in$(a,b) [5].

The concept of convergence and stability [14] are concerned with the limiting process as h → 0. In practice with finite number of steps, we are concerned with the size of errors

A numerical method is said to be a stable if its region of absolute stability contain the whole of the left hand plane Re $h < 0$.[6]

I. Matrix Direct product

Let A = [a] be an S×S matrix and Let B be an M×M matrix then the direct product of Aij and B denoted by A⊗B is an MS×MS matrix defined by

$$A\otimes\mathbf{B}=\begin{bmatrix} a_{11}B & a_{12}B & \cdots & \cdots & a_{1S}B\\ a_{21}B & a_{22}B & \cdots & \cdots & a_{2S}B\\ \cdots & \cdots & \cdots & \cdots\cdots & \cdots\\ a_{S1}B & a_{S2}B & \cdots & \cdots & a_{SS}B\end{bmatrix}$$

2. PREVIOUS STUDY

II. RUNGE-KUTTA METHODS

The easiest method for numerical Solution [9,15] of the initial value problem [2]

$\acute{y} = f(x, y)$ $y(a) = \eta$,is Euler's rule $y_{N+1} - y_N = h f(xn\ yn) \equiv h f_N$

It is explicit and, being a one- step method, it requires no additional starting values and readily permits a change of step-length during computation. It's low order, makes it of limited practical value.

This is the philosophy behind the method first proposed by Runge and subsequently developed by Kutta and Huen. Runge-Kutta method thus reflects the advantages of one-step methods.

Traditionally Runge-Kutta methods are all explicit, although recently implicit Runge- Kutta methods [2]

III. Development of Single Step of Runge-kutta Formulas

For x∈[x$_0$, b] is said to be a one- step method if y_{N+1} the approximation to y(x_{n+1}) is obtained by use of an algorithm of the form

$y_{New} = y_N + h\,\phi\,(xN\ , y_{N+1}; h)$

That is y_{n+1} is completely determined in terms of x n, y_N and h. The Taylor's algorithm of order n is a one-step method because

$$y_{i+1} = y_i + h(y_i + \frac{h}{2!}y_i + \ldots + \frac{h^{n-1}}{2!}y_i^{(n)})$$

Now we express the method in another way by the following formula $Y_{i+1} = Y_i + \sum_{i=1}^{n} w_i k_i$

Where w_i are weighting coefficients to determined r is a number of f(x, y) substitutions and k $_i$ satisfying the explicit sequence.

IV. Derivation of Second-Order R.K Method

We have

$$y_{n+1} = y_n + w_1 k_1 + w_2 k_2$$
$$K_1 = h f(xn\ yn)$$
$$K_2 = h f(xn + c_2 h, yn + a_{21} k_1)$$

And from Taylor series expansion

$y(x_n+1) = f(x_n\ , y_n) + h \acute{f}(x_n\ , y_n) + \ldots\ldots\ldots$

An equivalent form is

$$y_{N+1} = y_N + hf_N + h^2(\frac{1}{2}f_x + \frac{1}{2}f_y f)$$

And since we have

$$K_1 = h f(x_n, y_n)$$
$$K_2 = h f\left(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_1\right)$$

Then we can expand them in Taylor series by

$y_{n+1} = y_n + w_1 h f_n + w_2 H f[(x_n + c_2 h, y_n + a_{21}\ h f(x_n, y_n)]$

$y_{n+1} = y_n + (w_1 + w_2)hf_n + w_2 H\, 2\,(a_{21} C_2 f_x + a_{21} w_2 f f_y)$

Comparing 2.1.5 and 2.2. 4 they will be identical if

$$w_1 + w_2 = 1 , w_2 c_2 = \frac{1}{2}$$

$$w_2 a_{21} = \frac{1}{2}$$

Then we will have

$$y_{n+1} = y_n + \frac{2}{3} k_1 + \frac{1}{3} k_2$$
$$K_1 = h f (x_n \; y_n)$$
$$K_2 = h f (x_n + \frac{3}{2} h, y_n + \frac{3}{2} k_1)$$
$$y_{n+1} = y_n + \frac{1}{2} k_1 + \frac{1}{2} k_2$$
$$K_1 = h f (x_n \; y_n)$$
$$K_2 = h f (x_n + h, y_n + k_1)$$

V. Third-Order R.K Method derivation & it's stability

$K_1 = h f (x_n + h, y_n + k_1) K_2 = h f (x_n + c_2 h, y_n + a_{21} k_1)$ $K_3 = h f (xn + c_2 h, yn + a_{31} k_1 + a_{32} k_2) K\_i = hf(xn + c_i h, y_n + \sum_j^{i-1} a\, i\, jk\, j$

When r = 1 then only k_1 is needed and 1 yields Euler's formula

Let r = 3 then we need to determine (Eight parameters) w_1 ,w_2 ,w3 , c2, c3, a 31, a 32, a 21.

$\acute{y} = f(x , y) = f$

$$y_{n+1} = y_n + h fn + \frac{1}{2} h^2 (f_x + f_y f) + \frac{1}{2} h^3 (f_{xx} + 2 f f_{xy} + f^2 f_{yy} + f_y f_x + f (f_y)^2)$$

Then expanding the term 1 in Taylor series in the same manner of the

$= h f(x_n + c2h , y_n + a\,31\, k_1 + a\,32\, k_2)$

yn+1 = yn + w1K1 + w2K2 +w3K3

$$y_{n+1} = y_n + w1\, h\, f_n + w2\, h\, fn + w2h2\, (c2\, fX + a21\, ff\, y) + w2\, h\, fn + h^2 (c2\, fX + a21\, ff\, y) + w3\, h\, fn + w3 h^2$$
$$= (c3fX + a31ff\, y\, a32\, ffy) + w3h3\, (c2f\, xx + c3(a31 + a32)f2\, fXy + \frac{1}{2} a31 + a32)\, 2\, fY^2 ffy + a32(c3fX + a21ffY)fY)$$

$$w1 + w2 + w3 = 1 \qquad c2w2 + c3w3 = \frac{1}{2}$$

The $\frac{1}{2} a_{21} w_2 + \frac{1}{2} (a_{31} + a_{32}) \, 2 \, w_3 = \frac{1}{2}$

$a_{32} a_{21} w_3 = \frac{1}{6}$, $a_{32} c_3 w_3 = \frac{1}{6}$ In general we observed that the parameters and due to the above analysis will always lead to the following identities

$$\sum_{i=1}^{r} w_i = 1$$

$$c_i = \sum_{j=1}^{i-1} a_{ij} \qquad i \neq 1 \;\; i = 2, \ldots\ldots , r$$

These lead us to the following well known 3 stage methods

yn+1 = yn + [K1 + 4K2 +K3]

$K_1 = h * f (x_n, y_n) \; K_2 = h * f (xn + \frac{1}{2} h, yn + \frac{1}{2} k_1)$
$K_3 = h * f (xn + h, yn - K_1 + 2K_2)$

$$y_{n+1} = yn + \frac{1}{6} [k_1 + 4 k_2 + 2 k_3]$$

$K_1 = h * f (x_n, y_n) \; K_2 = h * f \left(xn + \frac{1}{2} h, yn \frac{1}{2} k_1\right)$
$K_3 = h * f (xn + h, yn - K_1 + 2K_2)$

$$y_{n+1} = yn + \frac{1}{9} [2 k_1 + 3 k_2 + 4 k_3]$$
$$K_1 = h * f (x_n \; y_n)$$

$$K_2 = h *(x_n + \frac{1}{2} h, y_n + \frac{1}{2} k_1)$$

$$K_3 = h*(x_n + \frac{3}{4} h, y_n + \frac{3}{4} k_2)$$

This last set of parameters gives the modified Euler algorithm (the modified Euler is the special case of second - order Rung- Kutta method

Shows the computation for Runge-Kutta fourth- order method

VI. Implicit Runge-Kutta Methods

The general form for implicit Rung- Kutta method [13]

$$Y_i = y_n + h \sum_{j=1}^{R} a_{ij} f(x_n + c_j h, Y_j)$$

$$Y_{n+1} = y_n + h \sum_{i=1}^{R} b_i f(x_n + c_i h, Y_i)$$

We have the Butcher array

$$\begin{array}{l|l}
\sum_{i=1}^{R} a_{1i} = c_1 & a_{11} \;\; a_{12} \;\; a_{13} \; a_{1r} \\
\sum_{i=1}^{R} a_{2i} = c_2 & a_{21} \;\; a_{22} \;\; a_{23} \; a_{2r} \\
\cdots\cdots\cdots & \cdots\cdots\cdots\cdots \\
\cdots\cdots\cdots & \cdots\cdots\cdots\cdots \\
\sum_{i=Ri}^{R} a_{2i} = c_R & a_{R1} \;\; a_{R2} \;\; a_{R3} a_{Rr}
\end{array}$$

$\sum_{i=1}^{r} w_i = 1$, $c = \sum_{j=1}^{i-1} a_{ij}$

VII. Commutation of r.k fourth-order is a further advance in efficiency in what we can refer as a group of methods due to German Mathematicians Runge-Kutta. The fourth-order Runge-Kutta methods(as shown below) are widely used in computer solutions to ordinary differential equations [10] [11,12]

$$y_{n+1} = y_n + [K_1 + 2K_2 + 2K_3 + K_4]$$
$$K_1 = h * f (x_n \, y_n)$$

$$K_2 = h * f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_1)$$

$$K_3 = h * f(x_n + \frac{1}{2}h, y_n + \frac{1}{2}k_2)$$

$$K_4 = h * f(x_n + h, y_n + k_2)$$

The general form for implicit Rung- Kutta method [13]

$$Y_i = y_n + h\sum_{j=1}^{R} a_{ij} f(x_n + c_j h, Y_j)$$

$$Y_{n+1} = y_n + h\sum_{i=1}^{R} b_i f(x_n + c_i h, Y_i)$$

To explain that equations researchers need to use famous array called Butcher array with which one can justify the parameters required for implicit and/or explicit Rung - Kutta method due to the following conditions

[C1] $A(\zeta) = \sum_{j=1}^{R} a_{ij} c_j^{k-1} = \frac{1}{k} c_i^k$

[C2] $B(\zeta) = \sum_{j=1}^{R} b_i c_i^{k-1} = \frac{1}{k} for\ k \le \zeta$

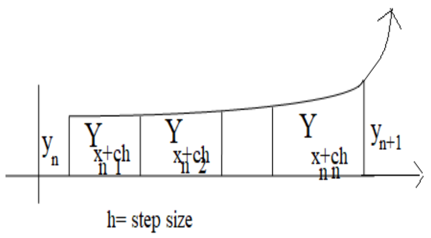
[C3] $D(\zeta) = \sum_{j=1}^{R} b_I c_j^{k-1} a_{ij} = \frac{b_i(1-c_i^k)}{k}$

for j=1,2,3…..r and k<=ζ

K is positive integer and ζ is an integration parameter

Then using C1 for s=3 and K= 1,2

$$\begin{array}{l|llll}
\sum_{i=1}^{R} a_{1i} = c_1 & a_{11} & a_{12} & a_{13} & a_{1r} \\
\sum_{i=1}^{R} a_{2i} = c_2 & a_{21} & a_{22} & a_{23} & a_{2r} \\
\cdots & \cdots & & & \\
\cdots & \cdots & & & \\
\cdots & \cdots & & & \\
\hline
\sum_{i=Ri}^{R} a_{2i} = c_R & a_{R1} & a_{R2} & a_{R3} & a_{Rr} \\
\sum_{i=1}^{R} b_i = 1 & b_1 & b_2 & b_3 & b_r
\end{array}$$



Where K_1= f(x_n , <u>k</u>)y_1,y_2,y_3,x_n ,x_{n+1} ,y_n
h=step size

3. STUDY METHODOLOGY

Our methodology is to satisfy a 3 stages method of order 5 with the same diagonal elements equal to λ which implies that a_{11}= a_{22} =a_{33} =λ this leaves us with 7 unknown parameters in Butcher array, namely a_{13}, a_{12} ,a_{23} , a_{21}, a_{31} ,a_{32} ,in addition to λ

We may represent the method in term of λ [the method parameter] and this leaves us with 6 unknown in six equations sketched above.
Having obtained the six Unknown family of 3 stages method is actually defined then we should have to investigate the methods parameter in line with the error estimate
let a11= a22 =a33 =λ then equations

$$b_1\lambda + b_2 a_{12} + b_3 a_{13} = b_1(1 - c_1) \quad (1)$$

$$b_1 a_{12} + b_2\lambda + b_3 a_{23} = b_2(1 - c_2) \quad (2)$$

$$b_1 a_{13} + b_2 a_{22} + b_3\lambda = b_3(1 - c_3) \quad (3)$$

$$b_1 c_1\lambda + b_2 c_2 a_{12} + b_3 c_3 a_{13} = \frac{1}{2} b_1(1 - c_1^2) (4)$$

$$b_1 c_1 a_{11} + b_2 c_2\lambda + b_3 c_3 a_{13} = \frac{1}{2} b_2(1 - c_2^2) \ (5)$$

$$b_1 c_1 a_{11} + b_2 c_2 a_{12} + b_3 c_3\lambda = \frac{1}{2} b_3(1 - c_3^2) \ (6)$$

Equation 1&4 can be solved for a_{21} ,a_{31}(in terms of λ) similarly a_{12},a_{32} can be obtained from eq. 2&5 finally equation 3&6 will yield a_{13},a_{23}

Subtract 1 from 4

$$b_1\lambda(1 - c_1) + b_2 a_{21}(1 - c_2) + b_3 a_{31}(1 - c_3) = b_1(1 - c_1)[1 - \frac{1}{2}(1 - c_1)]$$

$$b_2 a_{21}(1 - c_2) = b_1(1 - c_1)[\frac{1}{2} - \frac{1}{2}c_1)] - b_1\lambda(1 - c_1)$$

$$b_2 a_{21}(1 - c_2) = b_1(1 - c_1)[\frac{1}{2} - \frac{1}{2}c_1)\lambda]$$

$$a_{21} = \frac{b_1(1-c_1)[\frac{1}{2} - \frac{1}{2}c_1 - \lambda]}{b_2(1-c_2)}$$

Then by substation we get

$$a_{21} = \frac{\frac{(16-\sqrt{6})}{36}(1-\frac{(4-\sqrt{6})}{10})[\frac{1}{2} - \frac{1}{2}\frac{(4-\sqrt{6})}{10} - \lambda]}{\frac{(16+\sqrt{6})}{36}(1-\frac{(4+\sqrt{6})}{10})}$$

$$c_1 = \frac{(4-\sqrt{6})}{10} \quad c_2 = \frac{(4+\sqrt{6})}{10} \quad c_3 = 1$$

$$b_1 = \frac{(16-\sqrt{6})}{36} \quad b_2 = \frac{(16+\sqrt{6})}{36} \quad b_3 = \frac{1}{9}$$

$$a_{21} = \frac{(16-\sqrt{6})(6+\sqrt{6})[(6+\sqrt{6})-20\lambda]}{(6+\sqrt{6})(16-\sqrt{6})*20}$$

$$a_{21} = \frac{(90+10\sqrt{6})[6+\sqrt{6}-20\lambda]}{(90-10\sqrt{6})*20}$$

$$a_{21} = \frac{(29+10\sqrt{6})[6+\sqrt{6}-20\lambda]}{(90-10\sqrt{6})*20}$$

$$a_{21} = \frac{(42+13\sqrt{6}-116\lambda-24\sqrt{6}\,\lambda)}{25*20}$$

$$a_{21} = \frac{(42+13\sqrt{6}-116\lambda-24\sqrt{6}\,\lambda)}{100}$$

To find a_{12} substitute a_{21} in equation 1

$$b_1\lambda + b_2\frac{(42+13\sqrt{6}-116\lambda-24\sqrt{6}\,\lambda)}{100} + b_3 a_{13} = b_1(1-c_1)$$

$$a_{13} = \frac{\left[b_1(1-c_1)-b_1\lambda+b_2\frac{(42+13\sqrt{6}-116\lambda-24\sqrt{6}\,\lambda)}{100}\right]}{b_3}$$

$$b2\,\lambda(1-c2) + b1a12(1-c1) + b3a32(1-c3) = b2(1-c2)[1-\tfrac{1}{2}(1+c2)]$$

$$a_{12} = \frac{\left[\frac{(16+\sqrt{6})}{36}\left(1-\frac{(4+\sqrt{6})}{10}\right)[\frac{1}{2}-\frac{1}{2}\frac{(4+\sqrt{6})}{10}-\lambda\right]}{\frac{(16-\sqrt{6})}{36}\left(1-\frac{(4-\sqrt{6})}{10}\right)}$$

$$a_{12} = \frac{[(16+\sqrt{6})(6-\sqrt{6})[6-\sqrt{6}-20\lambda]}{(16+\sqrt{6})(6-\sqrt{6})*20}$$

$b_1(-a_{23}\frac{[29-6\sqrt{6}]}{25}) + b_2a_{22} + b_3\lambda = b_3(1-c_3)$then after substitute the value of b1, b2 ,b3 ,c1the

$$a_{12} = \frac{[(29-6\sqrt{6})[6-\sqrt{6}-20\lambda]}{20*25}$$

$$a_{12} = \frac{[42-13\sqrt{6}-116\lambda+24\sqrt{6}\lambda]}{100}$$

Equation 2&5 can be solved for a_{12} a_{32} Subtract5 from 2 to find a_{32} substitute a_{12} in equation 1 $b_2\lambda + b_1\frac{[42-13\sqrt{6}-116\lambda+24\sqrt{6}\lambda]}{100} + b_3a_{23} = b_2(1-c_2)$

$$b_3 3\lambda(1-c_3) + b_1a_{13}(1-c_1) + b_2a_{23}(1-c_2) = b_3(1-c_3)[1-\tfrac{1}{2}(1+c_3) = \frac{[b_2a_{23}(1-c_2)]}{b_1(1-c_1)}$$

then after substitute the value of b_1 , b_2 , b_3 ,c_2,then

Equations 3&6 can be solved for a_{13} a_{23} by Subtract 6 from3

$$a_{13} = -a_{23}\frac{[90-10\sqrt{6}]}{[90+10\sqrt{6}]} \qquad a_{13} = -a_{23}\frac{[29-6\sqrt{6}]}{25}$$

since c3 =1 then After substitution $a_{23}\frac{[6\sqrt{6}-4]}{36} = \frac{-\lambda}{9}$

$$a_{23} = \frac{-\lambda}{9} * \frac{36}{[6\sqrt{6}-4]}$$

$$a_{23} = -\lambda * \frac{[3\sqrt{6}+2]}{25}$$

And since

$$a_{13} = -a_{23}\frac{[29-6\sqrt{6}]}{25}$$

$$a_{23} = -a_{13}\frac{25}{(29-6\sqrt{6})}$$

$$b_1(-a_{23}\frac{[29-6\sqrt{6}]}{25}) + b_2a_{22} + b_3\lambda = b_3(1-c_3)$$

$$a_{23}\frac{[6\sqrt{6}-4]}{36} = \frac{-\lambda}{9}$$

$$a_{23} = \frac{-\lambda}{9} * \frac{36}{[6\sqrt{6}-4]}$$

$a_{23} = -\lambda * \frac{[3\sqrt{6}+2]}{25}$ And since

$a_{13} = -a_{23}\frac{[29-6\sqrt{6}]}{25}$ then $a_{13} = -\lambda * \frac{[3\sqrt{6}-2]}{25}$

$$\begin{array}{l|ccc}
\sum_{i=1}^{R} a\,1i = c_1 = \frac{(4-\sqrt{6})}{10} & \lambda & a12 & a13 \\
\sum_{i=1}^{R} a\,2i = c_2 = \frac{(4+\sqrt{6})}{10} & a21 & \lambda & a23 \\
\sum_{i=Ri}^{R} a\,2i = c_3 = 1 & a31 & a32 & \lambda \\
\hline
\end{array}$$

$$\sum_{i=1}^{R} b_i = 1 = \frac{(16-\sqrt{6})}{36} \quad b_2 = \frac{(16+\sqrt{6})}{36} \quad b_3 = \frac{1}{9}$$

our values for the parameters a_{11},a_{12},………a_{33}

$$\begin{array}{l|ccc}
\sum_{i=1}^{R} a\,1i = c_1 = \frac{(4-\sqrt{6})}{10} & \frac{1}{12} & \frac{[97-33\sqrt{6}]}{300} & \frac{[-2+3\sqrt{6}]}{300} \\
\sum_{i=1}^{R} a\,2i = c_2 = \frac{(4+\sqrt{6})}{10} & \frac{[97+33\sqrt{6}]}{300} & \frac{1}{12} & \frac{[-2-3\sqrt{6}]}{300} \\
\sum_{i=Ri}^{R} a\,2i = c_3 = 1 & \frac{[11+6\sqrt{6}]}{24} & \frac{[11-6\sqrt{6}]}{24} & \frac{1}{12} \\
\hline
\end{array}$$

$$\sum_{i=1}^{R} b\,i = 1, b_1 = \frac{(16-\sqrt{6})}{36} \quad b_2 = \frac{(16+\sqrt{6})}{36} \quad b_3 = \frac{1}{9}$$

- **Study algorithm and code**

VIII. Call A sample differential equation "dy/dx = (x - y)/2"…….etc

IX. Finds value of y for a given x using step size h initial value y_0 at x_0.

- call Runge-Kutta(x_0, y, x, h));
- Count number of iterations

using step size or step height h Iterate for n number of iterations

- Apply Runge Kutta Formulas to find next value of y Update next value of y
- Update next value of x

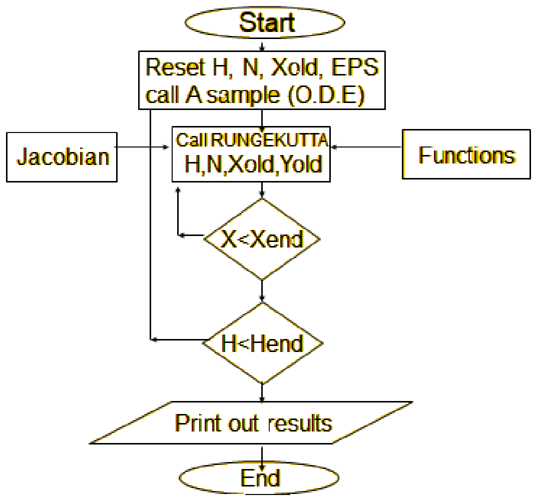


Fig 1 Algorithm flow chart

- **Study Code**

Call A sample differential equation "dy/dx = (x - y)/2"…….etc
Finds value of y for a given x using step size h,initial value y_0 at x_0.
call rungeKutta(x0, y, x, h)); Count number of iterations using step size or step height h Iterate for n number of iterations
Apply Runge Kutta Formulas to find next value of y Update next value of y
Update next value of x

```
// C++ program to implement Runge
// Kutta method #include <iostream>
// A sample differential equation
// "dy/dx = (x - y)/2"
float dydx(float x, float y)
{
return (x - y)/2;
}
float rungeKutta(float x0, float y0,float x, float h)
{
int n = (int)((x - x0) / h); float k1, k2, k3, k4,
float y = y0;
// Iterate for n number of iterations for (int i = 1; i <= n; i++)
{
k1 = h*dy/dx(x0, y);
k2 = h*dy/dx(x0 + 0.5*h, y + 0.5*k1); k3 = h*dy/dx(x0 +
0.5*h, y + 0.5*k2); k4 = h*dy/dx(x0 + h, y + k3);
y = y + (1.0/6.0)*(k1 + 2*k2 + 2*k3 + k4).
// Update next value of x x0 = x0 + h;
}
return y;
                              }
          int main()
                              {
float x0 = 0, y = 1, x = 2, h = 0.2;
Float x0 = 0, y = 1, x = 2, h = 0.2; printf("y(x)
= %f",rungeKutta(x0, y, x, h)); return 0;
}
Float x0 = 0, y = 1, x = 2, h = 0.2; float k1, k2, k3, k4,
float y = y0;
printf("\nThe value of y at x is : %f", rungeKutta(x0, y, x,
h)); return 0;
Finds value of y for a given x using step size h with initial
value y0 at x0.
Call rungeKutta(float x0, float y0, float x, float h)
// Count number of iterations using step size h ; int n =
(int)((x - x0) / h);
Iterate for number of iterations
15
for (int i=1; i<=n; i++)
// Apply Runge Kutta Formulas to find next value of y
k1 = h*dy/dx(x0, y);
k2 = h*dy/dx(x0 + 0.5*h, y + 0.5*k1); k3 = h*dy/dx(x0 +
0.5*h, y + 0.5*k2); k4 = h*dy/dx(x0 + h, y + k3);
// Update next value of y
y = y + (1.0/6.0)*(k1 + 2*k2 + 2*k3 + k4).
// Update next value of x x0 = x0 + h;
return y;
```

4. RESULTS DISCUSSIONS

The implementation of implicit Runge-Kutta (I.R.K) methods for initial value problem (I.V.P) for ordinary differential equations (O.D.E), was our concern in this study. By testing our diagonal implicit R.K methods for series problems see TABLE I-IV, some of them have a known exact solution with which accuracy of the method was tested, our method very efficient and successfully solved these problems in term of run time error. It is Notice that the implicit Runge-Kutta method is much more accurate to solve problems that has no exact solution. As confidentially, we can say Runge-Kutta is much more accurate than other methods it observed that the accuracy of the(I.R.K)approximation that compare the approximation to the exact solution. as shown below.

5. CONCLUSION AND FUTURE STUDIES

Our diagonal Implicit Runge-Kutta methods for 2 & 3 stages requires less time, steps and function evaluations compared to other methods, so we recommend generating parameters for 4th and 5th stages so as to solve more initial value problems for O.D.E.

TABLE I

```
FOR THE***************SOLUTION OF
      F=-Y+SIN(X),DIRKT3T TOL=      1.00000D-02
H = 1.0000000000000000E-004***
  H         X          YN       YTRUE     ERROR  ERROR%
 1.00D-04 1.0D-04 9.9991D-019.9990D-011.1113D-051.1114D-
03
 1.00D-04   2.0D-04 9.9982D-019.9980D-012.2223D-052.2227D-
03
 1.00D-04   3.0D-04 9.9973D-019.9970D-013.3329D-053.3339D-
03
 1.00D-04   4.0D-04 9.9964D-019.9960D-014.4433D-054.4450D-
03
 1.00D-04   5.0D-04 9.9956D-019.9950D-015.5533D-055.5560D-
03
 1.00D-04   6.0D-04 9.9947D-019.9940D-016.6630D-056.6670D-
03
 1.00D-04   7.0D-04 9.9938D-019.9930D-017.7723D-057.7778D-
03
 1.00D-04   8.0D-04 9.9929D-019.9920D-018.8814D-058.8885D-
03
 1.00D-04   9.0D-04 9.9920D-019.9910D-019.9901D-059.9991D-
03
```

TABLE II

```
SOLUTION OF F1=-1/Y1,F2=1/Y2,DIRK3SS AT
TOL=    1.00000D-02
H = 1.0000000000000000E-004***
  X       Y(1)       YT1      ERROR  Y(2)  YT2       ERROR
1.0D-04 1.0001D+00 1.0001D+00 1.1116D-059.9991D-019.9990D-011.1106D-05
2.0D-04 1.0002D+00 1.0002D+00 2.2240D-059.9982D-019.9980D-012.2200D-05
3.0D-04 1.0003D+00 1.0003D+00 3.3373D-059.9973D-019.9970D-013.3283D-05
4.0D-04 1.0004D+00 1.0004D+00 4.4515D-059.9964D-019.9960D-014.4355D-05
5.0D-04 1.0004D+00 1.0005D+00 5.5665D-059.9956D-019.9950D-015.5415D-05
6.0D-04 1.0005D+00 1.0006D+00 6.6825D-059.9947D-019.9940D-016.6465D-05
7.0D-04 1.0006D+00 1.0007D+00 7.7993D-059.9938D-019.9930D-017.7503D-05
8.0D-04 1.0007D+00 1.0008D+00 8.9170D-059.9929D-019.9920D-018.8530D-05
9.0D-04 1.0008D+00 1.0009D+00 1.0036D-049.9920D-019.9910D-019.9546D-05
```

TABLE III

```
FOR THE***************SOLUTION OF
F=-200*(Y(1)-(10-(10+X)*DEXP(-X)))
D1D2SM,TOL=1.00000D-02
H = 1.0000000000000000E-004***
  X          YN          YTRUE       ERROR    ERROR%
0.0D+001.0000D+011.0000D+010.0000D+000.0000D+00
1.0D-018.6114D-018.6114D-011.5987D-141.5987D-12
2.0D-011.6489D+001.6489D+002.3565D-142.3565D-12
3.0D-012.3696D+002.3696D+004.7228D-144.7228D-12
4.0D-013.0287D+003.0287D+005.7038D-145.7038D-12
5.0D-013.6314D+003.6314D+006.0778D-146.0778D-12
6.0D-014.1826D+004.1826D+006.1582D-146.1582D-12
7.0D-014.6865D+004.6865D+006.1593D-146.1593D-12
8.0D-015.1472D+005.1472D+006.0912D-146.1593D-12
9.0D-015.5684D+005.5684D+005.9335D-146.1593D-12
1.0D+005.9533D+005.9533D+005.7438D-146.1593D-12
```

1.0D+005.9533D+005.9533D+005.7438D-146.1593D-12

TABLE IV

```
FOR THE***************SOLUTION OF
F=-200*(Y(1)-(10-(10+X)*DEXP(-X)))
D1D2SM,TOL=1.00000D-02
H = 1.0000000000000000E-004***
  X          YN          YTRUE       ERROR    ERROR%
 0.0D+00 1.0000D+011.0000D+010.0000D+000.0000D+00
 1.0D-01 8.6114D-018.6114D-011.5987D-141.5987D-12
 2.0D-01 1.6489D+001.6489D+002.3565D-142.3565D-12
 3.0D-01 2.3696D+002.3696D+004.7228D-144.7228D-12
 4.0D-01 3.0287D+003.0287D+005.7038D-145.7038D-12
 5.0D-01 3.6314D+003.6314D+006.0778D-146.0778D-12
 6.0D-01 4.1826D+004.1826D+006.1582D-146.1582D-12
 7.0D-01 4.6865D+004.6865D+006.1593D-146.1593D-12
 8.0D-01 5.1472D+005.1472D+006.0912D-146.1593D-12
 9.0D-01 5.5684D+005.5684D+005.9335D-146.1593D-12
 1.0D+00 5.9533D+005.9533D+005.7438D-146.1593D-12
```

VI. REFERENCES

Journal papers

[1] D. F. Griffiths and D. J. Higham, Numerical methods for ordinary differential equations: initial value problems. Springer, 2010.

[2] J. C. Butcher, "A transformed implicit Runge-Kutta method," *Journal of the ACM (JACM),* vol. 26, no. 4, pp. 731-738, 1979.

[3] G. Kirlinger, "Linear multistep methods applied to stiff initial value problems—A survey," *Mathematical and Computer Modelling,* vol. 40, no. 11-12, pp. 1181-1192, 2004.

[4] J. Cash, "The integration of stiff initial value problems in ODEs using modified extended backward differentiation formulae," *Computers & mathematics with applications,* vol. 9, no. 5, pp. 645-657, 1983.

[5] J. D. Lambert, Numerical methods for ordinary differential systems. Wiley New York, 1991.

[6] C. F. Gerald, *Applied numerical analysis*. Pearson Education India, 2004.

[7] S. N. Jator, "Solving second order initial value problems by a hybrid multistep method without predictors," *Applied mathematics and Computation,* vol. 217, no. 8, pp. 4036-4046, 2010.

[8] K. Burrage and F. Chipman, "The stability properties of singly-implicit general linear methods," 1985.

[9] H. M. Ijam, Z. B. Ibrahim, Z. A. Majid, and N. Senu, "Stability analysis of a diagonally implicit scheme of block backward differentiation formula for stiff pharmacokinetics models," *Advances in Difference Equations,* vol. 2020, no. 1, pp. 1-22, 2020.

[10] J. E. Miller, D. G. Moursund, and C. S. Duris, *Elementary Theory & Application of Numerical Analysis*. Courier Corporation, 2011.

[11]A. C. Hindmarsh, "Ordinary differential equation system solver," Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States)1992.

[12]L. Lapidus and J. H. Seinfeld, *Numerical solution of ordinary differential equations*. Academic press, 1971.

[13]G. K. Gupta, R. Sacks-Davis, and P. E. Tescher, "A review of recent developments in solving ODEs," *ACM Computing Surveys (CSUR),* vol. 17, no. 1, pp. 5-47, 1985.

[14]J. C. Butcher, "Coefficients for the study of Runge-Kutta integration processes," *Journal of the Australian Mathematical Society,* vol. 3, no. 2, pp. 185-201, 1963.

[15] J. G. Oghonyon, S. A. Okunuga, K. S. Eke, and O. A. Odetunmibi, “Block Milne’s Implementation For Solving Fourth Order Ordinary Differential Equations”, Eng. Technol. Appl. Sci. Res., vol. 8, no. 3, pp. 2943–2948, Jun. 2018.

[16] A. Abbasi and F. Mohd Hashim, “Evaluating Pressure in Deepwater Gas Pipeline for the Prediction of Natural Gas Hydrate Formation”, Eng. Technol. Appl. Sci. Res., vol. 9, no. 6, pp. 5033–5036, Dec. 2019.

[17] M. D. Monsia and Y. J. F. Kpomahou, “Simulating Nonlinear Oscillations of Viscoelastically Damped Mechanical Systems”, Eng. Technol. Appl. Sci. Res., vol. 4, no. 6, pp. 714–723, Dec. 2014.

The authors would like to thank the Deanship of Scientific Research at Umm Al-Qura University for supporting this work by Grant Code: (22UQU4260103DSR01)"