FUDMA Journal of Sciences (FJS)
ISSN online: 2616-1370
ISSN print: 2645 - 2944
Vol. 4 No. 3, September, 2020, pp 513 – 522
DOI: https://doi.org/10.33003/fjs-2020-0403-371

A DIAGONALLY IMPLICIT RUNGE-KUTTA-NYSTROM (RKN) METHOD FOR SOLVING SECOND ORDER ODES ON PARALLEL COMPUTERS

Imoni, S. O.

Department of Mathematical Sciences, Federal University Lokoja, Kogi State, Nigeria

*Corresponding Author's Email: imoni4u@yahoo.com, Sunday.imoni@fulokoja.edu.ng

ABSTRACT

In this paper, diagonally implicit Runge-Kutta-Nystrom (RKN) method of high-order for the numerical solution of second order ordinary differential equations (ODE) possessing oscillatory solutions to be used on parallel computers is constructed. The method has the properties of minimized local truncation error coefficients as well as possessing non-empty interval of periodicity, thus suitable for oscillatory problems. The method was tested with standard test problems from the literature and numerical results compared with the analytical solution to show the advantage of the algorithm.

Keywords: ordinary differential equations, initial value problems, Runge-Kutta-Nystrom method, parallel method, oscillatory problems, analytical solution.

INTRODUCTION

Runge-Kutta–Nystrom method is widely used for the numerical approximation of the initial value problem (IVP)

$$y'' = f(x, y), y(x_o) = y_o, y'(x_o) = y'_0 \qquad (1)$$

having oscillatory solution often arises in application, for example, in astronomy, seismology and when a second order hyperbolic partial derivative equation is semi-discretized with respect to space variables. RKN method is usually employed to approximate (1) at a discrete set of points (x_n, y_n, y'_n). The form of this method is given (Sharp *et al.* (1990), Van de Houwen and Sommeijer (1989), Franco and Gomez (2009)) by

$$\begin{aligned} y_{n+1} &= y_n + h \, y'_n + h^2 \sum_{j=1}^{s} b_j f_j \\ y'_{n+1} &= y'_n + h \sum_{j=1}^{s} b'_j f_j \end{aligned} \qquad (2)$$

where

$$f_j = f\left(x_n + c_j h , \; y_n + c_j h y'_n + h^2 \sum_{k=1}^{s} a_{jk} f_k \right)$$

The RKN parameters a_{jk}, b_j, b_j and c_j are assumed to be real and a_{jk} are the stage weights b_j weights , c_j the nodes and s number of stages. In most methods, the c_j satisfy the row simplifying assumption

$$\frac{1}{2} c_j^2 = \sum_{k=1}^{s} a_{jk} \qquad j = 1, \cdots, s \qquad (3)$$

All the coefficients of the method can characterised by the Butcher tableau, Butcher (1964)

$$\begin{array}{c|c} c & A \\ \hline & b^T \\ & b'^T \end{array}$$

where

$c=[c_1,\cdots,c_s]^T$, $[b_1,\cdots,b_s]^T$, $b_1'=[b_1',\cdots,b_s']^T$ and $A=\lfloor a_{ij}\rfloor$ with $c\in\mathbb{R}$, $b^T, b'^T \in\mathbb{R}^s$ and $A\in\mathbb{R}^{sxs}$.

RKN methods are divided into two broad classes: explicit $(a_{jk}=0,\ k\geq j)$ and implicit ($a_{jk}\neq 0,\ k\geq j$). The later contains the class of diagonally implicit methods for which $(a_{jk}=0,\ \ k>j)$ and the a_{jj} are equal. In this article, the consideration is on diagonally implicit methods.

In the literature, several high-order diagonally implicit Runge-Kutta-Nystrom (DIRKN) methods have been proposed for the integration of the IVP (1) on one-processor computers. For example, the two-stage and three-stage DIRKN methods orders three and four of Sharp et al. (1990), the two-stage DIRKN methods of order four of Sommeijer (1987), DIRKN methods for oscillatroty problems by Van der Houwen and Sommeijer (1989), the RKN methods of orders three for solving fuzzy differential equations of Kanagarajam and Sambath (2010). However, parallel IVP solvers arise from the need to solve many substantial problems faster than is currently possible. The computational time on a conventional sequential machine is so large that it affects the productivity of scientists and engines working on the design of complex systems.

In this study, parallel diagonally implicit Runge-Kutta-Nystrom (PDIRKN) method is presented for the approximation of the IVP (1).

Parallel computers are computers with multiple processors and this facility helps to speed up the computations in the solution of ODEs. This is particularly useful for very large problems, costly function evaluations, problems with long integration intervals or for fast real-time simulations. A second motivation is the desire to make a code, with the help of parallel computations, not necessarily faster, but more robust and reliable, Hairer *et al.* (1993). In attempts to solve (1), three types of parallelism have been identified:

(i) Parallelism across the method
(ii) Parallelism across the system (space)
(iii) Parallelism across the steps

Parallelism across the method is to perform several function evaluations concurrently on different processors. The technique of parallelism across the system is via the decomposition of a problem into sub-problems which can then be solved in parallel with the processors communicating as appropriate. Parallelism across the step in which generation integration steps are perform concurrently with a given numerical method (Burrage (1997), Amodio and Brugnano (2008) for more details)

In this article, the development of PDIRKN method for solving the IVPs associated with the special second order IVP (1) in the first category, which is parallelism across the method is investigated.

Crisci, *et al.* (1993), introduced fully parallel RKN methods by imposing that the matrix A is diagonal. The method has algebraic order two in two stages. This means that the stages can be evaluated concurrently using two-processor machine. Other contributors to the development of parallel methods include Amodio and Brugnano (2008), which examined possible extensions of parallel methods previously proposed in the mid-nineties, Amodio and Brugnano (1997) which analysed its connections with subsequent approaches to the parallel solution of ODE-IVPs. Sommeijer (1993) describes the construction of explicit RKN methods for parallel computers.

In order to test the performance of this method for oscillatory problems, it is of interest to consider the test problem (Tsitouras, (1998))

$$y''=-\omega^2 y,\quad y(0)=1, y'(0)=i\omega, \omega\in\mathbb{R} \tag{4}$$

Application of the RKN method (2) to problem (4) yields the following recursive relation (Imoni (2017))

$$\begin{pmatrix} y_{n+1} \\ hy'_{n+1} \end{pmatrix} = R(Z)\begin{pmatrix} y_n \\ hy'_n \end{pmatrix} \tag{5}$$

where

$$R(z) = \begin{bmatrix} 1+zb^T(I-zA)^{-1}c & 1+zb^T(1-zA)^{-1}c \\ zb'^T(I-zA)^{-1} & 1+zb'^T(I-zA)^{-1}c \end{bmatrix} \tag{6}$$

and

$$A=\{a_{ik}\}_{j,k=1}^{s}\quad e=[1,\cdots,1]^T,\ b=[b_1,\cdots,b_s]^T\ b'=[b_1',\cdots,b_1']^T\ ,c=[c_1,\cdots,c_s]^T$$

The matrix $R(z)$ which determines the stability of the method is called the amplification matrix. Following Van der Houwen and Sommeijer (1989), we introduced the functions $s(z)$ and $p(z)$ with

$$s(z) = \text{trace}\,(R(z)) \text{ and } p(z) = \det\,(R(z)) \tag{7}$$

The characteristic equation corresponding to (6) is of the form

$$\zeta^2 - s(z)\zeta + p(z) = 0 \tag{8}$$

An essential property for computing periodic solution of (1) is the situation where the eigenvalues $\zeta_{1,2}$ are on the unit circle.

Definition: An RKN method has periodicity interval $I_z = (0,\ z_0)$ if the roots of its characteristic equation $\zeta_{1,2}$ are on the unit circle and $\zeta_1 \neq \zeta_2$, $\forall z \in (0,\ z_0)$, z_0 is called the stability boundary (Van de Houwen and Sommeijer (1989))

Construction of the new PDIRKN Method

A Six stage, 3-parallel, 3-processor sixth order DIRKN method is investigated. The method has the sparsity pattern and diagraph shown in figure 1

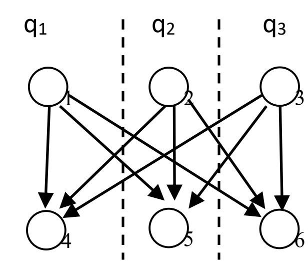
Runge –Kutta –Nystrom Matrix	Digraph
$\begin{bmatrix} x & 0 & 0 & 0 & 0 & 0 \\ 0 & x & 0 & 0 & 0 & 0 \\ 0 & 0 & x & 0 & 0 & 0 \\ x & x & x & x & 0 & 0 \\ x & x & x & 0 & x & 0 \\ x & x & x & 0 & 0 & x \end{bmatrix}$	

Figure 1: The 6-Stage PDIRKN Matrix and Digraph

The symbol $\times$ denotes non-zero elements, q_1, q_2 and q_3 denotes the number of processors Imoni (2017)

For this process with s = 6, p = 6, following Felberg (1972) the following order conditions are to be satisfied considering the simplifying assumption (3)

Order two: $$\sum_{i=1}^{6} b_i = b_1 + b_2 + b_3 + b_4 + b_5 + b_6 = \frac{1}{2} \tag{9}$$

Order three: $$\sum_{i=1}^{6} b_i c_i = b_1c_1 + b_2c_2 + b_3c_3 + b_4c_4 + b_5c_5 + b_6c_6 = \frac{1}{6} \tag{10}$$

Order four: $$\sum_{i=1}^{6} b_i c_i^2 = b_1c_1^2 + b_2c_2^2 + b_3c_3^2 + b_4c_4^2 + b_5c_5^2 + b_6c_6^2 = \frac{1}{12} \tag{11}$$

Order five: $$\sum_{i=1}^{6} b_i c_i^3 = b_1c_1^3 + b_2c_2^3 + b_3c_3^3 + b_4c_4^3 + b_5c_5^3 + b_6c_6^3 = \frac{1}{20} \tag{12}$$

$$\begin{aligned} \sum_{i,j=1}^{6} b_i a_{ij} c_j = {} & b_1a_{11}c_1 + b_2a_{21}c_1 + b_2a_{22}c_2 + b_3a_{31}c_1 + b_3a_{32}c_2 + b_3a_{33}c_3 + \\ & b_4a_{41}c_1 + b_4a_{42}c_2 + b_4a_{43}c_3 + b_4a_{44}c_4 + b_5a_{51}c_1 + b_5a_{52}c_2 + \\ & b_5a_{53}c_3 + b_5a_{54}c_4 + b_5a_{55}c_5 + b_6a_{61}c_1 + b_6a_{62}c_2 + b_6a_{63}c_3 + \\ & b_6a_{64}c_4 + b_6a_{65}c_5 + b_6a_{66}c_6 = \frac{1}{120} \end{aligned} \tag{13}$$

Order six: $$\sum_{i=1}^{6} b_i c_i^4 = b_1c_1^4 + b_2c_2^4 + b_3c_3^4 + b_4c_4^4 + b_5c_5^4 + b_6c_6^4 = \frac{1}{30} \tag{14}$$

$$\begin{aligned}\sum_{i,j=1}^{6} b_i c_i a_{ij} c_j = {} & b_1a_{11}c_1^2 + b_2a_{21}c_1c_2 + b_2a_{22}c_2 + b_2a_{22}c_2^2 + b_3a_{31}c_1c_3 + \\ & b_3a_{32}c_2c_3 + b_3a_{33}c_3^2 + b_4a_{41}c_1c_4 + b_4a_{42}c_2c_4 + b_4a_{43}c_3c_4 + \\ & b_4a_{44}c_4 + b_5a_{51}c_1c_5 + b_5a_{52}c_2c_5 + b_5a_{53}c_3c_5 + b_5a_{54}c_4c_5 + \\ & b_5a_{55}c_5^2 + b_6a_{61}c_1c_6 + b_6a_{62}c_2c_6 + b_6a_{63}c_3c_6 + b_6a_{64}c_4c_6 + \\ & b_6a_{65}c_6 + b_6a_{66}c_6^2 = \frac{1}{180}\end{aligned} \tag{15}$$

$$\begin{aligned}\sum_{i,j}^{6} b_i c_i a_{ij} c_j^2 = {} & b_1a_{11}c_1^2 + b_2a_{21}c_1^2 + b_2a_{22}c_2^2 + b_3a_{31}c_1^2 + b_3a_{32}c_2^2 + \\ & b_3a_{32}c_3^2 + b_4a_{41}c_1^2 + b_4a_{42}c_2^2 + b_4a_{43}c_3^2 + b_4a_{44}c_4^2 + b_5a_{51}c_1^2 + \\ & b_5a_{52}c_2^2 + b_5a_{53}c_3^2 + b_5a_{54}c_4^2 + b_5a_{55}c_5^2 + b_6a_{61}c_1^2 + b_6a_{62}c_2^2 \\ & + b_6a_{63}c_3^2 + b_6a_{64}c_4^2 + b_6a_{65}c_5^2 + b_6a_{66}c_6^2 = \frac{1}{360}\end{aligned} \tag{16}$$

Assumptions: $$\frac{1}{2}c_j^2 = \sum_{k=1}^{6} a_{jk} \, (j = 1,\ldots,6) \tag{17}$$

There are 14 equations in 27 unknowns. Thus there are 13 free parameters chosen to be $c_i, c_2, c_3, c_4, c_5, c_6, b_6, a_{42}, a_{44}, a_{53}, a_{55}, a_{62}$ and a_{66}.

The process in the derivation of the method is the following:

1. Solve equations (9) – (12) and (14) to obtain

$$b_1 = \frac{A}{60(c_1-c_2)(c_1-c_3)(c_1-c_4)(c_1-c_5)}, b_2 = \frac{B}{60(c_1-c_2)(c_2-c_3)(c_2-c_4)(c_2-c_5)}$$

$$b_3 = \frac{C}{60(c_1-c_3)(c_2-c_3)(c_3-c_4)(c_3-c_5)} \quad b_4 = \frac{D}{60(c_3-c_4)(-c_1+c_4)(-c_2+c_4)(c_4-c_5)}$$

$$b_5 = \frac{E}{60(c_3-c_5)(-c_1+c_5)(-c_2+c_5)(-c_4+c_5)}$$

2. Then use the assumption (17) to obtain

$$a_{11} = \frac{1}{2}c_1^2,\ a_{22} = \frac{1}{2}c_2^2,\ a_{33} = \frac{1}{2}c_3^2, a_{41} = \frac{1}{2}c_4^2 - a_{42} - a_{43} - a_{44}, a_{51} = \frac{1}{2}c_5^2 - a_{52} - a_{53} - a_{55},$$

$$a_{61} = \frac{1}{2}c_6^2 - a_{62} - a_{63} - a_{66}$$

3. Equation (13) is then use to obtain

$$a_{43} = \frac{F}{360b_4(c_4-c_3)(c_2-c_3(c_4-c_6))}$$

4. Solve equation (15) to obtain

$$a_{52} = \frac{G}{120b_5(c_1-c_2)(c_2-c_3)}$$

5. Then use equation (16) to get

$$a_{63} = \frac{H}{360b_6(c_1 - c_3)(c_2 - c_3(c_4 - c_6))}$$

$$A = 2 - 3c_4 - 3c_5 + 5c_4c_5 - 60b_6c_4c_5c_6^2 + 60b_6c_4c_6^3 + 60b_6c_5c_6^3 - 60b_6c_6^4 + c_3(-3 + 60b_6c_6^3 + c_5(5 - 60b_6c_6^2) + 5c_4(1 - 12b_6c_6^2 + 2c_5(-1 + 6b_6c_6))) + c_2(-3 + 5c_5 - 60b_6c_5c_6^2 + 60b_6c_6^3 5c_3(-1 + 12b_6c_6^2 + c_5(2 - 12b_6c_6)\ 2c_4(1 + (-3 + 6b_6) + c_5 - 6b_6c_5)) + 5c_4(1 - 12b_6c_6^2 + 2c_5(-1 + 6b_6c_6)))$$

$$B = -2 + 3c_4 + 3c_5 - 5c_4c_5 - 60b_6c_4c_5c_6^2 - 60b_6c_4c_6^3 - 60b_6c_5c_6^3 + 60b_6c_6^4 + c_3(3 - 60b_6c_6^3 + c_5(-5 + 60b_6c_6^2) + c_4(-5 + 60b_6c_6^2 + c_5(10 - 60b_6c_6)) + c_1(3 - 5c_5 + 60b_6c_5c_6^2 - 60b_6c_6^3 + c_4(-5 + 60b_6c_6^2 + c_5(10 - 60b_6c_6)) + 5c_3(-1 + 12b_6c_6^2 + c_5(2 - 12b_6c_6) + 2c_4(1 + (-3 + 6b_6)(5 - 6b_6c_6))))$$

$$C = (2 - 3c_3 - 3c_5 + 5c_4 - 60b_6c_4c_5c_6^2 + 60b_6c_4c_6^3 + 60b_6c_4c_6^3 + 60b_6c_5c_6^3 - 60b_6c_6^4 + c_2(-3 + 60b_6c_6^3 + c_5(5 - 60b_6c_6^2) + 5c_4(1 - 12b_6c_6^2 + 2c_5(-1 + 6b_6c_6))) + c_1(-3 + 5c_5 - 60b_6c_5c_6^2 + 60b_6c_6^3 - 5c_2(-1 + 12b_6c_6^2 + c_5(2 - 12b_6c_6) + 2c_4(1 + (-3 + 6b_6)c_5 - 6b_6c_6)) + 5c_4(1 - 12b_6c_6^2 + 2c_5(-1 + 6b_6c_6))))$$

$$D = (-2 + 3c_3 + 3c_5 - 5c_3c_5 + 60b_6c_3c_5c_6^2 - 60b_6c_4c_6^3 - 60b_6c_4c_6^3 + 60b_6c_5c + 60b_6c_6^4 + c_2(3 - 60b_6c_6^3 + c_5(-5 + 60b_6c_6^2) + c_3(-5 + 60b_6c_6^2 + c_5(10 - 60b_6c_6))) + c_1(3 - 5c_5 + 60b_6c_5c_6^2 - 60b_6c_6^3 + c_3(-5 + 60b_6c_6^2 + c_5(10 - 60b_6c_6)) + 5c_2(-1 + 12b_6c_6^2 + c_5(2 - 12b_6c_6) + 2c_3(1 + (-3 + 6b_6)c_5 - 6b_6c_6))))$$

$$E = (-2 + 3c_3 + 3c_4 - 5c_3c_4 + 60b_6c_3c_4c_6^2 - 60b_6c_3c_6^3 - 60b_6c_4c_6^3 + 60b_6c_4c_6^3 + 60b_6c_6^4 + c_2(3 - 60b_6c_6^3 + c_4(-5 + 60b_6c_6^2) + c_3(-5 + 60b_6c_6^2 + c_4(10 - 60b_6c_6))) + c_1(3 - 5c_4 + 60b_6c_4c_6^2 - 60b_6c_6^3 + c_3(-5 + 60b_6c_6^2 + c_4(10 - 60b_6c_6)) + 5c_2(-1 + 12b_6c_6^2 + c_4(2 - 12b_6c_6) + 2c_3(1 + (-3 + 6b_6)c_4 - 6b_6c_6))))$$

$$F = 180b_2c_2^5 - 180b_3c_3^5 - 180b_2c_2^4(c_3 + c_5) + 180b_2c_2^3(c_3c_5 + c_1(c_5 - c_6) - 360a_5 3b_5c_3^2(c_5 - c_6) + 180b_3c_1c_3^3(c_5 - c_6) + 180b_3c_3^4c_6 + c_2(-2 - 360a_{42}b_4c_1c_4 - 360a_{44}b_4c_1c_4 - 360a_{42}b_4c_3c_4 + 360a_{44}b_4c_4^2 + 180b_4c_1c_4^3 + 360a_{42}b_4c_1c_5 - 360a_{53}b_5c_1c_5 - 360a_{55}b_5c_1c_5 + 360a_{62}b_6c_1c_5 + 360a_{42}b_4c_3c_5 + 360a_{53}b_5c_3c_5 + 360a_{62}b_6c_3c_5 + 360a_{55}b_5c_5^2 + 180b_5c_1c_5^3 + 180b_1c_1^3(c_1 - c_6) + 180b_3c_3^3(c_3 - c_6) + 3c_6 + 360a_{44}b_4c_1c_6 + 360a_{53}b_5c_1c_6 + 360a_{55}b_5c_1c_6 - 360a_{62}b_6c_1c_6 - 360a_{53}b_5c_3c_6 - 360a_{62}b_6c_3c_6 - 360a_{44}b_4c_4c_6 - 180b_4c_1c_4^2c_6 - 360a_{55}b_5c_5c_6 - 180b_5c_1c_5^2c_6) + c_3(2 + 360a_{44}b_4c_1c_4 - 360a_{44}b_4c_4^2 - 180b_4c_1c_4 - 180b_1c_1^3(c_1 - c_5) + 360a_{42}b_4c_1(c_4 - c_5) - 3c_5 - 360a_{44}b_4c_1c_5 + 360a_{53}b_5c_1c_5 - 360a_{62}b_6c_1c_5 - 360a_{66}b_6c_1c_5 + 360a_{44}b_4c_4c_5 + 180b_4c_1c_4^2c_5 - 360a_{53}b_5c_1c_6 + 360a_{62}b_6c_1c_6 + 360a_{66}b_6c_1c_6 + 360a_{66}b_6c_5c_6 - 360a_{66}b_6c_6^2 + 180b_6c_1c_5c_6^2 - 180b_6c_1c_6^3) + 360c_2^2(a_{42}b_4(c_4 - c_5) + a_{62}b_6(-c_5 + c_6) - 3(c_5 - c_6)(-c_1(-1 + 1120a_{44}b_4c_4 + 120a_{55}b_5c_5 + 120a_{66}b_6c_6) + 2(-1 + 60a_{44}b_4c_4^2 + 60a_{55}b_5c_5^2 + 60a_{66}b_6c_6^2)))))$$

$$G = (-2 + 120a_{62}b_6c_2^2 + 60b_2c_2^4 + 120a_{42}b_4c_2(c_2 - c_3) + c_3 - 60b_1c_1^3c_3 - 120a_{62}b_6c_2c_3$$
$$-60b_2c_2^3c_3 - 120a_{44}b_4c_3c_4 + 120a_{44}b_4c_4^2 - 120a_{55}b_5c_3c_5 + 120a_{55}b_5c_3^2 - 120a_{66}b_6c_3c_6$$
$$+120a_{66}b_6c_6^2 - c_1(-1 + 60b_2c_2^3 + 120a_{42}b_4(c_2 - c_3) + 120a_{62}b_6(c_2 - c_3) - 120a_{62}b_6(c_2 - c_3))$$
$$-120a_{44}b_4c_3 - 120a_{55}b_5c_3 - 120a_{66}b_6c_3 + 60b_3c_3 + 120a_{44}b_4c_4 + 60b_4c_3c_4^2 + 120a_{55}b_5c_5 +$$
$$60b_5c_3c_5^2 + 120a_{66}b_6c_6 + 60b_6c_3c_6^2))$$
$$H = -180b_2c_2^5 + 180b_3c_3^5 - 180b_3c_1c_3^4c_4 + 180b_3c_1c_3^3(c_4 - c_5) + 180b_2c_2^4(c_3 + c_5) + 360a_{53}b_5c_3^2(-c_4 + c_5)$$
$$+180b_2c_2^3(c_1c_4 - (c_1 + c_3)c_5) - 360c_2^2(a_{42}b_4(c_4 - c_5) + a_{62}b_6(-c_5 + c_6)) - 3(c_4 - c_5)(-2 - c_1(-1 +$$
$$120a_{44}b_4c_4 + 120a_{55}b_5c_5 + 120a_{66}b_6c_6) + 120(a_{44}b_4c_4^2 + a_{55}b_5c_5^2 + a_{66}b_6c_6^2)) + c_2(2 + 3(-1 + 120(a_{42}b_4$$
$$-(a_{53} + a_{55})b_5 - a_{66}b_6)c_1)c_4 + 180b_1c_1^3(-c_1 + c_4) + 180(2a_{53}b_5c_3c_4 + b_3c_3^3(-c_3 + c_4) + 22a_{53}b_5c_1c_5 +$$
$$2a_{55}b_5c_1c_5 - 2a_{62}b_6c_1c_5 - 2a_{53}b_5c_3c_5 - 2a_{62}b_6c_3c_5 + 2a_{55}b_5c_4c_5 - 2a_{55}b_5c_5^2 + b_5c_1c_4c_5^2 - b_5c_1c_5^3 +$$
$$2a_{42}b_4(c_3c_4 - (c_1 + c_3)c_5) + 2b_6(a_{62}(c_1 + c_3) + a_{66}(c_1 + c_4))c_6 + b_6(-2a_{66} + c_1c_4)c_6^2 - b_6c_1c_6^3))) +$$
$$c_3(-2 + 180b_1c_1^3(c_1 - c_5) + 3c_5 + 360a_{42}b_4c_1(-c_4 + c_5) + 180(-2a_{44}b_4(c_1 - c_4)(c_4 - c_5) + c_1(2a_{53}b_5(c_4 - c_5)$$
$$+ b_4c_4^2(c_4 - c_5) + 2(a_{62} + a_{66})b_6c_5) - 2b_6(a_{62}c_1 + a_{66}(c_1 + c_5))c_6 + b_6(2a_{66} - c_1c_5)c_6^2 + b_6c_1c_6^3)))$$

Use assumption (3.9) to get

b_1', b_2', b_3', b_4', b_5' and b_6'

Minimization of Local Truncation Error

Using the explicit expressions of the above solution, the 7th order (or principal) local truncation error $\left\|\tau^{(7)}\right\|$ and $\left\|\tau'^{(7)}\right\|$ can be found explicitly. This is done by substituting the solutions into the principal truncation error coefficients of the 7th order equations given by

$$\left\|\tau^{(7)}\right\| = \sqrt{\sum_{j=1}^{N_7} \tau_j^{(7)}}, \qquad \left\|\tau'^{(7)}\right\| = \sqrt{\sum_{j=1}^{N_7'} \tau_j'^{(7)}} \tag{18}$$

where

$$\tau_1^{(7)} = \tau_2^{(7)} = \tau_3^{(7)} = \sum_{i=1}^{6} b_ic_i^5 - \frac{1}{42},\ \tau_4^{(7)} = \sum_{i,j=1}^{6} b_ic_i^2a_{i,j}c_j - \frac{1}{252} = \tau_5^{(7)},\ \tau_6^{(7)} = \sum_{i,j=1}^{6} b_ic_j^2a_{i,j}c_i - \frac{1}{504}$$
$$\tau_7^{(7)} = \tau_8^{(7)} = \tau_9^{(7)} = \sum_{i,j=1}^{6} b_ic_j^3a_{i,j} - \frac{1}{840},\ \tau_{10}^{(7)} = \sum_{i,j,k=1}^{6} b_ia_{i,j}a_{jk}c_k - \frac{1}{504}$$

The expression is a little lengthy but it can be used easily with a minimization package in order to find an optimal value for $\left\|\tau^{(7)}\right\|$ and $\left\|\tau'^{(7)}\right\|$ and numerically minimize to choose values for the free parameters. The resulting 6-stage order 6 PDIRKN method is expressed in Butcher tableau below

Table 1: Butcher Tableau of the Coefficients for 6-Stage PDIRKN Method

$$
\begin{array}{c|cccccc}
\frac{9}{50} & \frac{81}{5000} & & & & & \\
\frac{13}{100} & 0 & \frac{169}{20000} & & & & \\
\frac{1}{2} & 0 & 0 & \frac{1}{8} & & & \\
\frac{17}{25} & -\frac{1422296617561}{7904360320000} & \frac{43}{250} & \frac{166035272001}{316174412800} & \frac{143}{500} & & \\
\frac{29}{50} & \frac{169351174719}{380850250000} & -\frac{4633102482}{13329758750} & -\frac{43}{500} & 0 & \frac{87}{200} & \\
\frac{3}{5} & \frac{93807851587}{632966400000} & -\frac{77}{200} & -\frac{6753974598}{63296640000} & 0 & 0 & \frac{3}{20} \\
\hline
b & -\frac{37081}{120000} & \frac{665728}{1373625} & \frac{258941}{319680} & \frac{333799}{742500} & -\frac{26201}{54000} & -\frac{9}{20} \\
b' & -\frac{1520321}{6000000} & \frac{4826528}{11446875} & \frac{258941}{639360} & \frac{667598}{4640625} & -\frac{183407}{900000} & -\frac{9}{50}
\end{array}
$$

The stability interval of the method is examined using (6) and obtain the amplification matrix given by

$$
R(z) = \begin{bmatrix} 1+\frac{1}{2}z+\frac{34}{125}z^2+\frac{41}{500}z^3+\frac{3}{200}z^4-\frac{1}{2500}z^5-\frac{11}{5000}z^6 & 1+\frac{21}{125}z+\frac{21}{250}z^2+\frac{9}{400}z^3+\frac{1}{500}z^4-\frac{1}{500}z^5-\frac{1}{50}z^6 \\ \frac{31}{100}z+\frac{129}{1250}z^2+\frac{1}{50}z^3+\frac{1}{500}z^4-\frac{7}{5000}z^5-\frac{1}{500}z^6 & 1+\frac{2}{25}z+\frac{11}{500}z^2+\frac{1}{250}z^3-\frac{3}{2000}z^4-\frac{1}{625}z^5-\frac{49}{50000}z^6 \end{bmatrix}
$$

A boundary locus plot of $R(z)$ gives the stability interval of approximately (-3.2, 0).The stability region of the 6-stage PDIRKN method is shown in figure (2), where the stability region lies inside the boundary.

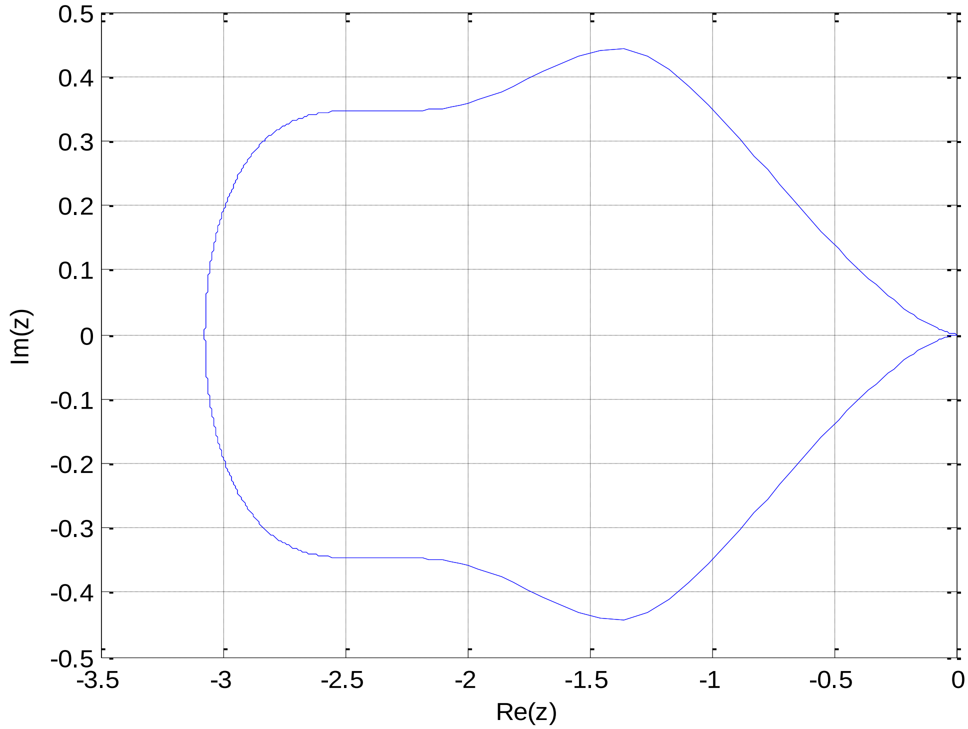


Figure 2: Stability Region for the 6-Stage PDIRKN Method

In order to implement the method in variable step size setting we would have to embed a lower order method, while using same function evaluations as the sixth order method. This means that the higher and lower order methods share the same matrix A and vector c but they have different weight vectors $\hat{b}$ and $\hat{b}'$

Numerical Examples

In this section, we present some problems which will be tested by the new method. The method was implemented sequentially since parallel computers were not readily available. We compare the new method derived in this paper with the analytical solution. The numerical results are given in Tables 2 and 3 and the nations used are as follows:

H-Step size, FCN - Number of function evaluations, STEP - Number of steps, EMAX- Max $\left\| y_n - y(x_n) \right\|$ that is, the absolute value of the computed solution minus the exact solution.

Example 1

$$y'' = -y, y(x_0) = 0, y'(x_0) = 1, 0 \le x \le 10$$

with analytical solution $y(x) = \sin(x)$

Example 2

$$y'' = -25y, y(0) = 0, y'(0) = 5$$

with analytical solution $y(x) = \sin(5x)$

The numerical results for the two examples above are depicted in Tables 2and 3. One measure of the accuracy of a method is to examine EMAX, the maximum error which is defined as the absolute value of the computed solution minus the analytical solution.

Table 2 Numerical Results for Example 1

H	FCN	STEP	EMAX
$\frac{1}{2}$	950	95	1.043822187
$\frac{1}{4}$	1890	189	8.787876×10^{-2}
$\frac{1}{8}$	3550	386	5.4388997×10^{-2}
$\frac{1}{10}$	4620	462	3.3768231×10^{-3}
$\frac{1}{16}$	7220	722	1.8982962×10^{-3}
$\frac{1}{20}$	9510	951	8.1211125×10^{-4}
$\frac{1}{250}$	93140	9314	5.1293866×10^{-4}

Table 3 Numerical Results for Example 2

H	FCN	STEP	EMAX
$\frac{1}{2}$	2310	231	0.119492675
$\frac{1}{4}$	5330	533	8.86332473×10^{-2}
$\frac{1}{10}$	12660	1266	2.35642176×10^{-3}
$\frac{1}{25}$	26230	2623	1.592566935×10^{-3}
$\frac{1}{50}$	136550	13655	7.481039222×10^{-4}
$\frac{1}{100}$	253420	25342	1.540506182×10^{-4}
$\frac{1}{200}$	1356720	135672	8.392525871×10^{-4}

DISCUSSION OF RESULTS AND CONCLUSION

Discussion of Results

The use of the usual test based on computing the maximum global error over the whole integration interval has been employed since it gives a more significant measure of efficiency. As it can be observed from Table 2 and Table 3, the numerical results compared favourably with the analytic solution. In terms of the global error, the higher the number of steps the smaller is the error produced.

CONCLUSION

In this paper, a class of a six-stage, 3-parallel, 3-processors sixth order diagonally implicit RKN method for the numerical of the special second order IVP (1) have been developed. The new method have the properties of minimized local truncation error coefficients as well as appropriate region of stability which is recommended for solving ODE problems possessing oscillatory solutions. Also, numerical tests were performed using sequential computer and from the results depicted in Tables 2 and 3, it is observed that the new method performed favourably with the analytic solution. In a future research, the method will be implemented on parallel codes with variable step size and to be compared with sequential code.

REFERENCE

Al-Khasawneh R.A., Ismail F. and Suleiman M. (2007). Embedded Diagonally Implicit Runge-Kutta-Nystrom 4(3) pair for Solving Special Second order

Amodio P.and Brugnano L. (1997), Parallel ODE Solvers Based on Block BVMs. Adv. Comput. Math. 7, 5-26

Amodio P.and Brugnano L. (2008), Parallel Solution in Time ODEs: Some Achievements and Perspectives. Applied Numerical Mathematics and Perspectives, Applied Numerical Mathematics, doi:10.1016/j.apnum.2008.03.024

Burrage K. (1997), Parallel Methods for ODEs. Advances in Computational Mathematics, 7, 1-3

Butcher J.C. (1964), On RK Processes of High Order. Jour. Austral. Math. Soc. iv (2), 179-194

Crisci M.R., Paternoster B. and Russo E. (1993), Fully Parallel RKN Methods for ODEs with Oscillating Solutions. Appl. Num. Math., 143-158

Franco J.M., and Gomez I. (2009), Accuracy and Linear Stability of RKN Methods for Solving Second-order Stiff Problems, Applied Numerical Mathematics, 59, 959-975

Fehlberg E. (1972), Classical Eight- and Lower-order RKN Formula with Stepsize Control for Special Second-order Differential Equations. NASA, Tech. Report R-381, Computing, 10, 305-31

Hairer E.,Norsett S.P. and Wanner G. (1993), Solving ODEs I: Nonstiff problems, Springer-Verlag, Berlin

Imoni S.O. and Ikhile M. N.O (2017), Zero Dissipative Parallel DIRKN Fourth Order Method for Second Order ODEs, the Journal of the Mathematical Association of Nigeria (ABACUS), Vol. 44, No.2, 233-243

Kanagarajam K. and Sambath M. (2010), RKN Method of Orders Three for Solving Fuzzy Differential Equations. Computational Methods in Applied Mathematics, Vol.10, No.2, 195-203

Sharp P.W and Fine J.M. and Burrage (1990), Two-stage and Three-stage DIRKN Methods of Orders Three and Four. IMA Journal of Numerical Analysis, 10, 489-504

Sommeijer B.P. (1993), Explicit, High Order RKN Methods for Parallel computers. Applied Numerical Mathematics, 13, 221-240 IVPs, Applied Mathematics and Computation, 190, 1803-1814

Sommeijer B.P. (1987), A Note on DIRKN Method. J. Comput. Appl. Math. 19, 395-399

Tsitouras Ch. (1998), High Order, Zero Dissipative RKN Methods. J. Comput. and Appl Math., 95, 157-161

Van de Houwen P.J. and Sommeijer B.P. (1989), DIRKN Methods for Oscillatory Problems. SIAM J. Numeri. Anal.Vol. 26, 414-429

Van de Vyver H. (2005), A RKN Pair for the Numerical Integration of Perturbed Oscillations. Computer Physics Communications, 167, 129-142