iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 10 **Issue:** IV **Month of publication:** April 2022

DOI: https://doi.org/10.22214/ijraset.2022.41779

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 10 Issue IV Apr 2022- Available at www.ijraset.com

The Direct Two-Point Block One-Step Method Which is efficient and Suitable for Solving Second-Order Differential Equation

Pramod Humane[1], Dipak Lanjewar[2], Dr. Rakesh Koushal
[1, 2]*Research Scholar, Chaudhary Charansingh University, Meerut (U.P)*
[3]*Guide*

Abstract: ***A direct two-point block one-step method for solving general second-order ordinary differential equations (ODEs) directly is presented in this paper. The one-step block method will solve the second-order ODEs without reducing to first-order equations. The direct solutions of the general second-order ODEs will be calculated at two points simultaneously using variable step size.Two point four step direct implicit block method is developed for solving directly the second order system of ordinary differential equations (ODEs) using variable step size. The method will estimate the solutions of Initial Value Problems (IVPs) at two points simultaneously by using four backward steps. The method developed is suitable for the numerical integration of non stiff and mildly stiff differential systems. Numerical results are given to compare the efficiency of the developed method to the existence block method.***

Keywords: ***Block Method, Two Point, First Order Ordinary Differential Equations , Higher Order Ordinary Differential Equations***

I. INTRODUCTION

The block method of Runge-Kutta type has been explored in [1], and it is suggested that a block of new approximation values is used simultaneously for solving first-order ODEs. The works in [3, 8, 9] have been considered in solving (1.1) using the block one-step method, while [3] has proposed a two-point implicit block one-step method for solving second-order ODEs directly and suggested that the method is suitable to be parallel.One-step block method such as the implicit Runge-Kutta method is also being referred to as one previous point to obtain the solution. The multistep block method in the form of Adams type formula is presented in [5, 6]. In [7], the block backward differentiation formula (BBDF) for solving stiff ODEs has been introduced and the solutions referred to as more than one previous point. The works in [6] showed the proposed two-point four-step block method presented as in a simple form of Adams Moulton method for solving second-order ODEs directly.

In this paper, we consider solving directly the second order non stiff and mildly stiff initial value problems (IVPs) for systems of ODEs of the form

$$y' = f(x, y, y'), \quad y(a) = y_0 \quad y'(a) = y_0', \quad x \in [a, b] \tag{1}$$

Eq. (1) arises from many physical phenomena in a wide variety of applications especially in engineering such as the motion of rocket or satellite, fluid dynamic, electric circuit and other area of application. The approach for solving the system of higher order ODEs directly has been suggested by several researchers such as Gear (1971), Suleiman (1979), Lambert (1993) and Omar (1999).

In the context of rough paths theory, numerical schemes are indispensable when simulating the solution to an RDE driven by a random rough path or when discretizing rough stochastic partial differential equations [BBR$^+$18]. In fact, numerical schemes played a fundamental role in rough paths theory from the very beginning. This is probably most visible in the work of Davie [Dav07], where the Milstein scheme is used to solve rough differential equations theoretically. This approach was generalized to higher order Taylor-type schemes by Friz and Victoir [FV10b]. However, in a stochastic context, these schemes are of little use in practice since they contain iterated stochastic integrals whose distribution is unknown in general. To overcome this difficulty, Deya, Neuenkirch and Tindel introduced so-called *simplified* schemes in [DNT12] in which the iterated stochastic integrals are replaced by products of increments of the driving process. These schemes were successfully used in different contexts, cf. e.g. [BFRS16, BBR$^+$18]. However, as Taylor methods, these numerical approximations suffer from the need to calculate or simulate derivatives of the vector

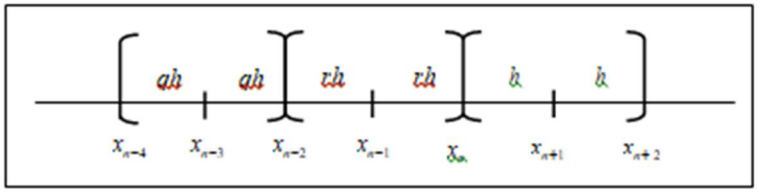
©IJRASET: All Rights are Reserved

fields f_k. As mentioned above, even if the derivatives are available, this can be very expensive especially in a large scale setting (e.g. spatially discretized rough partial differential equation). Moreover, the simplified scheme is difficult to implement in general. Therefore, we see the need of studying Runge-Kutta methods for rough differential equations that can easily be implemented and are derivative-free. We already pointed out that numerical schemes studied in the context of rough paths theory are mostly of Taylor-type. To our knowledge, the only exception is the article by Hong, Huang and Wang [HHW18] where a class of symplectic Runge-Kutta methods is considered to solve Hamiltonian equations driven by Gaussian processes. Our article differs from [HHW18] in several regards. On the technical level, no *B*-series are used in [HHW18], the authors have to prove all necessary estimates "by hand" in the framework of geometric rough paths. Consequently, they do not provide general order conditions. For instance, no explicit Runge-Kutta methods are deduced in [HHW18]. Moreover, their approach is probably hard to generalize to schemes of arbitrary order, whereas our approach does not have any limitations in this regard.

II. FORMULATION OF THE METHOD

In Figure 1, the two values of y_{n+1} and y_{n+2} are simultaneously computed in a block using the same five back values. The computing block has the step size h and the backward block has the step size rh and qh.

Figure 1: Two point two step block method



To approximate the first point y_{n+1} at x_{n+1} takes $x_{n+1} = x_n + h$ and integrate (1) once gives

$$\int_{x_n}^{x_{n+1}} y'(x)\,dx = \int_{x_n}^{x_{n+1}} f(x, y, y')\,dx \tag{2}$$

which is equivalent to

$$y'(x_{n+1}) = y'(x_n) + \int_{x_n}^{x_{n+1}} f(x, y, y')\,dx \tag{3}$$

The function $f(x, y, y')$ in (3) will be approximated using Lagrange interpolating polynomial and the interpolation points involved are $(x_{n-4}, f_{n-4}), \ldots, (x_{n+2}, f_{n+2})$. Taking $s = \frac{x - x_{n+2}}{h}$ and by replacing $dx = h\,ds$, the formulae of y_{n+1} can be obtained by integrating (3) over the interval $[x_n, x_{n+1}]$ using **MAPLE** and the corrector formulae in terms of r and q can be obtained.

During the implementation of the method, the choices of the next step size will be restricted to half, double or the same as the previous step size and the successful step size will remain constant for at least two blocks before being considered for the step size to be doubled. This will minimize the number of formulae required to be stored.

The step size ratio r and q vary for the code. In case of successful step size, the possible ratios for the next constant step size are $(r = 1, q = 1)$, $(r = 1, q = 2)$, or $(r = 1, q = 0.5)$. Whenever the step size is double, the possible ratio is $(r = 0.5, q = 0.5)$. In case of step size failure, the possible values are $(r = 2, q = 1)$ or $(r = 2, q = 2)$. Double step size failure is very rare where the ratios are $(r = 4, q = 2)$ or $(r = 4, q = 4)$. Substituting the common ratios r and q will give the formulae for the two point four step

implicit block method. For example, taking (r □ 0.5, q □ 0.5) will produce the following first point of the corrector formulae integrate once:-

A. First Integrating

$$y'_{n+1} = y'_n + \frac{h}{105840}[-705 f_{n+2} + 35686 f_{n+1} + 148512 f_n - 139328 f_{n-1} + 87402 f_{n-2} - 30144 f_{n-3} + 4417 f_{n-4}] \quad (4)$$

Now, integrating (1) twice gives

$$\int_{x_n}^{x_{n+1}} \int_{x_n}^{x} y'(x)\,dx\,dx = \int_{x_n}^{x_{n+1}} \int_{x_n}^{x} f(x, y, y')\,dx\,dx \quad (5)$$

Therefore,

$$y(x_{n+1}) - y(x_n) - h y'(x_n) = \int_{x_n}^{x_{n+1}} (x_{n+1} - x) f(x, y, y')\,dx \quad (6)$$

Replacing the $f(x, y, y')$ in (6) with the same interpolation polynomial through the points $(x_{n-4}, f_{n-4}), \ldots, (x_{n+2}, f_{n+2})$. Taking $s = \frac{x - x_{n+2}}{n}$ and by replacing $dx = h\,ds$ and integrate (6) over the interval $[x_n, x_{n+1}]$ using **MAPLE** and the corrector formulae in terms of r and q can be obtained.

The choices of r and q will be the same as mentioned above. For example, taking $(r = 0.5, q = 0.5)$ will produce the following first point of the corrector formulae integrate twice:-

B. Second Integrating

$$y_{n+1} = y_n + h y'_n + \frac{h^2}{211680}[-537 f_{n+2} + 18172 f_{n+1} + 160734 f_n - 127232 f_{n-1} + 77028 f_{n-2} - 26112 f_{n-3} + 3787 f_{n-4}] \quad (7)$$

The two point four step block method is the combination of predictor of order 6 and the corrector of order 7. The predictor formulae were similarly derived where the interpolation points involved are $(x_{n-5}, f_{n-5}), \ldots, (x_n, f_n)$.

Apply the same process above to find the integration coefficients of the second point, y_{n+2}, for the two point four step block method. Let $x_{n+2} = x_n + 2h$, integrating (1) once gives

$$\int_{x_n}^{x_{n+2}} y'(x)\,dx = \int_{x_n}^{x_{n+2}} f(x, y, y')\,dx \quad (8)$$

Therefore,

$$y'(x_{n+2}) = y'(x_n) + \int_{x_n}^{x_{n+2}} f(x, y, y') \quad (9)$$

Replacing the $f(x, y, y')$ in (9) with the same interpolation polynomial as in (3). Taking $s = \frac{x - x_{n+2}}{h}$ and by replacing $dx = h\,ds$ and integrate (9) over the interval $[x_n, x_{n+2}]$ using **MAPLE** and again the corrector formulae in terms of r and q can be obtained. For example, taking $(r = 0.5, q = 0.5)$, will produce the following second point of the corrector formulae integrate once:-

C. First Integrating

$$y'_{n+2} = y'_n + \frac{h}{13230}[3735 f_{n+2} + 22372 f_{n+1} - 21672 f_n + 47488 f_{n-1} - 38052 f_{n-2} + 14976 f_{n-3} - 2387 f_{n-4}] \quad (10)$$

Continue integrating (1) twice at the second point gives

$$\int_{x_n}^{x_{n+2}} \int_{x_n}^{x} y'(x)\, dx dx = \int_{x_{n-1}}^{x_{n+2}} \int_{x_n}^{x} f(x, y, y')\, dx dx \tag{11}$$

Therefore,

$$y(x_{n+2}) - y(x_n) - 2hy'(x_n) = \int_{x_n}^{x_{n+2}} (x_{n+2} - x) f(x, y, y')\, dx. \tag{12}$$

Replacing the $f(x, y, y')$ in (12) with the same interpolation polynomial through the points $(x_{n-4}, f_{n-4}), \ldots, (x_{n+2}, f_{n+2})$. Taking $s = \frac{x - x_{n+2}}{h}$ and by replacing $dx = h\, ds$ and integrate (12) over the interval $[x_n, x_{n+2}]$ using **MAPLE** and the second point of the corrector formulae can be obtained. For example, taking $(r = 0.5, q = 0.5)$, will produce the following second point of the corrector formulae integrate twice:-

D. Second Integrating

$$y_{n+2} = y_n + 2hy'_n + \frac{h^2}{13230}\left[783 f_{n+2} + 15064 f_{n-1} + 11466 f_n + 1792 f_{n-1} + 4536 f_{n-2} + 2304 f_{n-3} + 413 f_{n-4}\right] \tag{13}$$

III. IMPLEMENTATION

The code starts by finding the initial points in the starting block for the method. Initially we use the sequential direct second order Euler method to find the initial five starting points for the first block using constant *h*. The Euler method will be used only once at the beginning of the code. Once we find the initial points for the first starting blocks, then we could implement the block method until the end of the interval.

In the code, the values of y_{n+1} and y_{n+2} will be approximated using the predictor-corrector schemes. If s corrections are needed, then the sequence of computations at any mesh point is $(PE)(CE)^s$ where P and C indicate the application of the predictor and corrector formulae respectively and E indicate the evaluation of the function f. A simple iteration has been implemented to approximate the values of y_{n+1} and y_{n+2}. In the code, we iterate the corrector to convergence and the convergence test employed was

$$\left| y^{(s)}_{n+2} - y^{(s-1)}_{n+2} \right| < 0.1 \times \text{TOL} \tag{14}$$

where s is the number of iterations. After the successful convergence test, local errors estimate Est at the point x_{n+2} will be performed to control the error for the block. We obtained the Est by comparing the absolute difference of the corrector formula derived of order k and a similar corrector formula of order $k-1$. The error controls for the developed methods are at the second point in the block because generally it had given us better results.

The errors calculated in the code are defined as

$$(E)_t = \left| \frac{(y_i)_t - (y(x_i))_t}{A + B(y(x_i))_t} \right| \tag{15}$$

where $(y)_t$ is the t-th component of the approximate y. A=1, B=0 correspond to the absolute error test. A=1, B=1 correspond to the mixed test and finally A=0, B=1 correspond to the relative error test. The mixed error test is used for Problem 1, 2 and 3. The maximum error is defined as follows:-

$$\text{MAXE} = \max_{1 \le i \le SSTEP} \left(\max_{1 \le i \le N} (E_i)_i \right) \quad (16)$$

where N is the number of equations in the system and $SSTEP$ is the number of successful steps.

At each step of integration, a test for checking the end of the interval is made. If b denotes the end of the interval then

$$\text{if } x + 2h \ge b \text{ then } h = \frac{(b-x)}{2} \quad (17)$$

otherwise h remain as calculated. The technique above helped to reach the end point of the interval.

The step size when the integration steps are successful is given by

$$h_{new} = C \times h_{old} \left| \frac{TOL}{Est} \right|^{\frac{1}{r+1}}$$

$$\text{if } (h_{new} \ge 2 \times h_{old}) \text{ then } h_{next} = 2 \times h_{old} \quad (18)$$

$$\text{else } h_{next} = h_{old}$$

where $C = 0.5$ is a safety factor. The purpose of having the safety factor is to avoid having too many rejected step. The algorithm when the step failure occurs is

$$h_{next} = \frac{1}{2} \times h_{old} \quad (19)$$

The test in (18) and (19) will allow the step size to vary only by constant, halving or doubling. The predicted step size will perform only if the successful step size was remains constant for at least two blocks.

IV. RESULTS AND DISCUSSIONS

In order to study the efficiency of the developed code, we present some numerical experiments for the following three problems: The 2P4SDIR and 2PFDIR were applied to the following test problems:

A. *Problem 1*

$$y_1'' = -y_2' + \cos x,\ y_1(0) = -1,\ y_1'(0) = -1,$$
$$y_2'' = y_1 + \sin x,\ y_2(0) = 1,\ y_2'(0) = 0,\ x \in [0, 4\pi]$$

Solution: $y_1(x) = -\cos x - \sin x,$
$y_2(x) = \cos x.$

Source: Bronson (1973).

B. *Problem 2*

$$y_1'' = \frac{-y_1}{r},\ y_1(0) = 1,\ y_1'(0) = 0,$$
$$y_2'' = \frac{-y_2}{r},\ y_2(0) = 0,\ y_2'(0) = 1,$$
$$r = \sqrt{y_1^2 + y_2^2},\ x \in [0, 15\pi].$$

Solution: $y_1(x) = \cos x,$
$y_2(x) = \sin x.$

Source: Suleiman (1989).

C. Problem 3

$$y_1' = -y_2 + \sin \pi x,\ y_1(0) = 0,\ y_1'(0) = -1,$$
$$y_2'' = -y_1 + 1 - \pi^2 \sin \pi x,\ y_2(0) = 1,\ y_2'(0) = 1 + \pi,\ x \in [0,10]$$

Solution: $y_1(x) = 1 - e^x$,

$y_2(x) = e^x + \sin \pi x$.

Source: Bronson (1973).

The following notations are used in the tables:

TOL Tolerance
MTD Method employed
TS Total steps taken
FS Total failure step
FCN Total function calls
MAXE Magnitude of the maximum error of the computed solution
TIME The execution time taken in microseconds
RSTEP The ratio steps of 2P4SDIR compared to 2PFDIR
RTIME The ratio times of 2P4SDIR compared to 2PFDIR
2P4SDIR Implementation of the direct two point four step implicit block method of variable step size developed earlier
2PFDIR Implementation of the direct two point two step block method of variable step size in Majid (2007).

Source: Bronson (1973).

The following notations are used in the tables:

TOL Tolerance
MTD Method employed
TS Total steps taken
FS Total failure step
FCN Total function calls
MAXE Magnitude of the maximum error of the computed solution TIME The execution time taken in microseconds
RSTEP The ratio steps of 2P4SDIR compared to 2PFDIR RTIME The ratio times of 2P4SDIR compared to 2PFDIR
2P4SDIR Implementation of the direct two point four step implicit block method of variable step size developed earlier
2PFDIR Implementation of the direct two point two step block method of variable step size in Majid (2007).

The codes were written in C language and executed on DYNIX/ptx operating system. The numerical results for the three problems are presented in Table 1 – 3. The ratio of times and steps between 2P4SDIR and 2PFDIR are presented in Table 4.

Table 1: Comparison between 2P4SDIR and 2PFDIR for solving Problem 1

TOL	MTD	TS	FS	FCN	MAXE	TIME
10^{-2}	2PFDIR	33	0	180	1.60345e-2	564
	2P4SDIR	33	0	190	2.73003e-2	710
10^{-4}	2PFDIR	55	0	290	4.21048e-4	654
	2P4SDIR	42	0	256	1.72828e-3	837
10^{-6}	2PFDIR	74	0	452	9.05838e-5	940
	2P4SDIR	69	0	390	6.87609e-6	1182
10^{-8}	2PFDIR	130	0	860	3.32247e-6	1775
	2P4SDIR	84	0	556	9.64221e-7	1552
10^{-10}	2PFDIR	278	0	1600	3.75292e-8	3520
	2P4SDIR	160	0	926	2.04449e-9	2485

Table 2: Comparison between 2P4SDIR and 2PFDIR for solving Problem 2

TOL	MTD	TS	FS	FCN	MAXE	TIME
10^{-2}	2PFDIR	67	0	368	7.98175e-2	938
	2P4SDIR	67	0	382	1.03600e-2	1036
10^{-4}	2PFDIR	140	0	798	6.93117e-4	1472
	2P4SDIR	84	0	588	1.52571e-3	1426
10^{-6}	2PFDIR	316	0	1846	7.46033e-6	3318
	2P4SDIR	174	0	1016	4.49710e-6	2249
10^{-8}	2PFDIR	394	0	2306	2.45673e-6	4181
	2P4SDIR	216	0	1264	9.77410e-7	3051
10^{-10}	2PFDIR	938	0	5558	2.53897e-8	9932
	2P4SDIR	490	0	2900	1.68517e-9	6000

Table 3: Comparison between 2P4SDIR and 2PFDIR for solving Problem 3

TOL	MTD	TS	FS	FCN	MAXE	TIME
10^{-2}	2PFDIR	33	0	172	3.43536e-4	234
	2P4SDIR	32	0	178	2.08106e-4	456
10^{-4}	2PFDIR	55	0	290	1.08478e-5	713
	2P4SDIR	46	0	264	1.86939e-5	979
10^{-6}	2PFDIR	101	0	558	2.57639e-7	1399
	2P4SDIR	69	0	398	6.91125e-7	1259
10^{-8}	2PFDIR	152	0	856	2.28311e-8	2031
	2P4SDIR	89	0	528	6.74866e-8	1558
10^{-10}	2PFDIR	299	1	1730	5.68803e-10	4124
	2P4SDIR	163	0	946	2.85058e-10	2621

In Table 1 – 3, it is obvious that the total number of steps taken by 2P4SDIR is much lesser than 2PFDIR at all given tolerances. This is expected since 2P4SDIR is a variable step method with order higher than the 2PFDIR, and therefore the method will generally have larger step size and hence lesser total steps. The code 2P4SDIR is better in terms of execution times compared to 2PFDIR at smaller tolerances. However, at larger tolerances, the execution times of 2PFDIR are better than 2P4SDIR even though the total steps taken in 2PFDIR are slightly larger than the steps taken by 2P4SDIR in the tested problems. The extra terms in the 2P4SDIR has affected the computational times at larger tolerances. We also observed that in most cases the maximum error of 2P4SDIR is better or comparable to 2PFDIR at all tolerances.

Table 4: The ratios of execution times and steps for the 2P4SDIR method compared to 2PFDIR method for solving Problem 1 to 3

TOL	PROB 1		PROB 2		PROB 3	
	RTIME	RSTEP	RTIME	RSTEP	RTIME	RSTEP
10^{-2}	0.79	1.00	0.91	1.00	0.95	1.03
10^{-4}	0.78	1.31	1.03	1.67	0.73	1.20
10^{-6}	0.79	1.07	1.48	1.82	1.11	1.46
10^{-8}	1.14	1.55	1.37	1.82	1.30	1.70
10^{-10}	1.42	1.74	1.66	1.91	1.57	1.83

In Table 4, the ratios of steps (RSTEP) and the ratio of times (RTIME) are greater than one shows that the 2P4SDIR is efficient than 2PFDIR. In fact, the ratios are greater than 1.5 at smaller tolerances, which indicates a clear advantage of 2P4SDIR over 2PFDIR.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

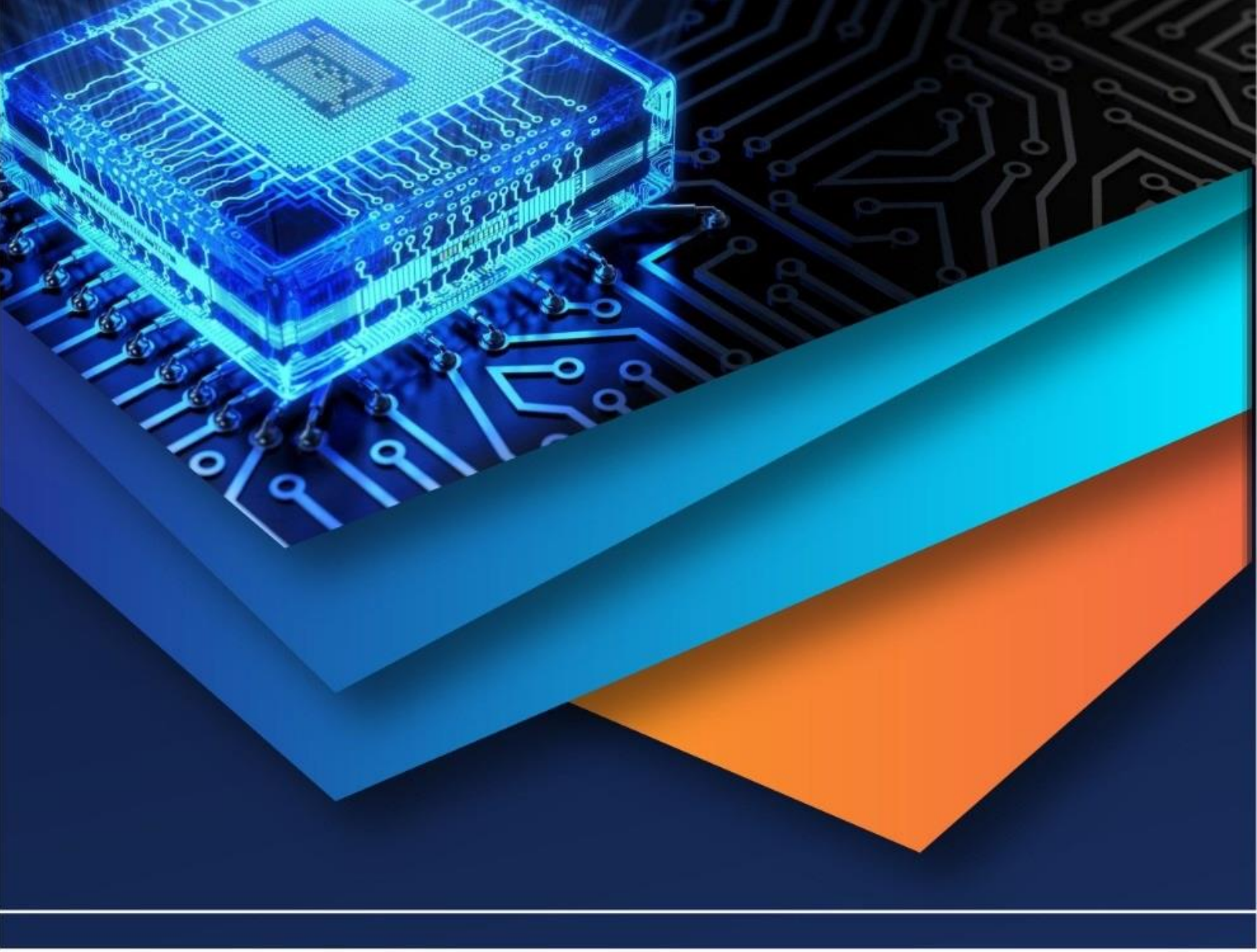
V. CONCLUSION

In this paper, we have shown the efficiency of the two point four steps direct integration implicit block method presented as in the form of Adams Moulton Method with variable step size is suitable for solving second order ODEs. The block method has shown better numerical results as the tolerances getting smaller. In this paper, we have constructed the direct two-point block one-step method which is efficient and suitable for solving general second-order ODEs directly. The block method has shown acceptable solutions and managed to solve the second-order ODE faster compared to the existing method.

REFERENCES

[1] Bronson. R, 1973. "Modern Introductory Differential Equation: Schaum's Outline Series". USA: McGraw-Hill Book Company.
[2] Gear, C.W. 1971. "Numerical Initial Value Problems in Ordinary Differential Equations", New Jersey: Prentice Hall, Inc.
[3] Lambert, J.D. 1993. "Numerical Methods For Ordinary Differential Systems. The Initial Value Problem". New York: John Wiley & Sons, Inc.
[4] Majid, Z.A. and Suleiman, M.B. 2007. "Two Point Block Direct Integration Implicit Variable Steps Method for Solving Higher Order Systems of Ordinary Differential Equations".
[5] Proceeding of the World Congress on Engineering 2007, WCE 2007, Volume II, pp 812-815
[6] Omar, Z. 1999. "Developing Parallel Block Methods For Solving Higher Order ODEs Directly", Ph.D. Thesis, University Putra Malaysia, Malaysia.
[7] Suleiman, M.B. 1979. "Generalised Multistep Adams and Backward Differentiation Methods for the Solution of Stiff and Non-Stiff Ordinary Differential Equations", Ph.D. Thesis, University of Manchester.
[8] Suleiman, M. B. 1989. "Solving Higher Order ODEs Directly by the Direct Integration Method," Applied Mathematics and Computation 33, pp 197-219.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)