MJFAS
Malaysian Journal of Fundamental and Applied Sciences

Abd Rasid *et al.* | Malaysian Journal of Fundamental and Applied Sciences, Vol. 20 (2024) 1083-1100

RESEARCH ARTICLE

Enhancing Accuracy and Efficiency in Stiff ODE Integration Using Variable Step Diagonal BBDF Approaches

Zarina Bibi Ibrahim[a], Hazizah Mohd Ijam[b], Saufianim Jana Aksah[c], Norshakila Abd Rasid[d*]

[a]Department of Mathematics and Statistics, Faculty of Science, Universiti Putra Malaysia, 43400 Serdang, Selangor, Malaysia; [b]School of Mathematical Science, College of Computing, Informatics and Mathematics, Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia; [c]Centre of Foundation Studies, Universiti Teknologi MARA, Cawangan Selangor, Kampus Dengkil, 43800 Dengkil, Selangor, Malaysia; [d]Mathematical Sciences Studies, College of Computing, Informatics and Mathematics, Universiti Teknologi MARA, Perak Branch, Tapah Campus, 35400 Tapah Road, Perak, Malaysia

Abstract Recent advancements in mathematical modelling have uncovered a growing number of systems exhibiting stiffness, a phenomenon that challenges the effectiveness of traditional numerical methods. Motivated by the need for more robust numerical techniques to address this issue, this paper presents an enhanced version of the Diagonally Block Backward Differentiation Formula (BBDF) that incorporates intermediate points, known as off-step points, to improve the accuracy and efficiency of solutions for stiff ordinary differential equations (ODEs). The new scheme leverages an adaptive step-size strategy to refine accuracy and efficiency between regular and off-grid integration steps. Theoretical analysis confirms that the proposed scheme is an A-stable and convergent method, as it satisfies the fundamental criteria of consistency, zero-stability, and A-stability. Numerical experiments on single and multivariable systems across varying time scales demonstrate significant improvements in solving stiff ODEs compared to existing techniques. Therefore, the new proposed method is an effective solver for stiff ODEs

Keywords: Variable step, diagonally, block backward differentiation formula, intermediate point.

***For correspondence:**
norshakila@uitm.edu.my

Received: 15 April 2024
Accepted: 14 Sept. 2024

Introduction

In real-world phenomena, especially in scientific and engineering disciplines, these phenomena are often characterized by single or systems of Ordinary Differential Equations (ODEs). ODE models, particularly systems of ODEs, may involve multiple variables, where the interconnection between variables can cause instability and exhibit significant behaviours called stiffness. In general, a system of ODEs is displayed by the following equation:

$$x' = \mathrm{f}\left(t, \mathrm{x}(t)\right), \quad x(a) = \mu \tag{1}$$

where $\mathrm{x} = [x_1, x_2, x_3, \dots, x_n]^T$, $\mathrm{f} = [f_1, f_2, f_3, \dots, f_n]^T$ and $\mu = [\mu_1, \mu_2, \mu_3, \dots, \mu_n]^T$ represent n-dimensional (column) vectors, t and a are scalars. Each element of x may indicate linear and nonlinear terms. The solution is computed over the interval $[a, b]$ with respect to t where a and b are finite. According to [1], the ODEs in (1) adhere to the Lipschitz conditions, thereby ensuring the existence and uniqueness of the solution.

Stiffness arises when the equations in (1) exhibit widely varying timescales, making it challenging to accurately approximate the solutions. According to the definition of stiffness in [1], in ODE systems with varying timescales, stiffness can be recognized through the eigenvalues of the Jacobian matrix

associated with system (1). Specifically, the ratio of the highest to the lowest eigenvalues quantifies the stiffness. This stiffness necessitates the use of small step sizes to maintain stability in rapidly changing components, which can lead to significant round-off errors and increased computational costs. To mitigate local instabilities during the integration step, implicit methods are preferred due to their optimal stability [1]-[2]. The implicit method described in [1] is represented by the following formula:

$$x_n = \varphi_1 x_{n-1} + \varphi_2 x_{n-2} + \cdots + \varphi_k x_{n-k} + \Delta t(\delta_0 f_n + \delta_1 f_{n-1} + \cdots + \delta_k f_{n-k} \quad (2)$$

where $n = k, k+1, \ldots$ and $x_i = x(t_i)$ and $f_i = f(t_i, x_i)$. The parameters φ_i and δ_i are manipulated to achieve the highest possible order subject to the following stability condition:

The method (2) is considered stable if the polynomial equation associated with it has no zeros outside the unit disk and no multiple zeros on the unit circle. The method is of order p if, for $y \in \mathbb{C}^{(p+2)}$

$$\rho(E)x(t) - h\sigma(E)x'(t) = D_{p+1}h^{p+1}x^{(p+1)}(t) + \mathcal{O}(h^{p+2}),$$

Where $Ex(t) = x(t+h)$ (refer to [2]).

However, due to the rigid theory of stability and convergence introduced by Dahlquist in [3], the order of the method in (2) cannot exceed $k+3$ to remain stable in solving (1). To achieve higher order while preserving the stability of the method, four independent researchers [4]-[7] proposed adding an extra term hf_θ into (2), where the derivative f is evaluated outside the grid point. The drawback of these methods was their complexity, as they required two separate predictors: one for approximating the solution at off-step points and another for approximating the solution $x(t_i)$. To address this, [8] combined the implicit continuous one-step method with four additional off-step points within the step, calling it a self-starting method since no predictor was needed for evaluation at off-step points. The meticulous selection of off-step points is crucial for optimizing local truncation error and maintaining local stability. It has become common practice among researchers to include off-step points in their formulas to enhance accuracy. For instance, [9] includes three off-step points x_{n+k} where $k = \frac{1}{3}, \frac{2}{3}$ and $\frac{1}{2}$. Similarly, [10] uses x_{n+k} where $k = \frac{1}{4}, \frac{1}{2}$, and $\frac{4}{3}$, and [11] selects x_{n+k} where $k = \frac{1}{5}, \frac{1}{2}$ and $\frac{4}{5}$. On the other hand, [12] opts for x_{n+k} where $k = \frac{1}{7}$ and $\frac{9}{7}$. Nevertheless, selecting off-grid points between two regular grid points often leads to zero-stable formulae (see [13]-[15]). This technique has become essential for developing advanced numerical integration solvers for stiff ODEs.

This trend of incorporating advanced numerical methods to handle stiffness in ODE systems has led to significant developments in the field. For instance, [16] introduced an implicit r-point Block Backward Differentiation Formula (BBDF) approach by modifying the classical Backward Differentiation Formula (BDF). While the classical BDF is typically known for solving nonlinear equations iteratively at each time step, the BBDF methods group multiple solution steps to solve stiff ODEs in a block scheme and reduce the computational cost. This novel methodology provides optimal stability for systems involving rapid changes in time scales and broadens the range of numerical techniques for handling stiff ODE problems. In a correlated study, [17] and [18] portrayed the absolute stable region and analysed the order and zero-stable of the method. It proved that the BBDF in [16] is A-stable and well-suited for addressing stiff problems. [19] highlighted that higher-order BBDF offers improved accuracy than the lower-order methods. [20] and [21] solved first-order fuzzy differential equations using new tailored BBDF methods. In reference [22] and [23], a fixed parameter ρ was incorporated into the modified BBDF, where the modification involved non-zero $\delta_{k-1,i}$ and the value of ρ was selected from the interval between -1 and 1.

For decades, the Diagonal BBDF introduced by [24] has emerged as one of the improved solvers for stiff ODE problems. The structure of the method blends the implementation of explicit methods with the stability of implicit methods, making it a compiling solver for stiff ODEs. The scheme builds upon the principles of the BBDF by focusing on the diagonal elements of the Jacobian matrix. This strategy strategically reduces computational complexity by exploiting the sparsity and structure of stiff ODE systems, resulting in faster and more efficient solution algorithms. [24] investigated the utilization of Diagonally Implicit two points BBDF of orders 2, 3, and 4 for stiff ODEs, emphasizing computational efficiency. Additionally, [25] explored the diagonalization of the fully implicit BBDF method presented in [23], demonstrating its superior stability, particularly when $\rho = -0.75$. [26] modified the Diagonally implicit BBDF by transforming the structure of the method into equal diagonal elements, giving only one evaluation of the Jacobian and one LU decomposition for each time step and eventually speeding up the algorithm. Moreover, [27] conducted a comprehensive study by providing the stability properties of the Diagonally Implicit BBDF (DIBBDF) scheme in the context of pharmacokinetics modelling. Meanwhile,

[28] offers insights into the Singly DIBBDF tailored to the unique characteristics of HIV infection dynamics and facilitates more accurate simulations for the disease progression.

As noted by [29], numerical integrators need to evolve into automatic codes that can adjust the step size dynamically throughout the solution interval, especially when dealing with rapid changes in the behavior of problems. The choice of step size is crucial when solving single or systems of ODEs with varying rates of change or sudden transitions. Implementing a constant step size strategy throughout the computation may lead to inefficiencies or inaccuracies in the findings. Thus employing the varying step size may circumvent the problem. Research in [30] suggested that to preserve the stability of the variable coefficient k-step BDF method, specific conditions on the step size ratio, r, must be met. For $k = 2$, the value of r must lie within the interval $[0,2]$. For $k = 3,4$, and 5, the value of r must be in the range $\left[0, \frac{k}{k-1}\right]$. These constraints ensure that the spectral radius of the iteration matrix remains less than or equal to 1, which is crucial for the stability of the numerical method. In the context of BBDF-tailored method, when a step is rejected, the step size is reduced by a factor of 0.5 $(r = 2)$. Conversely, when a step is successful, several studies have employed r values close to the upper bound of the stability interval, as highlighted in [30]. For instance, [31]-[33] used $r = \frac{5}{8}$ to increase the step size by a factor of 1.6 when the error is small. Similarly, [34] utilized $r = \frac{10}{19}$ to increase the step size by a factor of 1.9, while [35] opted for a factor of 1.196 $\left(r = \frac{1000}{1196}\right)$ to adjust the step size.

In response to the demand highlighted in [29], we employed the adaptive step technique in the framework design outlined in [15]. The step ratio was chosen based on the approach in [31], while the same off-grid point as used in [13] was maintained to ensure the method's stability. By dynamically adjusting the step sizes, the proposed scheme provides more accurate solutions between regular and intermediate integration steps, ultimately improving accuracy and efficiency, particularly in solving single and multivariable systems with components exhibiting varying time scales.

This paper is structured in the following manners. The first section under the title Materials and Methods presented the formulation of the new method, addressing its consistency and zero stability. In addition, We plotted the stability region to demonstrate the A-stable condition. The section includes implementing the running code using variable step strategies and a two-stage Newton iteration. Next, the Results and Discussion section presented the numerical results of the proposed scheme conducted on single and multivariable stiff first ODE problems and analysed them by comparing them with the existing works. Lastly, we provided a concise summary of the findings under the Conclusions section.

Materials and Methods

Formulation of VDBBDFO

In this section, we present the derivation of our newly proposed method, called the Variable step DIBBDF with Off-Step Points (VDBBDFO). This method extends the existing principles established by [15], which introduce a fixed-step A-stable diagonal BBDF formula with off-step points. The formulation of our new scheme begins with the following mathematical expression:

$$\sum_{p=0}^{z} \varphi_{p-z,q} x_{n+p-2} = \Delta t \delta_q f_{n+q} - \sum_{p=0}^{k-1} \varphi_{p+1,q} x_{n+\left(\frac{p+1}{2}\right)} \tag{3}$$

where x_{n+p-2} are the values of x at previous time step, t_{n+p-2}, and Δt is the step size defined as $\Delta t = t_n - t_{n-1}$. The coefficients $\varphi_{p-z,q}, \varphi_{p+1,q}$ and δ_q are determined based on the order of the method and the sequence of the step size. Here, z symbolizes the number of back values, $q = \frac{1}{2}, 1, \frac{3}{2}, 2$, representing the points in the formula, and k is the integer used to structure the diagonal form of VDBBDFO. The VDBBDFO algorithm revolves around approximating the solution for problem defined in (1) using a sequence of blocks within the partition of interval $[a, b]$, where $a = t_0 < t_1 < \cdots < t_{K-1} < t_K = b$. Our method estimates multiple solution points, including intermediate point, x_{n+1}, x_{n+2}, $x_{n+\frac{1}{2}}$ and $x_{n+\frac{3}{2}}$, using constrained ratio within a block series.

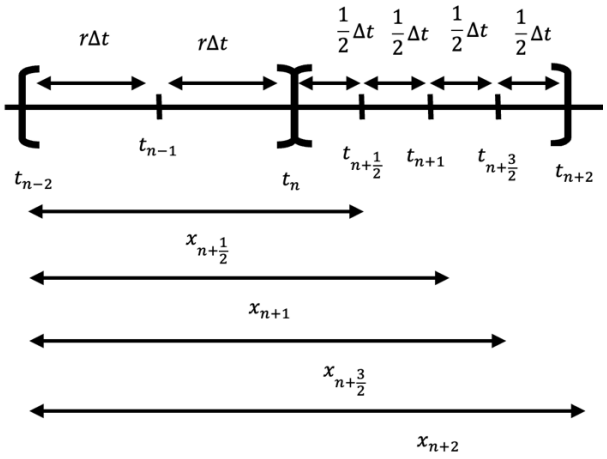

Figure 1. Variable step Diagonally Implicit Block Backward Differentiation Formula (VDBBDFO)

The VDBBDFO algorithm guarantees that the intervals between consecutive blocks are non-equidistant and adhere to a predefined ratio. Based on Figure 1, The current step size is $2\Delta t$, while in the previous block the step size is $2r\Delta t$, with r representing the step size ratio. This adaptive approach enables dynamic adjustment of step sizes based on solution behaviour, allowing larger steps in slowly changing regions for increased efficiency. The step size ratio r influences the relationship between consecutive block step sizes, affecting how quickly the method adapts to solution changes. Error estimates guide adjustments to the step size ratio during integration, refining accuracy. Dynamic step size adjustments maintain stability and accuracy, balancing computational efficiency with solution dynamics. This innovative approach integrates the following linear difference operator and express as:

$$L_q[x(t_n),\Delta t]=\sum_{p=0}^{z}\varphi_{p-2,q}x_{n+p-2}=\Delta t\delta_q f_{n+q}-\sum_{p=0}^{k-1}\varphi_{p+1,q}x_{n+\left(\frac{p+1}{2}\right)} \tag{4}$$

The preliminary step to construct the diagonal structure methods is to arrange the solution points, x_{n+q} where $q=\frac{1}{2},1,\frac{3}{2},2$ separately. Let $z=2,k=1,q=\frac{1}{2}$, we obtain the first row of the formula

$$\begin{aligned}L_{\frac{1}{2}}[x(t_n),\Delta t]=\varphi_{-2,\frac{1}{2}}x(t_n-2r\Delta t)+\varphi_{-1,\frac{1}{2}}x(t_n-r\Delta t)+\varphi_{0,\frac{1}{2}}x(t_n)+\varphi_{1,\frac{1}{2}}x\left(t_n+\frac{1}{2}\Delta t\right)\\-\Delta t\delta_{\frac{1}{2}}x'\left(t_n+\frac{1}{2}\Delta t\right)\end{aligned} \tag{5}$$

The second row of formula when $k=2,q=1$,

$$\begin{aligned}L_1[x(t_n),\Delta t]=\varphi_{-2,1}x(t_n-2r\Delta t)+\varphi_{-1,1}x(t_n-r\Delta t)+\varphi_{0,1}x(t_n)+\varphi_{1,1}x\left(t_n+\frac{1}{2}\Delta t\right)\\+\varphi_{2,1}x(t_n+\Delta t)-\Delta t\delta_1x'(t_n+\Delta t)\end{aligned} \tag{6}$$

The third row of the formula when $k=3,q=\frac{3}{2}$

$$\begin{aligned}L_{\frac{3}{2}}[x(t_n),\Delta t]=\varphi_{-2,\frac{3}{2}}x(t_n-2r\Delta t)+\varphi_{-1,\frac{3}{2}}x(t_n-r\Delta t)+\varphi_{0,\frac{3}{2}}x(t_n)+\varphi_{1,\frac{3}{2}}x\left(t_n+\frac{1}{2}\Delta t\right)\\+\varphi_{2,\frac{3}{2}}x(t_n+\Delta t)+\varphi_{3,\frac{3}{2}}x\left(t_n+\frac{3}{2}\Delta t\right)-\Delta t\delta_{\frac{3}{2}}x'\left(t_n+\frac{3}{2}\Delta t\right)\end{aligned} \tag{7}$$

The fourth row of the formula when $k=4,q=2$

$$\begin{aligned}L_2[x(t_n),\Delta t]=\varphi_{-2,2}x(t_n-2r\Delta t)+\varphi_{-1,2}x(t_n-r\Delta t)+\varphi_{0,2}x(t_n)+\varphi_{1,2}x\left(t_n+\frac{1}{2}\Delta t\right)\\+\varphi_{2,2}x(t_n+\Delta t)+\varphi_{3,2}x\left(t_n+\frac{3}{2}\Delta t\right)+\varphi_{4,2}x(t_n+2\Delta t)-\Delta t\delta_2x'(t_n+2\Delta t)\end{aligned} \tag{8}$$

Next, expanding the terms $x_{n+p-2}, x_{n+\left(\frac{p+1}{2}\right)}$ and $f_{n+q} = x'_{n+q}$ in the Taylor series around t_n.

$$\begin{aligned}
x(t_n - 2r\Delta t) &= x(t_n) - 2rx'(t_n) + \frac{4r^2}{2}x''(t_n) - \frac{8r^3}{6}x'''(t_n) + \cdots,\\
x(t_n - r\Delta t) &= x(t_n) - rx'(t_n) + \frac{r^2}{2}x''(t_n) - \frac{r^3}{6}x'''(t_n) + \cdots,\\
x(t_n) &= x(t_n),\\
x\left(t_n + \frac{1}{2}\Delta t\right) &= x(t_n) + \frac{1}{2}x'(t_n) + \frac{1}{8}x''(t_n) + \frac{1}{48}x'''(t_n) + \cdots,\\
x(t_n + \Delta t) &= x(t_n) + x'(t_n) + \frac{1}{2}x''(t_n) + \frac{1}{6}x'''(t_n) + \cdots,\\
x\left(t_n + \frac{3}{2}\Delta t\right) &= x(t_n) + \frac{3}{2}x'(t_n) + \frac{9}{8}x''(t_n) + \frac{27}{48}x'''(t_n) + \cdots,\\
x(t_n + 2\Delta t) &= x(t_n) + 2x'(t_n) + \frac{4}{2}x''(t_n) + \frac{8}{6}x'''(t_n) + \cdots,\\
x'\ \left(t_n + \tfrac{1}{2}\Delta t\right) &= x'^{(t_n)} + \tfrac{1}{2}x''(t_n) + \tfrac{1}{8}x'''^{(t_n)} + \tfrac{1}{48}x^{(4)}(t_n) + \cdots,\\
x'(t_n + \Delta t) &= x'(t_n) + x''(t_n) + \frac{1}{2}x'''(t_n) + \frac{1}{6}x^{(4)}(t_n) + \cdots,\\
x'\left(t_n + \frac{3}{2}\Delta t\right) &= x'(t_n) + \frac{3}{2}x''(t_n) + \frac{9}{8}x'''(t_n) + \frac{27}{48}x^{(4)}(t_n) + \cdots,\\
x'(t_n + 2\Delta t) &= x'(t_n) + 2x''(t_n) + \frac{4}{2}x'''(t_n) + \frac{8}{6}\,x^{(4)}(t_n) + \cdots,
\end{aligned} \tag{9}$$

We truncate the Taylor series expansion in (9) at the third derivative to secure the method to achieve a third-order accuracy. By substituting the equation (9) into (5) – (8) and rearranging terms, we get the

$$\mu_{0,q}x(t_n) + \mu_{1,q}\Delta t x'(t_n) + \mu_{2,q}(\Delta t)^2 x''^{(t_n)} + +\mu_{3,q}(\Delta t)^3 x'''^{(t_n)} + \cdots + \mu_{n,q}(\Delta t)^n x''x^n(t_n) \tag{10}$$

Where $q = \frac{1}{2}, 1, \frac{3}{2}, 2$

$$\begin{aligned}
\mu_{0,q} &= \sum_{p=0}^{2}\varphi_{p-2,q} + \sum_{p=0}^{k-1}\varphi_{p+1,q} = 0\,,\ \ k = 1\\
\mu_{1,q} &= \frac{(p-2)r}{1!}\sum_{p=0}^{2}\varphi_{p-2,q} + \frac{(q)r}{1!}\sum_{p=0}^{k-1}\varphi_{p+1,q} - \delta_q = 0\,,\ \ k = 2\\
\mu_{2,q} &= \frac{((p-2)r)^2}{2!}\sum_{p=0}^{2}\varphi_{p-2,q} + \frac{(q)^2}{2!}\sum_{p=0}^{k-1}\varphi_{p+1,q} - q\delta_q = 0\,,\ \ k = 3\\
\mu_{3,q} &= \frac{((p-2)r)^3}{3!}\sum_{p=0}^{2}\varphi_{p-2,q} + \frac{(q)^3}{3!}\sum_{p=0}^{k-1}\varphi_{p+1,q} - \frac{(q)^2}{2!}\delta_q = 0\,,\ \ k = 4
\end{aligned} \tag{11}$$

By letting $\mu_{0,q} = \mu_{1,q} = \mu_{2,q} = \mu_{3,q} = 0$ and $\varphi_{p-2,q} = 1$, where $p = 1, q = \frac{1}{2}, 1, \frac{3}{2}, 2$, and simultaneously solve the equation in (11) to produce set formula for VDBBDFO. The formula for each step ratio are organised in the following tables.

Table 1. The formula of the method when $r = 1$

q	δ_q	$\varphi_{0,q}$	$\varphi_{1,q}$	$\varphi_{2,q}$	$\varphi_{3,q}$	$\varphi_{4,q}$	$\varphi_{5,q}$	$\varphi_{6,q}$
$\frac{1}{2}$	$-\frac{15}{46}$	$-\frac{9}{184}$	$\frac{25}{92}$	$-\frac{225}{184}$	1	0	0	0
1	$-\frac{6}{23}$	$\frac{2}{115}$	$-\frac{3}{23}$	$\frac{18}{23}$	$-\frac{192}{115}$	1	0	0
$\frac{3}{2}$	$-\frac{105}{457}$	$-\frac{15}{1828}$	$\frac{147}{1828}$	$-\frac{1225}{457}$	$-\frac{735}{457}$	$-\frac{3675}{1828}$	1	0
2	$-\frac{4}{19}$	$\frac{3}{665}$	$-\frac{16}{285}$	$\frac{12}{19}$	$-\frac{512}{285}$	$\frac{48}{19}$	$-\frac{1536}{665}$	1

Table 2. The formula of the method when $r = 2$

q	δ_q	$\varphi_{0,q}$	$\varphi_{1,q}$	$\varphi_{2,q}$	$\varphi_{3,q}$	$\varphi_{4,q}$	$\varphi_{5,q}$	$\varphi_{6,q}$
$\frac{1}{2}$	$-\frac{25}{1888}$	$-\frac{9}{184}$	$\frac{25}{92}$	$-\frac{225}{184}$	1	0	0	0
1	$\frac{1}{424}$	$\frac{2}{115}$	$-\frac{3}{23}$	$\frac{18}{23}$	$-\frac{192}{115}$	1	0	0
$\frac{3}{2}$	$-\frac{49}{457}$	$-\frac{15}{1828}$	$\frac{147}{1828}$	$-\frac{1225}{457}$	$-\frac{735}{457}$	$-\frac{3675}{1828}$	1	0
2	$-\frac{4}{19}$	$\frac{3}{665}$	$-\frac{16}{285}$	$\frac{12}{19}$	$-\frac{512}{285}$	$\frac{48}{19}$	$-\frac{1536}{665}$	1

Table 3. The formula of the method when $r = \frac{5}{8}$

q	δ_q	$\varphi_{0,q}$	$\varphi_{1,q}$	$\varphi_{2,q}$	$\varphi_{3,q}$	$\varphi_{4,q}$	$\varphi_{5,q}$	$\varphi_{6,q}$
$\frac{1}{2}$	$-\frac{25}{1888}$	$-\frac{9}{184}$	$\frac{25}{92}$	$-\frac{225}{184}$	1	0	0	0
1	$\frac{1}{424}$	$\frac{2}{115}$	$-\frac{3}{23}$	$\frac{18}{23}$	$-\frac{192}{115}$	1	0	0
$\frac{3}{2}$	$-\frac{49}{457}$	$-\frac{15}{1828}$	$\frac{147}{1828}$	$-\frac{1225}{457}$	$-\frac{735}{457}$	$-\frac{3675}{1828}$	1	0
2	$-\frac{4}{19}$	$\frac{3}{665}$	$-\frac{16}{285}$	$\frac{12}{19}$	$-\frac{512}{285}$	$\frac{48}{19}$	$-\frac{1536}{665}$	1

The coefficient formulas listed in Tables 1, 2, and 3 have been stored in the codes. To solve the problem (1) using the formulas in (11), a distinct predictor formula is required. This is because computing the solutions x_{n+q} requires an approximate $x_{n+\left(\frac{p+1}{2}\right)}$ at the current step point $t_{n+\left(\frac{p+1}{2}\right)}$ and x_{n+p-2} from the previous step. As a result, we implemented the formula in (11) as a corrector scheme. For the predictor formula, the same derivation predictor method as outlined in [15] has been applied. The predictor formulas are expressed in equation (12) and the corresponding parameter values are tabulated in Table 4 to 6:

$$y_{n+q} = \sum_{p=0}^{2} \alpha_{p-2,q} x_{n+\left(\frac{p-2}{2}\right)} \tag{12}$$

Table 4. The predictor formula when $r = 1$

q	$\alpha_{-2,q}$	$\alpha_{-1,q}$	$\alpha_{0,q}$
$\frac{1}{2}$	1	-3	3
1	3	-8	6
$\frac{3}{2}$	6	-15	10
2	10	-24	15

Table 5. The predictor formula when $r = 2$

q	$\alpha_{-2,q}$	$\alpha_{-1,q}$	$\alpha_{0,q}$
$\frac{1}{2}$	$\frac{15}{8}$	$-\frac{21}{4}$	$\frac{35}{8}$
1	3	-8	6
$\frac{3}{2}$	$\frac{35}{8}$	-45	$\frac{63}{8}$
2	6	-15	10

Table 6. The predictor formula when $r = \frac{5}{8}$

q	$\alpha_{-2,q}$	$\alpha_{-1,q}$	$\alpha_{0,q}$
$\frac{1}{2}$	$\frac{7}{25}$	$-\frac{24}{25}$	$\frac{42}{25}$
1	3	-8	6
$\frac{3}{2}$	$\frac{207}{25}$	$-\frac{504}{25}$	$\frac{322}{25}$
2	$\frac{403}{25}$	$-\frac{936}{25}$	$\frac{558}{25}$

The next section explores the theoretical analysis of the method in (11). For several reasons, it is necessary to theoretically validate each one, encompassing things like the method's convergence, and A-stability analysis to guarantee dependable outcomes.

Convergence of VDBBDFO

Firstly, to better illustrate and analyse in a unified manner, the equation in (11) is rearranged as a general k-block, r-point method in a matrix finite difference equation form as follows:

$$\Psi X_p + \psi X_{P-1} + \xi X_{p-2} = \Delta t \phi X'_p. \tag{13}$$

Ψ, ψ, ξ and ϕ are all four by four matrices representing the coefficients of the equation. The functions X_p, X_{p-1}, X_{p-2} is the approximate solution at the p-*th* step. To organize numerical computations into blocks efficiently, the formula $n = pr$ is employed. This formula succinctly expresses that the step number n is equal to the product of the block number p and the block size r (number of points). This approach is referenced from [36]. Here $r = 2, n = 2p$

$$X_p = \begin{bmatrix} x_{2p+\frac{1}{2}} \\ x_{2p+1} \\ x_{2p+\frac{3}{2}} \\ x_{2p+2} \end{bmatrix} = \begin{bmatrix} x_{n+\frac{1}{2}} \\ x_{n+1} \\ x_{n+\frac{3}{2}} \\ x_{n+2} \end{bmatrix}, \quad X_{p-1} = \begin{bmatrix} x_{2(p-1)+\frac{1}{2}} \\ x_{2(p-1)+1} \\ x_{2(p-1)+\frac{3}{2}} \\ x_{2(p-1)+2} \end{bmatrix} = \begin{bmatrix} x_{2p-2+\frac{1}{2}} \\ x_{2p-2+1} \\ x_{2p-2+\frac{3}{2}} \\ x_{2p-2+2} \end{bmatrix} = \begin{bmatrix} x_{n-\frac{3}{2}} \\ x_{n-1} \\ x_{n-\frac{1}{2}} \\ x_n \end{bmatrix},$$

$$X_{p-2} = \begin{bmatrix} x_{2(p-2)+\frac{1}{2}} \\ x_{2(p-2)+1} \\ x_{2(p-2)+\frac{3}{2}} \\ x_{2(p-2)+2} \end{bmatrix} = \begin{bmatrix} x_{2p-4+\frac{1}{2}} \\ x_{2p-4+1} \\ x_{2p-4+\frac{3}{2}} \\ x_{2p-4+2} \end{bmatrix} = \begin{bmatrix} x_{n-\frac{7}{2}} \\ x_{n-3} \\ x_{n-\frac{5}{2}} \\ x_{n-2} \end{bmatrix} \quad X'_p = \begin{bmatrix} x'_{2p+\frac{1}{2}} \\ x'_{2p+1} \\ x'_{2p+\frac{3}{2}} \\ x'_{2p+2} \end{bmatrix} = \begin{bmatrix} x'_{n+\frac{1}{2}} \\ x'_{n+1} \\ x'_{n+\frac{3}{2}} \\ x'_{n+2} \end{bmatrix},$$

where,

$$\begin{bmatrix} \varphi_{3,\frac{1}{2}} & 0 & 0 & 0 \\ \varphi_{3,1} & \varphi_{4,1} & 0 & 0 \\ \varphi_{3,\frac{3}{2}} & \varphi_{4,\frac{3}{2}} & \varphi_{5,\frac{3}{2}} & 0 \\ \varphi_{3,2} & \varphi_{4,2} & \varphi_{5,2} & \varphi_{6,2} \end{bmatrix} \begin{bmatrix} x_{n+\frac{1}{2}} \\ x_{n+1} \\ x_{n+\frac{3}{2}} \\ x_{n+2} \end{bmatrix} + \begin{bmatrix} 0 & \varphi_{1,\frac{1}{2}} & 0 & \varphi_{2,\frac{1}{2}} \\ 0 & \varphi_{1,1} & 0 & \varphi_{2,1} \\ 0 & \varphi_{1,\frac{3}{2}} & 0 & \varphi_{2,\frac{3}{2}} \\ 0 & \varphi_{1,2} & 0 & \varphi_{2,2} \end{bmatrix} \begin{bmatrix} x_{n-\frac{3}{2}} \\ x_{n-1} \\ x_{n-\frac{1}{2}} \\ x_n \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & \varphi_{0,\frac{1}{2}} \\ 0 & 0 & 0 & \varphi_{0,1} \\ 0 & 0 & 0 & \varphi_{0,\frac{3}{2}} \\ 0 & 0 & 0 & \varphi_{0,2} \end{bmatrix} \begin{bmatrix} x_{n-\frac{7}{2}} \\ x_{n-3} \\ x_{n-\frac{5}{2}} \\ x_{n-2} \end{bmatrix}$$

$$= h \begin{bmatrix} \delta_{\frac{1}{2}} & 0 & 0 & 0 \\ 0 & \delta_1 & 0 & 0 \\ 0 & 0 & \delta_{\frac{3}{2}} & 0 \\ 0 & 0 & 0 & \delta_2 \end{bmatrix} \begin{bmatrix} f_{n+\frac{1}{2}} \\ f_{n+1} \\ f_{n+\frac{3}{2}} \\ f_{n+2} \end{bmatrix},$$

Convergence and stability analysis are fundamental to the theoretical study of numerical methods for solving differential equations. As referenced in [37] and [38], the following theorem and definitions illustrate this analysis.

Theorem 1 *The necessary and sufficient condition for a linear multistep method to be convergent are that it be consistent and zero stable.*

Definition 1 *The linear multistep method (2) is said to be consistent if it has order* $p \geq 1$.

As in the [2] the method (2) proved to be consistent if and only if

$$\sum_{j=0}^{k}\alpha_j = 0\,; \qquad \sum_{j=0}^{k} j\alpha_j = \sum_{j=0}^{k} 2\beta_j\,; \qquad k = 0,1,2,\ldots \tag{14}$$

Definition 2 *The linear multistep method (2) is said to be zero-stable if no root of the first characteristic polynomial $\rho(\xi)$ has modulus greater than one, and if every root with modulus one is simple.*

By truncating the Taylor series expansion in equation (9) at the third derivative, the method achieves third-order accuracy, establishing the VDBBDFO method as a consistent third-order approach. To ascertain consistency criteria as per (14), calculations are performed involving matrices Ψ, ψ, ξ and ϕ from Table 1, 2, and 3:

$$\sum_{j=0}^{k}\alpha_j = \begin{bmatrix}0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}\varphi_{0,\frac{1}{2}}\\ \varphi_{0,1}\\ \varphi_{0,\frac{3}{2}}\\ \varphi_{0,2}\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}\varphi_{1,\frac{1}{2}}\\ \varphi_{1,1}\\ \varphi_{1,\frac{3}{2}}\\ \varphi_{1,2}\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}\varphi_{2,\frac{1}{2}}\\ \varphi_{2,1}\\ \varphi_{2,\frac{3}{2}}\\ \varphi_{2,2}\end{bmatrix} + \begin{bmatrix}\varphi_{3,\frac{1}{2}}\\ \varphi_{3,1}\\ \varphi_{3,\frac{3}{2}}\\ \varphi_{3,2}\end{bmatrix} + \begin{bmatrix}0\\ \varphi_{4,1}\\ \varphi_{4,\frac{3}{2}}\\ \varphi_{4,2}\end{bmatrix} + \begin{bmatrix}0\\0\\ \varphi_{5,\frac{3}{2}}\\ \varphi_{5,2}\end{bmatrix} + \begin{bmatrix}0\\0\\0\\ \varphi_{6,2}\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\end{bmatrix}$$

$$\sum_{j=0}^{k} j\alpha_j - \sum_{j=0}^{k} 2\beta_j = 0$$

$$\sum_{j=0}^{k} j\alpha_j = 0\cdot\begin{bmatrix}0\\0\\0\\0\end{bmatrix} + 1\cdot\begin{bmatrix}0\\0\\0\\0\end{bmatrix} + 2\cdot\begin{bmatrix}0\\0\\0\\0\end{bmatrix} + 3\cdot\begin{bmatrix}\varphi_{0,\frac{1}{2}}\\ \varphi_{0,1}\\ \varphi_{0,\frac{3}{2}}\\ \varphi_{0,2}\end{bmatrix} + 4\cdot\begin{bmatrix}0\\0\\0\\0\end{bmatrix} + 5\cdot\begin{bmatrix}\varphi_{1,\frac{1}{2}}\\ \varphi_{1,1}\\ \varphi_{1,\frac{3}{2}}\\ \varphi_{1,2}\end{bmatrix} + 6\cdot\begin{bmatrix}0\\0\\0\\0\end{bmatrix} + 7\cdot\begin{bmatrix}\varphi_{2,\frac{1}{2}}\\ \varphi_{2,1}\\ \varphi_{2,\frac{3}{2}}\\ \varphi_{2,2}\end{bmatrix} + 8\cdot\begin{bmatrix}\varphi_{3,\frac{1}{2}}\\ \varphi_{3,1}\\ \varphi_{3,\frac{3}{2}}\\ \varphi_{3,2}\end{bmatrix}$$
$$+9\cdot\begin{bmatrix}0\\ \varphi_{4,1}\\ \varphi_{4,\frac{3}{2}}\\ \varphi_{4,2}\end{bmatrix} + 10\cdot\begin{bmatrix}0\\0\\ \varphi_{5,\frac{3}{2}}\\ \varphi_{5,2}\end{bmatrix} + 11\cdot\begin{bmatrix}0\\0\\0\\ \varphi_{6,2}\end{bmatrix} - 2\left(\begin{bmatrix}0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\0\end{bmatrix} + \cdots + \begin{bmatrix}\delta_{\frac{1}{2}}\\0\\0\\0\end{bmatrix} + \begin{bmatrix}0\\ \delta_1\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\ \delta_{\frac{3}{2}}\\0\end{bmatrix} + \begin{bmatrix}0\\0\\0\\ \delta_2\end{bmatrix}\right) = \begin{bmatrix}0\\0\\0\\0\end{bmatrix}$$

Therefore, the VDBBDFO method satisfies the consistency conditions outlined in (14).

Consistency alone does not guarantee convergence. The method is required to satisfy zero-stable conditions since it will ensure that the errors introduced by the numerical method remain bounded and do not grow unboundedly as the step size decreases toward zero. To derive the characteristic polynomial of the VDBBDFO method, equation (11) is associated with the scalar test equation $x' = \lambda x$. The characteristic polynomial of VDBBDFO in term of λ is then given by $\left(\Psi - \phi(\Delta t\lambda)\right)X_p - \psi X_{P-1} - \xi X_{p-2} = 0$, where the matrix coefficients are specified as

$$\Psi - \phi(\Delta t\lambda) = \begin{bmatrix} 1-\Delta t\lambda\varphi_{\frac{3,1}{2}} & 0 & 0 & 0 \\ \varphi_{3,1} & 1-\Delta t\lambda\varphi_{4,1} & 0 & 0 \\ \varphi_{3,\frac{3}{2}} & \varphi_{4,\frac{3}{2}} & 1-\Delta t\lambda\varphi_{5,\frac{3}{2}} & 0 \\ \varphi_{3,2} & \varphi_{4,2} & \varphi_{5,2} & 1-\Delta t\lambda\varphi_{6,2} \end{bmatrix},$$

$$\psi = \begin{bmatrix} 0 & \varphi_{1,\frac{1}{2}} & 0 & \varphi_{2,\frac{1}{2}} \\ 0 & \varphi_{1,1} & 0 & \varphi_{2,1} \\ 0 & \varphi_{1,\frac{3}{2}} & 0 & \varphi_{2,\frac{3}{2}} \\ 0 & \varphi_{1,2} & 0 & \varphi_{2,2} \end{bmatrix}, \qquad \xi = \begin{bmatrix} 0 & 0 & 0 & \varphi_{0,\frac{1}{2}} \\ 0 & 0 & 0 & \varphi_{0,1} \\ 0 & 0 & 0 & \varphi_{0,\frac{3}{2}} \\ 0 & 0 & 0 & \varphi_{0,2} \end{bmatrix}$$

Next, assuming $\overline{\Delta t} = \Delta t\lambda$, and computing the determinant: $\det\left((\Psi - \phi\overline{\Delta t}) - \psi - \xi\right)$. By letting $\overline{\Delta t} = 0$, we deduce the following roots:

$(r = 1)$

$$t_1 = \cdots = t_5 = 0, t_6 = -0.09251, t_7 = 0.00528, t_8 = 1. \tag{15}$$

$(r = 2)$

$$t_1 = \cdots = t_5 = 0, t_6 = -0.02404, t_7 = 0.00012, t_8 = 1 \quad (16)$$

$\left(r = \frac{5}{8}\right)$

$$t_1 = \cdots = t_5 = 0, t_6 = -0.21894, t_7 = 0.01965, t_8 = 1 \quad (17)$$

According to the roots in equations (15) to (17), the VDBBDFO method meets the root condition for zero stability and exhibits convergence.

Stability Analysis of VDBBDFO

Definition 3 *The linear multistep method (2) is said to be absolute-stable for a given* $\overline{\Delta t}$ *if, for that* $\overline{\Delta t}$*, all the roots no root of the first characteristic polynomial* $\rho(\xi)$ *has modulus greater than one, and if every root with modulus one is simple.*

Definition 4 A numerical method is said to be A-stable if its region of absolute stability contains the whole of the left-hand half plane $Re\Delta t\lambda < 0$.

In our study, we graphically plotted the stability region for the VDBBDFO method by substituting $t = e^{i\theta}$ where $0 < \theta < 2\pi$ into the stability polynomial, $\pi(t, \Delta t) = |(\Psi - \phi_2\overline{\Delta t})t^2 - \psi t - \xi|$. The range $0 < \theta < 2\pi$ indicates that θ varies from 0 to 2π, covering a full circle in the complex plane. This substitution allowed us to transform the equation into a suitable form for graphical representation on the complex plane. Using MAPLE 2015 software, we generated the graph of a complex plane of the VDBBDFO method with $r = 1, r = 2$ and $r = \frac{5}{8}$ in Figure 2, 3, and 4 respectively.

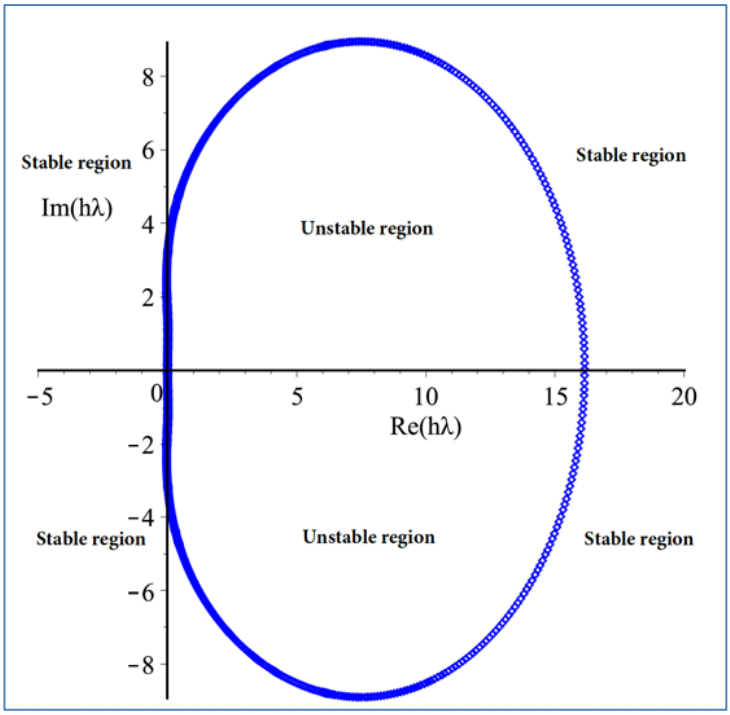

Figure 2. Stability region for VDBBDFO $(r = 1)$

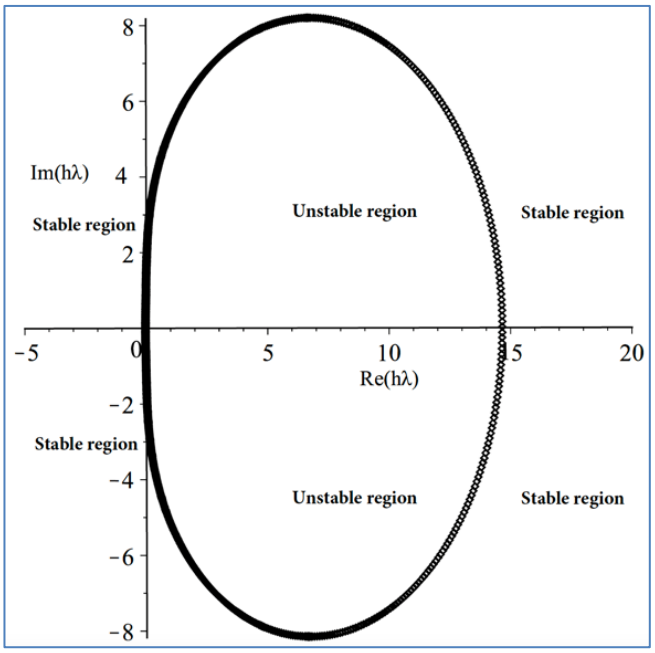

Figure 3. Stability region for VDBBDFO ($r = 2$)

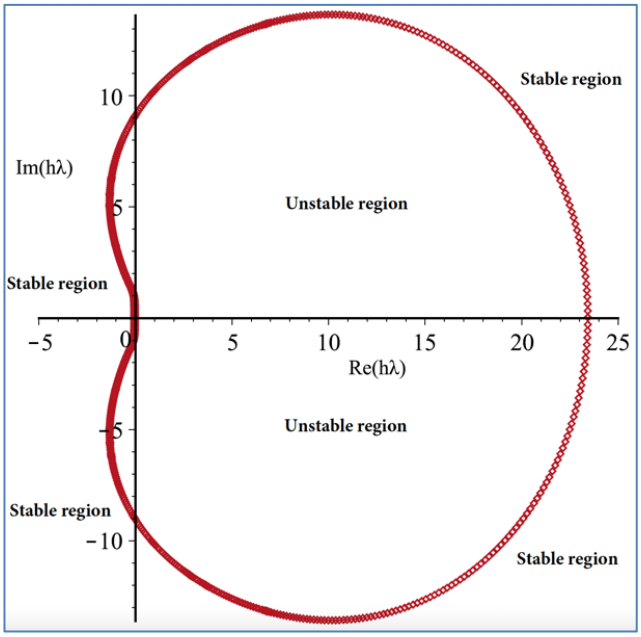

Figure 4. Stability region for VDBBDFO $\left(r = \frac{5}{8}\right)$

Referring to the graphs in Figures 2, 3 and 4, which depict the stability regions, the area outside the circle represents the stable region, while the area inside the circle is unstable. We observe that the entire left half-plane of the circle falls within the stable region for both $r = 1$ and $r = 2$ formulas. This observation confirms that the VDBBDFO method is A-stable for both cases. In contrast, the stability region for $r = \frac{5}{8}$ covers nearly the entire negative half-plane, indicating that it is almost A-stable. Therefore, this approach is well-suited for solving first-order stiff initial value problems, as it satisfies all the necessary theoretical criteria.

Variable Step Strategy

The variable step strategy involves adjusting the step size during the iterative process by employing adaptive control mechanisms that use error estimation techniques to determine when changes are necessary. Local truncation error (LTE) estimates error by comparing the solutions obtained from higher-order and lower-order methods. This concept, as discussed in [39], is mathematically expressed in equation (17)

$$\mathrm{LTE} = \left|x_{n+2}^{k+1} - x_{n+2}^{k}\right| < \mathrm{TOL} \tag{18}$$

where k is the order of the method and TOL is known as user-imposed tolerance. TOL in the formula sets the maximum allowable error considered acceptable for the numerical solution. When the local truncation error (LTE) falls below this tolerance TOL, it indicates solutions are converged in the iterative process. At this point, the next step size can stay constant $(r = 1)$ or be increased by a factor of 1.6 $\left(r = \frac{5}{8}\right)$, balancing efficiency while maintaining acceptable accuracy. If the local truncation error (LTE) surpasses the TOL, solutions are deemed unacceptable and rejected. Subsequently, the step size undergoes a reduction of half $(r = 2)$ and the solutions are recalculated based on the information from the preceding block. This process aims to enhance accuracy in the computation process.

Implementation of VDBBDFO

The code based on VDBBDFO employed a two-stage Newton iteration technique for solving stiff ODEs. This strategy allowed us to linearize the equation to iteratively refine initial approximations and effectively address the stiffness and nonlinearity in the ODE system. As a result of this approach, we achieved improved convergence and accuracy when dealing with complex stiff ODEs within practical computational timeframes. The formula for Newton's method for solving $F(t) = 0$,

$$x^{[k+1]} = x^{[k]} - J^{-1}\left(x^{[k]}\right)F\left(x^{[k]}\right) \tag{19}$$

Where $x^{[k]}$ is the current predict solution, $x^{[k+1]}$ is the next iteration and the Jacobian matrix, $J(x)$ is partial derivatives of $F\left(x^{[k]}\right)$ with respect to each component of x_{n+i}. To simplified the notation we let $\Delta x^{[k]} = x^{[k+1]} - x^{[k]}$, where $\Delta x^{[k]}$ is symbolized as correction term.

The method in (19) is applied to (11), where the formula in (11) can be generalised as follows:

$$\begin{aligned}
x_{n+\frac{1}{2}} &= \delta_{\frac{1}{2}}\Delta t f_{n+\frac{1}{2}} - A_{\frac{1}{2}} \\
x_{n+1} &= -\varphi_{3,1}x_{n+\frac{1}{2}} + \delta_1\Delta t f_{n+1} - A_1 \\
x_{n+\frac{3}{2}} &= -\varphi_{3,\frac{3}{2}}x_{n+\frac{1}{2}} - \varphi_{4,\frac{3}{2}}x_{n+1} + \delta_{\frac{3}{2}}\Delta t f_{n+\frac{3}{2}} - A_{\frac{3}{2}}(x_n) \\
x_{n+2} &= -\varphi_{3,2}x_{n+\frac{1}{2}} - \varphi_{4,2}x_{n+1} - \varphi_{5,2}x_{n+\frac{3}{2}} + \delta_2\Delta t f_{n+2} - A_2
\end{aligned} \tag{20}$$

where $A_{\frac{1}{2}}, A_1, A_{\frac{3}{2}}$ and A_2 are the back values. The Newton's method associated with equation in (20) is transform into matrix-vector form equivalent to

$$\left[(1-\Psi) - \Delta t\phi_i\frac{\partial f(t_{n+i}, x_{n+i}^{[k]})}{\partial x}\right]\Delta x^{[k]} = \left[x_{n+i} - \Delta t\phi_i f\left(t_{n+i}, x_{n+i}^{[k]}\right) - A_i\right]$$

with matrices are equal to

$$\left[(1-\Psi)-\Delta t\phi_i\frac{\partial f(t_{n+i},x_{n+i}^{[k]})}{\partial x}\right]$$

$$=\begin{bmatrix} 1-\delta_{\frac{1}{2}}\frac{\partial f_{n+\frac{1}{2}}^{[k]}}{\partial x_{n+\frac{1}{2}}^{[k]}} & 0 & 0 & 0 \\ -\varphi_{3,1}-\delta_1\frac{\partial f_{n+1}^{[k]}}{\partial x_{n+\frac{1}{2}}^{[k]}} & 1-\delta_1\frac{\partial f_{n+1}^{[k]}}{\partial x_{n+1}^{[k]}} & 0 & 0 \\ -\varphi_{3,\frac{3}{2}}-\delta_{\frac{3}{2}}\frac{\partial f_{n+\frac{3}{2}}^{[k]}}{\partial x_{n+\frac{1}{2}}^{[k]}} & -\varphi_{4,\frac{3}{2}}-\delta_{\frac{3}{2}}\frac{\partial f_{n+\frac{3}{2}}^{[k]}}{\partial x_{n+1}^{[k]}} & 1-\delta_{\frac{3}{2}}\frac{\partial f_{n+\frac{3}{2}}^{[k]}}{\partial x_{n+\frac{3}{2}}^{[k]}} & 0 \\ -\varphi_{3,2}-\delta_2\frac{\partial f_{n+2}^{[k]}}{\partial x_{n+\frac{1}{2}}^{[k]}} & -\varphi_{4,2}-\delta_2\frac{\partial f_{n+2}^{[k]}}{\partial x_{n+1}^{[k]}} & -\varphi_{5,2}-\delta_2\frac{\partial f_{n+2}^{[k]}}{\partial x_{n+\frac{3}{2}}^{[k]}} & 1-\delta_2\frac{\partial f_{n+2}^{[k]}}{\partial x_{n+2}^{[k]}} \end{bmatrix}$$

$$\left[x_{n+i}-\Delta t\phi_i f\left(t_{n+i},x_{n+i}^{[k]}\right)-A_i\right]=\begin{bmatrix} -x_{n+\frac{1}{2}}+\delta_{\frac{1}{2}}\Delta t f_{n+\frac{1}{2}}^{[k]}-A_{\frac{1}{2}} \\ -x_{n+1}-\varphi_{3,1}x_{n+\frac{1}{2}}+\delta_1\Delta t f_{n+1}^{[k]}-A_1 \\ -x_{n+\frac{3}{2}}-\varphi_{3,\frac{3}{2}}x_{n+\frac{1}{2}}-\varphi_{4,\frac{3}{2}}x_{n+1}+\delta_{\frac{3}{2}}\Delta t f_{n+\frac{3}{2}}^{[k]}-A_{\frac{3}{2}} \\ -x_{n+2}-\varphi_{3,2}x_{n+\frac{1}{2}}-\varphi_{4,2}x_{n+1}-\varphi_{5,2}x_{n+\frac{3}{2}}+\delta_2\Delta t f_{n+2}^{[k]}-A_2 \end{bmatrix}$$

The algorithm is implemented using two-stage Newton's method as follows:

- First stage: Calculate the correction term $\Delta x^{[k]}$ using the linear system $\Delta x^{[k]}=-J^{-1}(x^{[k]})F(x^{[k]})$.
- Second stage: Update the solution using $x^{[k+1]}=x^{[k]}+\Delta x^{[k]}$.

Repeat the Newton iteration process until the solution converges within the desired tolerance. Once convergence is achieved the code is advance to the next time step and process is repeated.

Results and Discussion

To test our proposed method, we considered three scientific models from the literature structured as equation (1) encountered in various scientific and engineering contexts. The problems included linear and non-linear single and systems of first-order ODEs with high stiffness ratio values that reflected the stiffness often encountered in real-world applications.

Problem 1:

$$x'=-300xy, \qquad x(0)=1, \quad 0\le t\le 20$$

Theoretical solution:

$$x(t)=e^{-150t^2}$$

Source: [40]

Problem 2:

$$x'_1=-998x_1+1998x_2, \quad x_1(0)=1, \quad 0\le t\le 20$$
$$x'_2=999x_1+1998x_2, \qquad x_2(0)=0$$

Theoretical solution:

$$x_1(t)=2e^{-t}-e^{-1000t}$$
$$x_2(t)=-e^{-t}+e^{-1000t}$$

The eigenvalues of the system are: $-1.-1000$

Stiffness ratio: 1000

Source: [41]

Problem 3:

$$x'_1 = 1195x_1 + 1995x_2, \quad x_1(0) = 2, \quad 0 \le t \le 20,$$
$$x'_2 = 1197x_1 + 1997x_2, \quad x_2(0) = -2,$$

Theoretical solution:

$$x_1(t) = 10e^{-2t} - 8e^{-800t},$$
$$x_2(t) = 6e^{-6t} - 8e^{-800t}.$$

The eigenvalues of the system are: $-2.-800$

Stiffness ratio: 400

Source: [42]

We validated the VDBBDFO method using the three test problems mentioned above. Our evaluation focused on metrics such as tolerance, maximum error, total number of steps, successful steps, failure steps, and execution time to comprehensively assess the method's performance. To investigate how the method adapts to varying discretization levels, we tested it with tolerances of $10^{-2}, 10^{-4}$ and 10^{-6}. We benchmarked VDBBDFO against an existing variable-step solver based on BBDF approaches and the MATLAB ODE solver to gauge its effectiveness in solving stiff ODEs and to provide valuable insights into its numerical capabilities. The notations used in the results table are defined as follows:

TOL	: Tolerances is a parameter used to control the accuracy of the numerical solution
MAXE	: The highest absolute difference between computed values x_i and theoretical values $x(t_i)$ across all data points or a specific interval. MAXE $max\|x_i - x(t_i)\|$).
TS	: Total number of steps
SS	: Successful steps
FS	: Failure steps
TIME	: CPU time measured in milliseconds
ode15s	: Variable-step, variable-order numerical differentiation formula ranging from order 1 to 5
VSBBDF	: Variable-step BBDF developed by [18]
VSBBDF-α	: Variable-step BBDF-α $(\alpha = -0.3)$ of order three developed by [43]
VDBBDFO	: An alternative solver introduced in this paper.

Table 7. Numerical performance for Problem 1

TOL	Method	TS	SS	FS	MAXE	TIME
10^{-2}	ode15s	46	35	11	1.2×10^{-2}	91676
	VSBBDF	44	37	7	1.7×10^{-2}	7471
	VDBBDFO	22	22	0	5.6×10^{-6}	1305
10^{-4}	ode15s	76	61	15	3.6×10^{-4}	60244
	VSBBDF(3)	67	57	10	2.8×10^{-4}	8687
	VDBBDFO	36	36	0	5.6×10^{-8}	3564
10^{-6}	ode15s	113	101	12	4.1×10^{-6}	77734
	VSBBDF(3)	97	87	10	1.4×10^{-6}	10122
	VDBBDFO	51	51	0	5.9×10^{-10}	5250

Table 8. Numerical performance for Problem 2

TOL	Method	TS	SS	FS	MAXE	TIME
10^{-2}	ode15s	37	37	0	1.3×10^{-3}	62400
	VSBBDF	26	26	0	2.3×10^{-4}	5549
	VS- BBDF-α	38	38	0	7.2×10^{-5}	3549
	VDBBDFO	31	31	0	1.0×10^{-5}	2847
10^{-4}	ode15s	89	89	0	2.6×10^{-5}	78000
	VSBBDF	53	53	0	2.4×10^{-6}	8429
	VS- BBDF-α	107	107	0	7.3×10^{-7}	5852
	VDBBDFO	46	46	0	1.0×10^{-7}	3336
10^{-6}	ode15s	164	161	3	8.9×10^{-7}	109200
	VSBBDF	130	130	0	2.3×10^{-6}	15481
	VS- BBDF-α	417	417	0	7.4×10^{-9}	6334
	VDBBDFO	61	61	0	1.0×10^{-9}	5390

Table 9. Numerical performance for Problem 3

TOL	Method	TS	SS	FS	MAXE	TIME
10^{-2}	ode15s	52	50	2	1.3×10^{-3}	46800
	VSBBDF	32	32	0	2.7×10^{-4}	6083
	VS- BBDF-α	47	47	0	1.2×10^{-4}	9210
	VDBBDFO	29	29	0	1.9×10^{-5}	1440
10^{-4}	ode15s	67	61	6	2.6×10^{-5}	62400
	VSBBDF	65	63	2	3.5×10^{-6}	9866
	VS- BBDF-α	147	147	0	1.0×10^{-6}	12137
	VDBBDFO	34	34	0	1.9×10^{-7}	2060
10^{-6}	ode15s	85	79	6	8.9×10^{-7}	78000
	VSBBDF	170	170	0	2.6×10^{-8}	19231
	VS- BBDF-α	609	609	0	9.2×10^{-9}	14020
	VDBBDFO	69	69	0	1.9×10^{-9}	3270

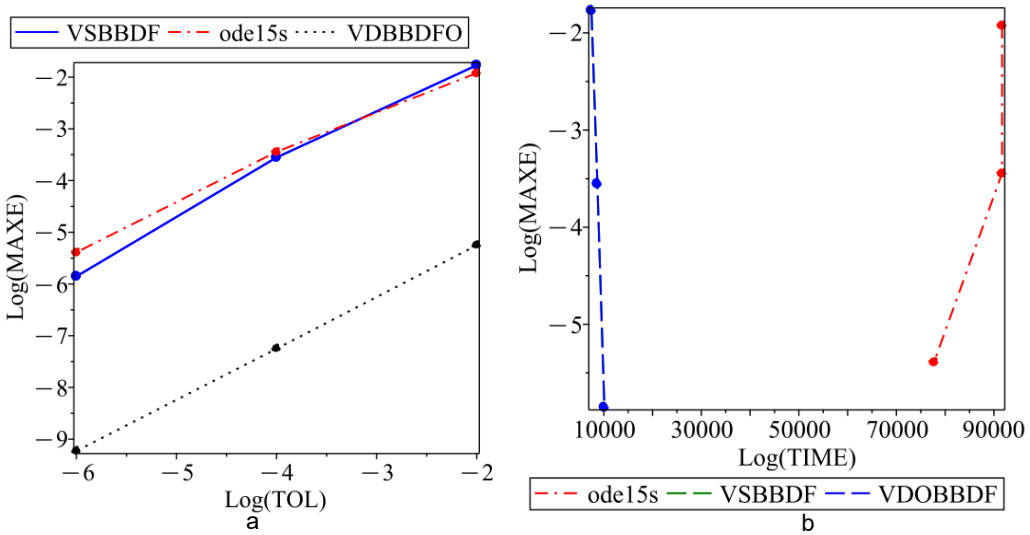

Figure 5. Efficiency curve for Problem 1 (a) Graph of Log(TOL) vs Log(MAXE) (b) Graph of Log(TIME) vs Log(MAXE)

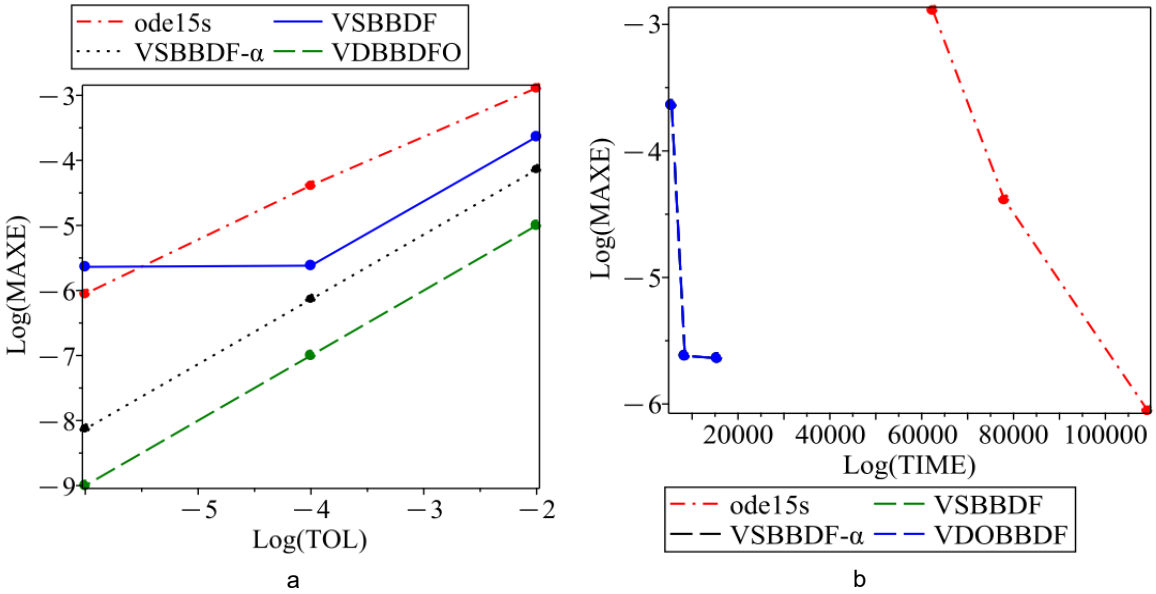

Figure 6. Efficiency curve for Problem 2 (a) Graph of Log(TOL) vs Log(MAXE) (b) Graph of Log(TIME) vs Log(MAXE)

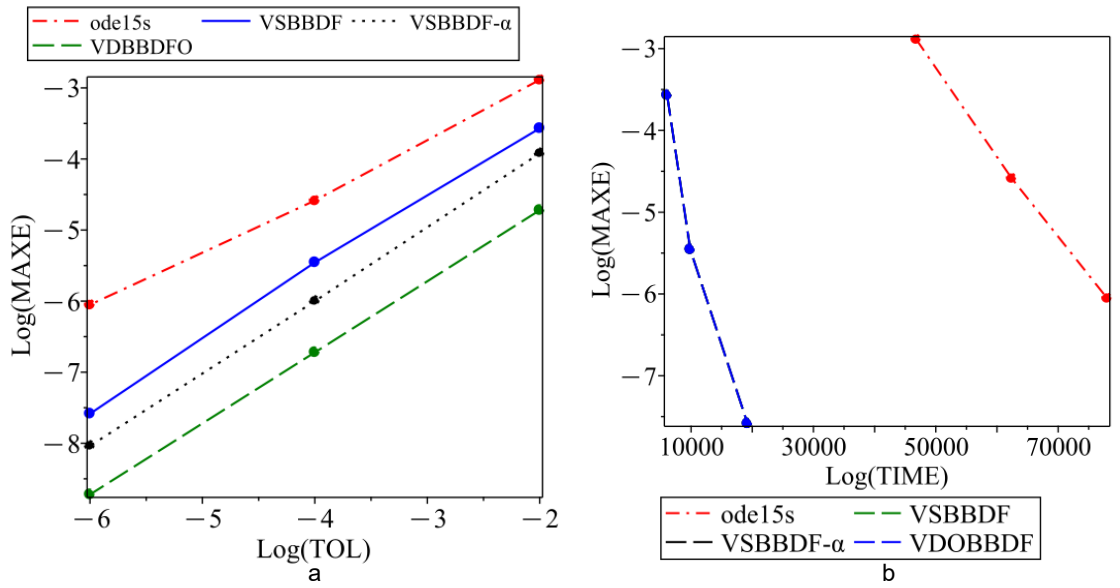

Figure 7. Efficiency curve for Problem 3 (a) Graph of Log(TOL) vs Log(MAXE) (b) Graph of Log(TIME) vs Log(MAXE)

We conducted a comparative analysis using the evaluation metrics of MAXE, TS, and TIME. The results for all solvers are detailed in Tables 7 to 9, with graphical representations provided in Figures 5 to 7 to elucidate the outcomes. The initial analysis of Tables 7 to 9 shows that ode15s, VSBBDF, VSBBDF$-\alpha$, and VDBBDFO consistently maintained errors within the specified tolerance (TOL) across all three problems. As expected, all solvers demonstrated improved accuracy as TOL decreased. This behavior aligns with the adaptive or variable-step nature of the methods, which have proven effective in solving stiff ODE problems.

VDBBDFO achieves a significant reduction in the number of steps, decreasing them by over 50% compared to both VSBBDF and ode15s when solving the non-linear Problem 1. For the highly stiff problems in Problems 2 and 3, VDBBDFO consistently requires fewer total steps than other methods (ode15s, VSBBDF, and VS-BBDF$-\alpha$), with reductions also exceeding 50%. However, in Problem 2 at $TOL = 10^{-2}$, VDBBDFO requires five more steps than VSBBDF, indicating that the method may occasionally need more steps depending on the specific tolerance and problem characteristics.

As encapsulated in Figures 5 to 7, the VDBBDFO line depicts a lower position on the graph, reflecting its lower MAXE values than other methods across all problems and tolerances. In contrast, Tables 8 and 9 show that VDBBDFO provides MAXE values comparable to VSBBDF-α in problems 2 and 3. However, VDBBDFO still outperforms VSBBDF-α in step reduction and gives better accuracy in Problem 1. The comparable MAXE values in Problem 2 and 3 can be attributed to the fact that both methods are well-suited for stiff ODEs and are optimized to handle the challenges presented by these problems. Additionally, the stringent tolerances used in the analysis contribute to the close matching of errors. However, it's important to note that VDBBDFO still offers significant advantages in terms of step reduction and overall computational efficiency.

On the other hand, VDBBDFO consistently demonstrates improve efficiency compared to other methods, showing significantly lower CPU time (TIME) across all problems and tolerances. For instance, in Problem 1 at $TOL = 10^{-2}$, VDBBDFO achieves a notable reduction in TS, which directly correlates with its reduced TIME compared to ode15s and VSBBDF. This efficiency is due to VDBBDFO's ability to handle problems with fewer steps, leading to lower overall computational time. In contrast, ode15s require more TIME due to their general-purpose nature, while VSBBDF shows improvements over ode15s but does not match the efficiency of VDBBDFO. VSBBDF$-\alpha$ also exhibits variable TIME results, often requiring more TIME than VDBBDFO despite competitive performance in some cases.

In conclusion, based on the comprehensive numerical comparison and efficiency assessments, the VDBBDFO emerges as a more efficient and accurate solver for stiff first-order ODEs.

Conclusions

In conclusion, this paper has highlighted significant advancements in the VDBBDFO method through its transition to a variable step scheme for solving stiff first-order ODEs. The method's capacity to simultaneously generate two solution values with two intermediate values within the block underscores its robustness and efficiency. Restoring the formulas with ratios equal to 2 and $5/8$ has reduced computational complexities and streamlined calculations of divided differences and integration coefficients. The step-changing scheme and diagonally implicit structure contribute further to computational cost savings. Our thorough evaluation has demonstrated that the VDBBDFO method surpasses other variable-step and variable-order solvers in accuracy and computational efficiency, as evidenced by consistent performance within specified tolerance levels and shorter execution times. The VDBBDFO method meets the theoretical criteria of convergence and A-stability condition, ensuring reliable and precise solutions across diverse numerical analysis scenarios. Future research directions could be optimizing the computational algorithms within VDBBDFO and evaluating their effectiveness in solving higher-order and more complex ODE systems. Potential enhancements may involve restructuring the diagonal form towards a Singly diagonally structured method or implementing parallel technique schemes. This initiative will surely improve VDBBDFO's adaptability and usability, reaffirming its value as a computational modelling and numerical analysis in the scientific and engineering fields.

Conflicts of Interest

The author(s) declare(s) that there is no conflict of interest regarding the publication of this paper.

Acknowledgement

This research was funded by MyRA Research Grant (MYRA-LPHD 2023) 600-RMC/GPM LPHD 5/3 (055/2023), Universiti Teknologi MARA (UiTM). The authors also gratefully acknowledge to School of Mathematical Sciences, College of Computing, Informatics and Mathematics, UiTM for supporting this research.

References

[1] Lambert, J. D. (1973). *Computational methods in ordinary differential equations*. John Wiley and Sons.
[2] Henrici, P. (1962). *Discrete variable methods in ordinary differential equations*. John Wiley & Sons, Inc.
[3] Dahlquist, G. (1956). Convergence and stability in the numerical integration of ordinary differential equations. *Mathematica Scandinavica*, 33–53.

[4] Gragg, W. B., & Stetter, H. J. (1964). Generalized multistep predictor-corrector methods. *Journal of the ACM (JACM), 11*(2), 188–209.

[5] Butcher, J. C. (1965). A modified multistep method for the numerical integration of ordinary differential equations. *Journal of the ACM (JACM), 12*(1), 124–135.

[6] Gear, C. W. (1965). Hybrid methods for initial value problems in ordinary differential equations. *Journal of the Society for Industrial and Applied Mathematics, Series B: Numerical Analysis, 2*(1), 69–86.

[7] Kohfeld, J. J., & Thompson, G. T. (1968). A modification of Nordsieck's method using an "off-step" point. *Journal of the ACM (JACM), 15*(3), 390–401.

[8] Akinfenwa, O. A., Jator, S. N., & Yao, N. M. (2011). A linear multi-step hybrid method with continuous coefficient for solving stiff ordinary differential equations. *Journal of Modern Mathematics and Statistics, 5*(2), 47–53.

[9] Kashkari, B. S., & Syam, M. I. (2019). Optimization of one-step block method with three hybrid points for solving first-order ordinary differential equations. *Results in Physics, 12*, 592–596.

[10] Ra'ft Abdelrahim, Z. O., & Kuboye, J. O. (2016). New hybrid block method with three off-step points for solving first-order ordinary differential equations. *American Journal of Applied Sciences, 13*, 209–212.

[11] Rufai, M. A., Carpentieri, B., & Ramos, H. (2023). A new hybrid block method for solving first-order differential system models in applied sciences and engineering. *Fractal and Fractional, 7*(10), 703.

[12] Emmanuel, S., Sathasivam, S., & Ogunniran, M. O. (2024). Multi-derivative hybrid block methods for singular initial value problems with application. *Scientific African, 24*, e02141.

[13] Abasi, N., Suleiman, M., Abbasi, N., & Musa, H. (2014). 2-point block BDF method with off-step points for solving stiff ODEs. *Journal of Soft Computing and Applications, 2014*, 1–15.

[14] Nasarudin, A. A., Ibrahim, Z. B., & Rosali, H. (2020). On the integration of stiff ODEs using block backward differentiation formulas of order six. *Symmetry, 12*(6), 952.

[15] Rasid, N. A., Ibrahim, Z. B., Majid, Z. A., & Ismail, F. (2021). Formulation of a new implicit for group implicit BBDF in solving related stiff ordinary differential equations. *Mathematics and Statistics, 9*(2), 144–150.

[16] Ibrahim, Z. B., Othman, K. I., & Suleiman, M. (2007). Implicit r-point block backward differentiation formula for solving first-order stiff ODEs. *Applied Mathematics and Computation, 186*(1), 558–565.

[17] Ibrahim, Z. B., Suleiman, M., Nasir, N., & Othman, K. I. (2011). Convergence of the 2-point block backward differentiation formulas. *Applied Mathematical Sciences, 70*(3), 3473–3480.

[18] Ibrahim, Z. B. (2006). *Block multistep methods for solving ordinary differential equations* (Doctoral dissertation). Universiti Putra Malaysia.

[19] Nasir, N. A. A. M., Ibrahim, Z. B., Suleiman, M., & Othman, K. I. (2011). Fifth order two-point backward differentiation formula for solving ordinary differential equations. *Applied Mathematical Sciences, 5*(71), 3505–3518.

[20] Ismail, N., Ibrahim, Z. B., & Othman, K. I. (2017). 2-point block backward differentiation formulas for solving fuzzy differential equations directly. *Chiang Mai Journal of Science, 44*(4), 1781–1791.

[21] Ismail, N., Ibrahim, Z. B., Othman, K. I., & Suleiman, M. (2014). 3-point block backward differentiation formulas for solving fuzzy differential equations. *Malaysian Journal of Mathematical Sciences, 8*, 139–151.

[22] Musa, H., Suleiman, M., Ismail, F., Senu, N., Majid, Z. A., & Ibrahim, Z. B. (2014). A new fifth order implicit block method for solving first order stiff ordinary differential equations. *Malaysian Journal of Mathematical Sciences, 8*, 45–59.

[23] Ibrahim, Z. B., Noor, N. M., & Othman, K. I. (2019). Fixed coefficient A(α) stable block backward differentiation formulas for stiff ordinary differential equations. *Symmetry, 11*(7), 846.

[24] Zawawi, I. S. M., Ibrahim, Z. B., & Othman, K. I. (2015). Derivation of diagonally implicit block backward differentiation formulas for solving stiff initial value problems. *Mathematical Problems in Engineering, 2015*, Article ID 179231.

[25] Ijam, H. M., & Ibrahim, Z. B. (2019). Diagonally implicit block backward differentiation formula with optimal stability properties for stiff ordinary differential equations. *Symmetry, 11*(11), 1342.

[26] Aksah, S. J., & Ibrahim, Z. B. (2019). Stability analysis of singly diagonally implicit block backward differentiation formulas for stiff ordinary differential equations. *Mathematics, 7*(2), 211.

[27] Ijam, H. M., Ibrahim, Z. B., Majid, Z. A., & Senu, N. (2020). Stability analysis of a diagonally implicit scheme of block backward differentiation formula for stiff pharmacokinetics models. *Advances in Difference Equations, 2020*, 1–22.

[28] Aksah, S. J., & Ibrahim, Z. B. (2019). Singly diagonally implicit block backward differentiation formulas for HIV infection of CD4+ T cells. *Symmetry, 11*(5), 625.

[29] Shampine, L. F. (1975). *Computer solution of ordinary differential equations: The initial value problem*.

[30] Calvo, M., Montijano, J. I., & Rández, L. (1993). A0-stability of variable stepsize BDF methods. *Journal of Computational and Applied Mathematics, 45*(1–2), 29–39.

[31] Ibrahim, Z. B., Othman, K. I., & Suleiman, M. (2007, July). Variable step block backward differentiation formula for solving first-order stiff ODEs. In *World Congress on Engineering* (pp. 785–789).

[32] Zawawi, I. S. M., Ibrahim, Z. B., & Othman, K. I. (2021, July). Variable step block backward differentiation formula with independent parameter for solving stiff ordinary differential equations. In *Journal of Physics: Conference Series* (Vol. 1988, No. 1, p. 012031). IOP Publishing.

[33] Ijam, H. M., Ibrahim, Z. B., & Zawawi, I. S. M. (2024). Stiffly stable diagonally implicit block backward differentiation formula with adaptive step size strategy for stiff ordinary differential equations. *Matematika*, 27–47.

[34] Yatim, S. A. M., Ibrahim, Z. B., Othman, K. I., & Suleiman, M. B. (2011). A quantitative comparison of numerical methods for solving stiff ordinary differential equations. *Mathematical Problems in Engineering, 2011*(1), 193691.

[35] Abasi, N., Suleiman, M., Ismail, F., & Musa, H. (2014). A new formula of variable step 3-point block BDF method for solving stiff ODEs. *Journal of Pure and Applied Mathematics: Advances and Applications, 12*(1),

49–76.
[36] Fatunla, S. (1991). Block methods for second-order ODEs. *International Journal of Computer Mathematics, 41*, 55–63.
[37] Dahlquist, G. (1956). Convergence and stability in the numerical integration of ordinary differential equations. *Mathematica Scandinavica, 4*, 33–53.
[38] Dahlquist, G. (1959). Stability and error bounds in the numerical integration of ordinary differential equations. *Kungl. Tekniska Högskolans Handlingar No. 130*.
[39] Butcher, J. C. (2016). *Numerical methods for ordinary differential equations*. John Wiley & Sons.
[40] Russell, R. D. (1977). A comparison of collocation and finite differences for two-point boundary value problems. *SIAM Journal on Numerical Analysis, 14*(1), 19–39.
[41] Gear, C. W. (1971). *Numerical initial value problems in ordinary differential equations*. Prentice Hall, Inc.
[42] Gerald, C. F., & Wheatley, P. O. (1989). *Applied numerical analysis* (4th ed.). Addison-Wesley Publishing Company.
[43] Zawawi, I. S. M. (2017). *Block backward differentiation alpha-formula for solving ordinary differential equations* (Doctoral dissertation). Universiti Putra Malaysia.