Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2014, Article ID 348912, 10 pages
http://dx.doi.org/10.1155/2014/348912

Research Article

Solving Delay Differential Equations of Small and Vanishing Lag Using Multistep Block Method

Nurul Huda Abdul Aziz,[1] Zanariah Abdul Majid,[1,2] and Fudziah Ismail[1,2]

[1] *Institute for Mathematical Research, Universiti Putra Malaysia (UPM), 43400 Serdang, Selangor, Malaysia*
[2] *Mathematics Department, Faculty of Science, Universiti Putra Malaysia (UPM), 43400 Serdang, Selangor, Malaysia*

Correspondence should be addressed to Zanariah Abdul Majid; zana_majid99@yahoo.com

Received 8 May 2014; Accepted 6 September 2014; Published 29 October 2014

Academic Editor: Ferenc Hartung

This paper considers the numerical solution of delay differential equations for solving the problem of small and vanishing lag using multistep block method. This problem arises when the size of a delay value is smaller than the step size, $x - \tau < h$, and the delay time may even vanish when $\tau \to 0$ in a current step. The proposed approach that is based on interpolation of Newton divided difference has been implemented by adapting this problem to the multistep block method. In order to achieve the required accuracy, this approach considered the appropriate degree of interpolation polynomial in approximating the solution of delay term. The developed code for solving small and vanishing lag is done using C program and we called it as DDEB5. The P-stability and Q-stability of this method are also studied. Numerical results are presented and compared to the existing method in order to illustrate the efficiency of the proposed method.

1. Introduction

Delay differential equation (DDE) is one of the mathematical models that commonly possess the result in differential equations with time delay. In general, the unknown function of this derivative equation not only depends on the current value but also depends on the past value which is called a delay term. This equation can be given in the form of

$$\begin{aligned} y'(x) &= f\left(x, y(x), y(x-\tau)\right), \quad x \in [a,b], \\ y(x) &= \phi(x), \quad x \in [-\tau, a], \end{aligned} \tag{1}$$

where τ is a time delay or lag which is a positive constant, $\phi(x)$ is a prescribed initial function, and $y(x-\tau)$ is the solution of delay term.

There are two families of difficulties that may exist in the numerical solution of DDEs: the occurrences of derivative discontinuity along the integration interval and the small and vanishing lag where the delay term is smaller than current step size and the delay time may even vanish as $\tau \to 0$ in $(x - \tau)$, respectively. In this paper, we will focus on the study of dealing with the second difficulty in the adaptation of multistep block method which is specifically 2-point modified block method.

In the previous work, there are several authors that have investigated the numerical solution of DDEs with the problem of vanishing delay and small delay. For instance, Enright and Hu [1] developed an approach which combines an iteration scheme and interpolation technique that is based on two time steps for Runge-Kutta methods to handle the vanishing delay. Their main idea is to use all the information from the last step including the early stages of the current step in the interpolation technique.

Karoui and Vaillancourt [2] developed a SYSDEL code which is based on the numerical method of Runge-Kutta for the formula pair of order (5, 6) for solving the case of vanishing lag and asymptotically vanishing lag of (1). In their study of solving vanishing lag case, they have used interpolation or extrapolation of 3-point Hermite polynomial to approximate the solution of the delay term by depending on the location

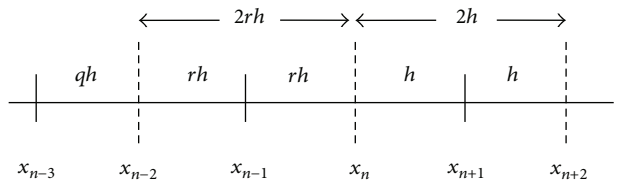

Figure 1: 2-point modified block method.

of delay time fall in the history queue or in a neighborhood of vanishing lag point, respectively. Then for the asymptotically vanishing lag case, they have used the approach of the first case as starter and then the delay equation is approximated by solving it as an ODE.

The latest code for the numerical scheme of small, vanishing, and asymptotically vanishing delay in DDEs has come out in Yagoub et al. [3] in the name of HBODDE. This code is based on a hybrid variable-step variable-order 3-stage Hermite-Birkhoff-Obrechkoff ODE solver that has been adapted in DDEs. The delay values are computed by Hermite interpolation and the small delay deal with extrapolation.

Hayashi [4] handled small and vanishing delay by proposing three algorithms of iterative scheme which are extrapolation, special interpolant, and iteration procedure with the adaptation of continuous Runge-Kutta method, while Neves and Thompson [5] handled small and vanishing delay by restricting the step size to be smaller and using extrapolation, respectively. All of the approaches that have been proposed are due to the one-step integration method where when the delay time falls in the current step, there is no any available approximant information that can be used for computing the delay term, $(x - \tau)$. This is lead to the use of extrapolation in their way to handle the small or vanishing lag.

In the past eight years, the study of DDE in the numerical solution of multistep block method has gained some attention among the researchers. These methods were initially investigated in solving ODE and have shown the advantages of less computational works and manage to obtain good accuracy. For the previous work please see [6–8]. In the study of DDE involving block method, Ishak et al. [9] has developed the two-point block for solving delay differential equations by using six points of Lagrange interpolation to evaluate the delay solution, while San et al. [10] has investigated a coupled block method that can integrate the solution within two different blocks which are namely two-point two-step block and three-point two-step block method. In both works, they only solved a typical retarded DDE problem without any difficulty that may arise in delay differential equations. These situations allow us to fill the gap taking into account the second difficulty with the adaptation of 2-point modified block method.

In this paper, we have developed a new DDEB5 code of C program for handling small and vanishing lag using the current approximate information provided from the approximation of PE(CE)^s mode to evaluate the small and vanishing lag using Newton divided difference interpolation. The detail of this numerical scheme is described in Section 3 in conjunction with the summary of DDEB5 code.

2. Formulation of the Method

The multistep block method based on k-step predictor-corrector pair can be defined as

$$\begin{aligned} \text{Predictor}: &\quad \sum_{q=0}^{k-1} \alpha_q^* y_{n+q} = h \sum_{q=0}^{k-1} \beta_{-q}^* f_{n-q}, \\ \text{Corrector}: &\quad \sum_{q=0}^{k} \alpha_q y_{n+q} = h \sum_{q=0}^{k} \beta_{2-q} f_{n+2-q}, \end{aligned} \tag{2}$$

where the step number from $q = 0$ to $(k - 1)$ and k refers to the order of the predictor and corrector, respectively.

There are many types of methods that can be generated from (2). Specifically, in this paper, we will only focus on the 2-point modified block method. According to Figure 1, this 2-point modified block method will approximate the solution of y_{n+1} and y_{n+2} at two points x_{n+1} and x_{n+2} simultaneously using the information in the previous block at points x_{n-2}, x_{n-1}, and x_n. This process will continue on the next block until it reaches the end of the interval. In particular, the computed block has step size $2h$, while the previous block has the step size $2rh$ where the use of r and q in this method is for variable step size implementation.

Basically, this 2-point modified block method is differing from the block method that has been proposed by Abdul Majid and Suleiman [6]. In this method, the approximation of these two solutions y_{n+1} and y_{n+2} will be integrated over the interval $[x_n, x_{n+1}]$ and $[x_{n+1}, x_{n+2}]$, respectively. While in Abdul Majid and Suleiman [6], the approximation of the solution is obtained by integrating over the interval $[x_n, x_{n+1}]$ and $[x_n, x_{n+2}]$, respectively. Our considered approach is called Gauss-Seidel method which is an iterative procedure that is based on a modification of the Jacobi method. In Gauss-Seidel method, the first approximation is computed in the same manner as in the Jacobi method. However, in computing the second approximation, the Gauss-Seidel method assumes that the new solution values are a better approximant to the solution than the initial values. In other words, to approximate the second point y_{n+2} in our algorithm, the most recently calculated approximation which is y_{n+1} is used instead of the initial approximation of y_n.

The formula of predictor and corrector in 2-point modified block method can be obtained by the derivation of Lagrange interpolation polynomial. For corrector formula, the

interpolation points involved for $y_{n-2}, y_{n-1}, y_n, y_{n+1}$, and y_{n+2} are $\{(x_{n-2}, f_{n-2}), (x_{n-1}, f_{n-1}), (x_n, f_n), (x_{n+1}, f_{n+1}), (x_{n+2}, f_{n+2})\}$ and by using *MAPLE*, the formula in terms of r can be written in the matrix form as the follows:

$$\begin{aligned}
\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} y_{n+1} \\ y_{n+2} \end{bmatrix} &= \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} y_{n+1} \\ y_{n+2} \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y_{n-1} \\ y_n \end{bmatrix} \\
&+ \frac{h}{240} \begin{bmatrix} \dfrac{\left(672r^3 + 144r^2 + 940r^4 + 320r^5\right)}{(r+1)(r+2)(2r+1)r^2} & \dfrac{\left(-21r^3 - 3r^2 - 50r^4 - 40r^5\right)}{(r+1)(r+2)(2r+1)r^2} \\ \dfrac{-\left(-1632r^3 - 1300r^4 - 320r^5 - 624r^2\right)}{(r+1)(r+2)(2r+1)r^2} & \dfrac{-\left(-549r^3 - 610r^4 - 200r^5 - 147r^2\right)}{(r+1)(r+2)(2r+1)r^2} \end{bmatrix} \\
&\times \begin{bmatrix} f_{n+1} \\ f_{n+2} \end{bmatrix} \\
&+ \frac{h}{240} \begin{bmatrix} \dfrac{\left(-176r - 28 - 240r^2\right)}{(r+1)(r+2)(2r+1)r^2} & \dfrac{\left(1029r^3 + 564r^2 + 139r + 14 + 790r^4 + 200r^5\right)}{(r+1)(r+2)(2r+1)r^2} \\ \dfrac{-\left(-92 - 304r - 240r^2\right)}{(r+1)(r+2)(2r+1)r^2} & \dfrac{-\left(46 + 230r^4 + 40r^5 + 501r^3 + 516r^2 + 251r\right)}{(r+1)(r+2)(2r+1)r^2} \end{bmatrix} \\
&\times \begin{bmatrix} f_{n-1} \\ f_n \end{bmatrix} \\
&+ \frac{h}{240} \begin{bmatrix} 0 & \dfrac{\left(37r + 14 + 15r^2\right)}{(r+1)(r+2)(2r+1)r^2} \\ 0 & \dfrac{\left(46 + 53r + 15r^2\right)}{(r+1)(r+2)(2r+1)r^2} \end{bmatrix} \begin{bmatrix} f_{n-3} \\ f_{n-2} \end{bmatrix}.
\end{aligned} \tag{3}$$

Letting us consider $r = 1$ and substituting this value in (3) will produce the following first and second point of the corrector formula:

$$\begin{aligned}
&r = 1, \\
\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} y_{n+1} \\ y_{n+2} \end{bmatrix} &= \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} y_{n+1} \\ y_{n+2} \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y_{n-1} \\ y_n \end{bmatrix} \\
&+ \frac{h}{720} \begin{bmatrix} 346 & -19 \\ 646 & 251 \end{bmatrix} \begin{bmatrix} f_{n+1} \\ f_{n+2} \end{bmatrix} \\
&+ \frac{h}{720} \begin{bmatrix} -74 & 456 \\ 106 & -264 \end{bmatrix} \begin{bmatrix} f_{n-1} \\ f_n \end{bmatrix} \\
&+ \frac{h}{720} \begin{bmatrix} 0 & 11 \\ 0 & -19 \end{bmatrix} \begin{bmatrix} f_{n-3} \\ f_{n-2} \end{bmatrix}.
\end{aligned} \tag{4}$$

The coefficients of the formula (3) are then recalculated whenever the step size changes in each integration of steps. These approaches avoid the storing of the coefficients at the start of the code that may give too many subroutines in the algorithm. The same way in getting the predictor formula is employed by involving the interpolation points $(x_{n-3}, f_{n-3}), \{(x_{n-2}, f_{n-2}), (x_{n-1}, f_{n-1}),$ and $(x_n, f_n)\}$.

Now, we consider the implementation of PE(CE)^s mode where P denotes the application of the predictor of order $(k-1)$, C denotes the application of the corrector of order k, E denotes the evaluation of the function f, and s denotes the number of iterations that is needed in a convergence test. The considered implementation can be described as follows:

$$\begin{aligned}
P: &\begin{cases} {}^{[i]}y_p\left(x_{n+1}\right) = y\left(x_n\right) + h\sum_{q=0}^{3}\beta_{-q}z\left(x_{n-q}\right), \\ {}^{[i]}y_p\left(x_{n+2}\right) = {}^{[i]}y_p\left(x_{n+1}\right) + h\sum_{q=0}^{3}\beta_{-q}z\left(x_{n-q}\right), \end{cases} \\
E: &\begin{cases} {}^{[i]}z_p\left(x_{n+1}\right) = f\left(x_{n+1}, {}^{[i]}y_p\left(x_{n+1}\right), {}^{[i]}y_p\left(x_{n+1}-\tau\right)\right), \\ {}^{[i]}z_p\left(x_{n+2}\right) = f\left(x_{n+2}, {}^{[i]}y_p\left(x_{n+2}\right), {}^{[i]}y_p\left(x_{n+2}-\tau\right)\right), \end{cases} \\
C: &\begin{cases} {}^{[j]}y_c\left(x_{n+1}\right) = y\left(x_n\right) + h\sum_{q=0}^{4}\beta_{2-q}z\left(x_{n+2-q}\right), \\ {}^{[j]}y_c\left(x_{n+2}\right) = {}^{[j]}y_c\left(x_{n+1}\right) + h\sum_{q=0}^{4}\beta_{2-q}z\left(x_{n+2-q}\right), \end{cases} \\
E: &\begin{cases} {}^{[j]}z_c\left(x_{n+1}\right) = f\left(x_{n+1}, {}^{[j]}y_c\left(x_{n+1}\right), {}^{[j]}y_c\left(x_{n+1}-\tau\right)\right), \\ {}^{[j]}z_c\left(x_{n+2}\right) = f\left(x_{n+2}, {}^{[j]}y_c\left(x_{n+2}\right), {}^{[j]}y_c\left(x_{n+2}-\tau\right)\right), \end{cases}
\end{aligned} \tag{5}$$

for $i = 0$ and $j = 1, 2, \ldots, s$.

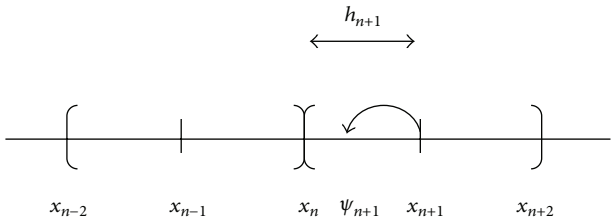

Figure 2: Small lag in 2-point modified block method.

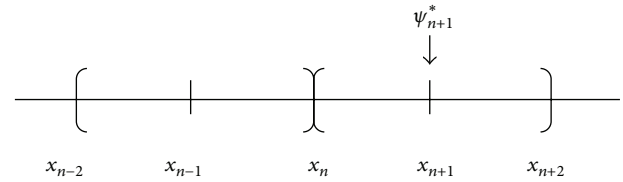

Figure 3: Vanishing lag in 2-point modified block method.

3. Handling Small and Vanishing Lag with Newton Divided Difference Interpolation

In solving delay differential equations, one must be aware of the presence of the delay value in order to achieve the smooth solution and desired accuracy in the numerical solution. During the integration of DDEs, the delay time may even lie in the previous step, current step, or in the next step. The main difficulty in this numerical solution may arise when the delay falls in the current step where no approximation information can be used to evaluate the delay term; in particular, when one-step integration method is applied. But it might be slightly different with 2-point modified block method, where the approximate solution at the current step has been obtained from the application of predictor in PE(CE)^s mode. Therefore, in this section, we will compute the solution of small and vanishing lag using Newton divided difference interpolation by taking five points to interpolate in the code.

3.1. Small Lag. The small lag occurs when the delay time falls in the current step, $[x_n, x_{n+1}]$ which is caused when the delay value is smaller than the step size, $\psi_{n+1} < h_{n+1}$, where $\psi_{n+1} = x_{n+1} - \tau_{n+1}$ as illustrated in Figure 2.

3.2. Vanishing Lag. The vanishing lag may occur when the delay time vanishes as $\tau_{n+1} \to 0$ at the current step ψ^*_{n+1} as illustrated in Figure 3.

Detailed information of vanishing lag can be described by considering the case of problem 1 in Section 7 as follows:

$$y'(x) = 1 - y\left(\exp\left(1 - \frac{1}{x}\right)\right), \quad x \in [0.1, 10]. \tag{6}$$

In this problem, the vanishing lag occurs when the lag at $x = 1$. It can be seen that when we substitute the lag in (6), it yields

$$\begin{aligned} y'(x) &= 1 - y\left(\exp\left(1 - \frac{1}{1}\right)\right). \\ y'(x) &= 1 - y(1), \end{aligned} \tag{7}$$

where the delay term $\psi^*_{n+1} = \exp(1 - (1/1)) = 1 = x_{n+1}$. Basically, if this happens, there is no approximate solution that can be used in the interpolation to evaluate the delay term because the delay falls in the current mesh point.

In the implementation of 2-point modified block method, the location of the delay term, $(x-\tau)$, is sought first to specify whether the delay lies as the small lag or vanishing lag. This DDEB5 code will detect and handle these cases automatically and hence identifying the points that would be involved in the interpolation. In order to get an accurate approximation in delay solution, the number of interpolation points must be chosen properly as the order of interpolation is one order higher than or equal to the order of integration method. Since our method is of order five, we choose five-point number of approximate solution that has been stored closest to the delay value to do the interpolation. The approach of this case can be summarize in Algorithm 1.

For the case if the small and vanishing lags occur at starting point of this 2-point modified block method, the approach of linear extrapolation will be used by determining the solution at previous point, $x_n - h_n$. Then, the solution of delay term can be obtained by extrapolating the points in the interval $[x_n - h, x_n]$.

4. Stability of the Method

The general linear test equation for DDE is

$$\begin{aligned} y'(x) &= \lambda y(x) + \mu y(x - \tau), \quad x \geq x_0, \\ y(x) &= \phi(x), \quad -\tau \leq x \leq x_0, \end{aligned} \tag{8}$$

where λ and μ are complex and $\phi(x)$ is a continuous function.

Definition 1. For the step size h consider the following.

(i) If λ and μ are real in (8), the region R_P in the (H_1, H_2)-plane is called the P-stability region if for any $(H_1, H_2) \in R_P$ the numerical solution of (8) satisfies $y(x_n) \to 0$ as $x_n \to \infty$. The test equation for P-stability is

$$\begin{aligned} y'(x) &= \lambda y(x) + \mu y(x - \tau), \quad x \geq x_0, \\ y(x) &= \phi(x), \quad -\tau \leq x \leq x_0. \end{aligned} \tag{9}$$

(ii) If $\lambda = 0$ and μ is complex in (8), the region R_Q in the (H_2)-plane is called the Q-stability region if for any $(H_2) \in R_Q$ the numerical solution of (8) satisfies $y(x_n) \to 0$ as $x_n \to \infty$. The test equation for Q-stability is

$$\begin{aligned} y'(x) &= \mu y(x - \tau), \quad x \geq x_0, \\ y(x) &= \phi(x), \quad -\tau \leq x \leq x_0. \end{aligned} \tag{10}$$

POINT = number of points to be computed simultaneously in a block,
EQN = number of equations in a system,
DDE$(n+d, X, Y, DX, DY)$ = subroutine function to approximate the delay term,
FN$(n+d, X, XD, YD, XDP, YDP, Y, K)$ = subroutine function of function evaluation.

(1) **for** $d = 1$ **to** POINT **do**
(2) **for** $k = 1$ **to** EQN **do**
(3) **P**: if $d = 1$, then ${}^{[i]}y_p(x_{n+d}) = y(x_n) + h\sum_{q=0}^{3}\beta_{-q}z(x_{n-q})$
if $d = 2$, then ${}^{[i]}y_p(x_{n+d}) = y(x_{n+1}) + h\sum_{q=0}^{3}\beta_{-q}z(x_{n-q})$
(4) end for
(5) end for
(6) **for** $d = 1$ **to** POINT **do**
(7) **E**: subroutine DDE$(n+d, X, Y, DX, DY)$
if $(-\tau \le \psi_{n+d} \le a)$
then $y(\psi_{n+d}) = \phi(\psi_{n+d})$
else
$y(\psi_{n+d}) = Newton\_Divided\_Difference\_Interpolation$
evaluate FN$(n+d, X, XD, YD, XDP, YDP, Y, K)$
(8) end for
(9) **for** $d = 1$ **to** POINT **do**
(10) **for** $k = 1$ **to** EQN **do**
(11) **C**: if $d = 1$, then ${}^{[j]}y_c(x_{n+d}) = y(x_n) + h\sum_{q=0}^{4}\beta_{2-q}z(x_{n+2-q})$,
if $d = 2$, then ${}^{[j]}y_c(x_{n+d}) = y(x_{n+1}) + h\sum_{q=0}^{4}\beta_{2-q}z(x_{n+2-q})$,
(12) end for
(13) end for
(14) **for** $d = 1$ **to** POINT **do**
(15) **E**: subroutine DDE$(n+d, X, Y, DX, DY)$
if$(-\tau \le \psi_{n+d} \le a)$
then $y(\psi_{n+d}) = \phi(\psi_{n+d})$
else
$y(\psi_{n+d}) = Newton\_Divided\_Difference\_Interpolation$
evaluate FN$(n+d, X, XD, YD, XDP, YDP, Y, K)$
(16) end for

Algorithm 1

The corrector formulae of 2-point modified block method in (4) can be written as follows:

$$A_2 Y_{N+2} = A_1 Y_{N+1} + h\sum_{i=0}^{2} B_i F_{N+i}. \tag{11}$$

Applying (11) to the test equations in (9) and rearranging the equation to be equal to zero will give

$$\begin{aligned} A_2 Y_{N+2} &= A_1 Y_{N+1} + h\sum_{i=0}^{2} B_i\left(\lambda Y_{N+i} + \mu Y_{N+i-m}\right). \\ &\left(A_2 - H_1 B_2\right) Y_{N+2} - \left(A_1 + H_1 B_1\right) Y_{N+1} \\ &- H_1 B_0 Y_N - H_2 \sum_{i=0}^{2} B_i Y_{N+i-m} = 0, \end{aligned} \tag{12}$$

where

$$\begin{aligned} A_2 &= \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}, \qquad A_1 = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \\ Y_{N+2} &= \begin{bmatrix} y_{n+1} \\ y_{n+2} \end{bmatrix}, \qquad Y_{N+1} = \begin{bmatrix} y_{n-1} \\ y_n \end{bmatrix}, \\ Y_N &= \begin{bmatrix} y_{n-3} \\ y_{n-2} \end{bmatrix}, \qquad F_{N+2} = \begin{bmatrix} f_{n+1} \\ f_{n+2} \end{bmatrix}, \\ F_{N+1} &= \begin{bmatrix} f_{n-1} \\ f_n \end{bmatrix}, \qquad F_N = \begin{bmatrix} f_{n-3} \\ f_{n-2} \end{bmatrix}, \end{aligned} \tag{13}$$

and the matrices of B_0, B_1, and B_2 are depending on the step size ratio r which has been formulated in corrector formula in (4).

For $r = 1$,

$$\begin{aligned} B_2 &= \frac{1}{720}\begin{bmatrix} 346 & -19 \\ 646 & 251 \end{bmatrix}, \qquad B_1 = \frac{1}{720}\begin{bmatrix} -74 & 456 \\ 106 & -264 \end{bmatrix}, \\ B_0 &= \frac{1}{720}\begin{bmatrix} 0 & 11 \\ 0 & -19 \end{bmatrix}. \end{aligned} \tag{14}$$

Thus, the P-stability polynomial, $\pi(\varsigma)$ of 2-point modified block method, is given by

$$\begin{aligned} \pi(\varsigma) = \det \Bigg| &\left(A_2 - H_1 B_2\right) t^{2+m} - \left(A_1 + H_1 B_1\right) t^{1+m} \\ &- H_1 B_0 t^m - H_2 \sum_{i=0}^{2} B_i t^i \Bigg|. \end{aligned} \tag{15}$$

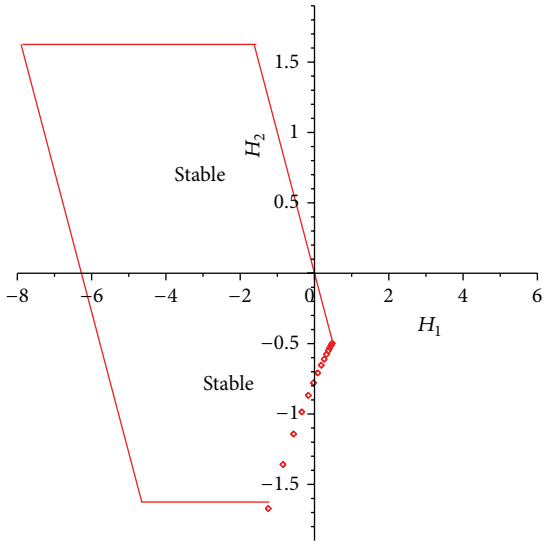

Figure 4: P-stability of 2-point modified block method.

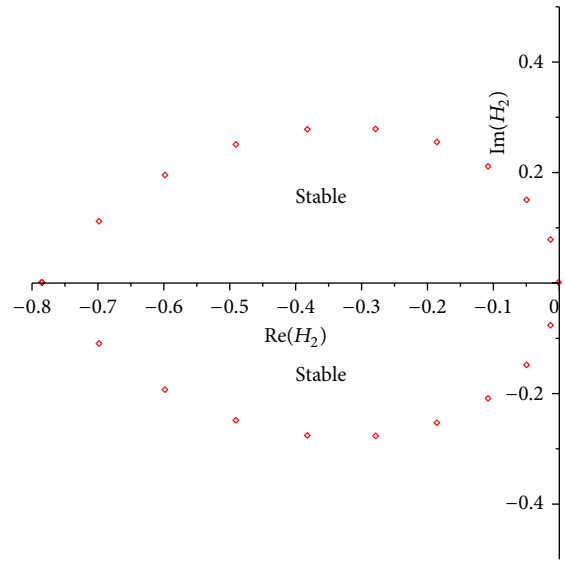

Figure 5: Q-stability of 2-point modified block method.

When the same approaches are applied to the test equation (10), it will give the Q-stability polynomial, $\psi(\varsigma)$, as follows:

$$\psi(\varsigma) = \det\left|A_1 t^{2+m} - A_0 t^{1+m} - H_2 \sum_{i=0}^{2} B_i t^i\right|. \tag{16}$$

By solving $\pi(\varsigma) = 0$ and $\psi(\varsigma) = 0$, therefore the P- and Q-stability regions are as shown in Figures 4 and 5.

5. Order and Error Constant

From (2), we associate that the difference operator L is defined by

$$\begin{aligned} L[y(x);h] &= \sum_{q=0}^{k}\left[\alpha_q y_{n+q} - h\beta_{2-q} f_{n+2-q}\right] \\ &= \sum_{q=0}^{k}\left[\alpha_q y(x+qh) - h\beta_{2-q} y'(x+(2-q)h)\right], \end{aligned} \tag{17}$$

where $y(x) \in C^1[a,b]$ is an arbitrary function. Expanding the functions $y(x+qh)$ and $y'(x+(2-q)h)$ as Taylor series about x gives

$$\begin{aligned} y(x+qh) &= y(x) + qhy'(x) + \frac{(qh)^2}{2!}y''(x) \\ &\quad + \frac{(qh)^3}{3!}y'''(x) + \cdots, \\ y'(x+(2-q)h) &= y'(x) + (2-q)hy''(x) \\ &\quad + \frac{((2-q)h)^2}{2!}y'''(x) \\ &\quad + \frac{((2-q)h)^3}{3!}y^{(4)}(x) + \cdots. \end{aligned} \tag{18}$$

Substitute into (17)

$$\begin{aligned} &L[y(x);h] \\ &= \sum_{q=0}^{k}\left[\alpha_q\left\{y(x) + qhy'(x) + \frac{(qh)^2}{2!}y''(x)\right.\right. \\ &\qquad \left. + \frac{(qh)^3}{3!}y'''(x) + \cdots\right\} \\ &\qquad - h\beta_{2-q}\left\{y'(x) + (2-q)hy''(x)\right. \\ &\qquad\quad + \frac{((2-q)h)^2}{2!}y'''(x) \\ &\qquad\quad \left.\left. + \frac{((2-q)h)^3}{3!}y^{(4)}(x) + \cdots\right\}\right]. \end{aligned} \tag{19}$$

Collecting terms

$$\begin{aligned} &L[y(x);h] \\ &= \sum_{q=0}^{k}\left[\alpha_q y(x) + h\left(q\alpha_q - \beta_{2-q}\right)y'(x)\right. \\ &\qquad + h^2\left(\frac{q^2\alpha_q}{2!} - (2-q)\beta_{2-q}\right)y''(x) \\ &\qquad \left. + h^3\left(\frac{q^3\alpha_q}{3!} - \frac{(2-q)^2\beta_{2-q}}{2!}\right)y'''(x) + \cdots\right], \end{aligned} \tag{20}$$

where

$$C_0 = \sum_{q=0}^{k} \alpha_q,$$

$$C_1 = \sum_{q=0}^{k} \left(q\alpha_q - \beta_{2-q}\right),$$

$$C_2 = \sum_{q=0}^{k} \left(\frac{q^2\alpha_q}{2!} - (2-q)\,\beta_{2-q}\right), \tag{21}$$

$$\vdots$$

$$C_r = \sum_{q=0}^{k} \left(\frac{q^r\alpha_q}{r!} - \frac{(2-q)^{r-1}\beta_{2-q}}{(r-1)!}\right).$$

Definition 2. The multistep block method in (2) and the difference operator (17) are said to be of order P, if $C_0 = C_1 = \cdots = C_P = 0$ and $C_{P+1} \neq 0$.

By applying the formulae in (4), we obtained

$$C_0 = C_1 = C_2 = C_3 = C_4 = C_5 = 0,$$

$$C_6 = \begin{bmatrix} y_{n+1} \\ y_{n+2} \end{bmatrix} = \begin{bmatrix} \frac{11}{1440} \\ \frac{-3}{160} \end{bmatrix} \neq 0. \tag{22}$$

From Definition 2, we can conclude that the 2-point modified block method is of order five with the error constant

$$C_6 = \begin{bmatrix} \frac{11}{1440} & \frac{-3}{160} \end{bmatrix}^T \neq 0. \tag{23}$$

6. Variable Step Size Strategy

The developed algorithm starts by finding the values of starting point at x_{n-2}, x_{n-1}, and x_n by using the Euler method. The initial block in 2-point modified block method can be obtained by the approximation of the solutions y_{n+1} and y_{n+2} with the starting step size ratio $r = 1$ and $q = 1$. For obtaining the next block, the above PE(CE)^s approaches will be repeated until it reaches the end of the interval.

In order to achieve the desired accuracy and the most optimal total steps in the whole interval, we use the Runge-Kutta Fehlberg variable step size strategy which has been introduced by [11]. If the integration step is successful, then the new step size in the next step should be determined by using the following formula:

$$h_{\text{new}} = C \times \left(\frac{\text{TOL}}{E_{k-1}}\right)^{1/4}, \tag{24}$$

where $C = 0.5$ is a safety factor. The code then will recalculate the coefficient of the formula whenever the step size changes by using the step size ratio r and q as follows:

$$r = \frac{h_{\text{old}}}{h}, \qquad q = r_{\text{old}}\left(\frac{h_{\text{old}}}{h}\right), \tag{25}$$

where h_{old} and r_{old} are the step size and the step size ratio in the previous block, respectively, and h is the current step size that is used in the computed block. In the case of failure step, the step size will be half of the h_{old}.

The convergence test is done by the iteration of corrector in the second point as given below:

$$\left|y_{n+2}^{(s+1)} - y_{n+2}^{(s)}\right| < 0.1 \times \text{TOL}. \tag{26}$$

By comparing the corrector formula of the method of order k and the same corrector formulae of order $k-1$ at the second point x_{n+2}, the local truncation error, E_{k-1}, can be estimated as

$$E_{k-1} = \left|y_{n+2}^{(k)} - y_{n+2}^{(k-1)}\right|. \tag{27}$$

Finally, the numerical results are compared with the exact solutions and the maximum error of the mixed test can be defined as follows:

$$\text{MAXE} = \max_{1\le i\le \text{SSTEP}} \left\{ \max_{1\le i\le N} \left| \frac{(y_i)_x - (y(x_i))_x}{A + B(y(x_i))_x} \right| \right\}, \qquad x_i \in [a,b], \tag{28}$$

where $(y_i)_x$ is the xth component of the approximate y, $y(x_i)$ is the exact solution, N is the number of equations in the system, SSTEP is the number of successful steps, and $A = 1, B = 1$.

7. Numerical Results

In this section, we have tested three problems of vanishing lag delay differential equations in order to show the efficiency and reliability on the performance of 2-point modified block method where all these implementations have been tested by using the C program. The following notations are used in the notations section.

Problem 1 ((vanishing lag at $x = 1$), see [3]). Consider

$$y'(x) = 1 - y\left(\exp\left(1 - \frac{1}{x}\right)\right), \quad x \in [0.1, 10],$$
$$y(x) = \ln(x), \quad x \in (0, 0.1]. \tag{29}$$

Exact solution is

$$y(x) = \ln(x). \tag{30}$$

Problem 2 ((state dependent delay with vanishing lag at $x = 1$), see [4]). Consider

$$y_1'(x) = y_2(x), \quad x \in [0.1, 5],$$
$$y_2'(x) = -y_2\left(e^{1-y_2(x)}\right) y_2^2(x)\, e^{1-y_2(x)}, \quad x \in [0.1, 5],$$
$$y_1(x) = \ln(x), \quad x \in [0, 0.1], \tag{31}$$
$$y_2(x) = \frac{1}{x}, \quad x \in [0, 0.1].$$

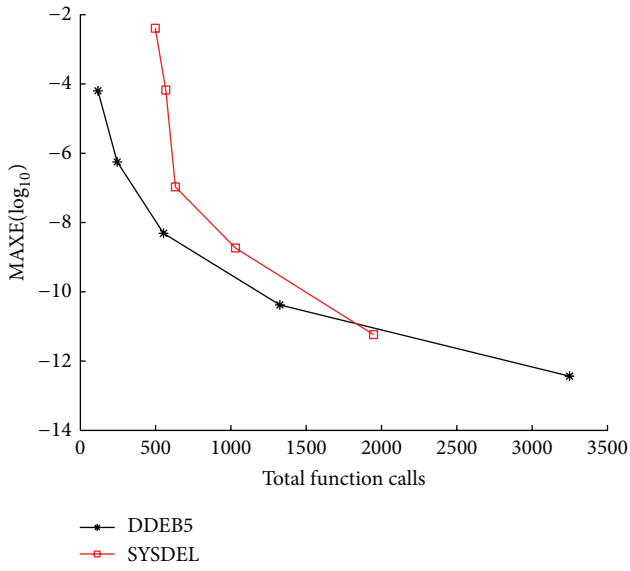

Figure 6: Comparison of total function calls versus maximum error for Problem 1.

Exact solution:

$$y_1(x) = \ln(x), \quad x \in [0.1, 5],$$
$$y_2(x) = \frac{1}{x}, \quad x \in [0.1, 5]. \tag{32}$$

Problem 3 (delay equation with vanishing lag, see [2]). Consider

$$y'(x) = \frac{x^4 - 3}{(x^5 + x)\ln\left(x - x^{-3} + \left(x - x^{-3}\right)^{-3}\right)} y\left(x - x^{-3}\right), \quad x \in [2, 30],$$
$$y(x) = \ln\left(x + x^{-3}\right), \quad x \in [1.5, 2]. \tag{33}$$

Exact solution is

$$y(x) = \ln\left(x + x^{-3}\right). \tag{34}$$

From the numerical results that are tabulated in Table 1 to Table 3, it can be observed that DDEB5 code gave better accuracy of maximum error with less number of function calls in the prescribed tolerances. This is also depicted clearly in the graphs shown in Figures 6, 7, and 8. For example at tolerance 10^{-6} in Table 3, the DDEB5 code only needs 157 total function evaluations and obtains a good accuracy of $7.17E{-}7$, while SYSDEL code needs 483 total function calls and obtains maximum error $1.20E - 4$.

At tolerance 10^{-6} in Table 2, it can be seen that both codes have a comparable maximum error, but DDEB5 required less total function evaluations in the integration. At tolerance 10^{-8} to 10^{-12}, SYSDEL achieved better accuracy, but it needs more function evaluations at each tolerance compared to DDEB5.

Table 1: Numerical results for Problem 1.

TOL	MTD	TS	FS	MAXE	AVERR	FNC
10^{-4}	DDEB5	27	0	$6.33E-05$	$3.13E-05$	117
	SYSDEL	—	—	$4.05E-03$	—	498
10^{-6}	DDEB5	59	0	$5.61E-07$	$2.83E-07$	245
	SYSDEL	—	—	$6.69E-05$	—	568
10^{-8}	DDEB5	136	0	$4.86E-09$	$2.51E-09$	553
	SYSDEL	—	—	$1.07E-07$	—	631
10^{-10}	DDEB5	329	0	$4.18E-11$	$2.27E-11$	1325
	SYSDEL	—	—	$1.83E-09$	—	1030
10^{-12}	DDEB5	810	0	$3.67E-13$	$2.12E-13$	3249
	SYSDEL	—	—	$5.81E-12$	—	1947

Table 2: Numerical results for Problem 2.

TOL	MTD	TS	FS	MAXE	AVERR	FNC
10^{-4}	DDEB5	34	0	$2.04E-04$	$9.10E-05$	145
	SYSDEL	—	—	—	—	—
10^{-6}	DDEB5	68	0	$1.89E-06$	$7.61E-07$	281
	SYSDEL	—	—	$1.85E-06$	—	553
10^{-8}	DDEB5	160	0	$2.00E-08$	$8.33E-09$	649
	SYSDEL	—	—	$5.05E-09$	—	959
10^{-10}	DDEB5	387	0	$2.07E-10$	$8.78E-11$	1557
	SYSDEL	—	—	$2.85E-11$	—	1946
10^{-12}	DDEB5	957	0	$2.10E-12$	$9.03E-13$	3837
	SYSDEL	—	—	$1.11E-13$	—	4214

Table 3: Numerical results for Problem 3.

TOL	MTD	TS	FS	MAXE	AVERR	FNC
10^{-4}	DDEB5	22	0	$3.21E-05$	$7.82E-06$	97
	SYSDEL	—	—	—	—	—
10^{-6}	DDEB5	37	0	$7.17E-07$	$2.59E-07$	157
	SYSDEL	—	—	$1.20E-04$	—	483
10^{-8}	DDEB5	80	0	$1.16E-08$	$3.69E-09$	329
	SYSDEL	—	—	$9.39E-07$	—	588
10^{-10}	DDEB5	188	0	$1.23E-10$	$3.85E-11$	761
	SYSDEL	—	—	$8.23E-09$	—	1099
10^{-12}	DDEB5	459	0	$1.05E-12$	$3.04E-13$	1845
	SYSDEL	—	—	$7.96E-11$	—	2233

Overall, it can be conclude that DDEB5 code has its own advantages in handling small and vanishing lag. This could be justified by the fact that the approach of Newton divided difference interpolation has given better accuracy in approximating delay value compared to extrapolation. The lesser number of function calls also has shown that DDEB5 code does not need more iteration in order to check the convergence of the solutions.

8. Conclusion

The approach of Newton divided difference interpolation in 2-point modified block method is proposed for solving the

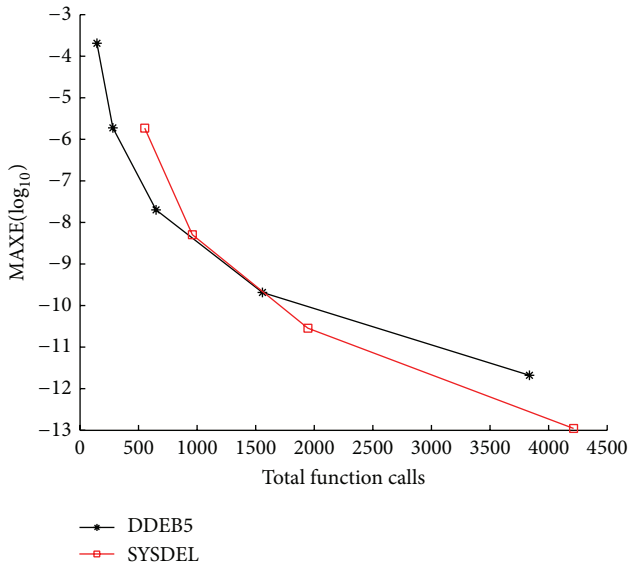

Figure 7: Comparison of total function calls versus maximum error for Problem 2.

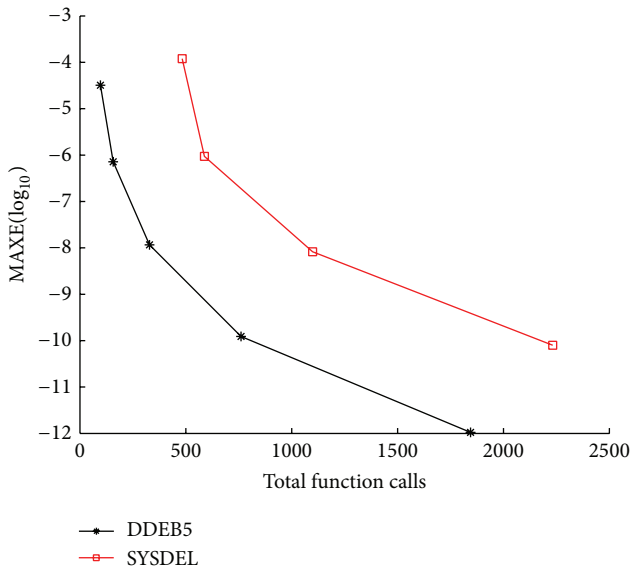

Figure 8: Comparison of total function calls versus maximum error for Problem 3.

case of small and vanishing lag of delay differential equation. The numerical results proved that our developed DDEB5 code is reliable to handle the related difficulty that may exist in DDE. The DDEB5 code also has its own advantages when it can detect and automatically treat the small and vanishing lag in every step of integration without requiring any user guidance. It also has shown the capability in solving a system of state dependent vanishing lag as well.

Notations

TOL: The prescribed tolerance
MTD: Method employed
TS: The total number of steps
FS: The total number of failure steps
MAXE: Maximum value of mixed error test of the computed solution
FNC: Total function calls
DDEB5: A developed code in this paper, which handled the small and vanishing lag by using 5-point Newton divided difference interpolation; the integration method is 2-point modified block method of order five with the variable step size implementation
SYSDEL: A code developed by Karoui and Vaillancourt [2], which handled the vanishing lag by using the extrapolation of 3-point Hermite polynomial; the integration method is Runge-Kutta formula pair of order (5, 6) with the variable step size implementation as described in [11]
—: No data provided in the reference.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors would like to acknowledge the financial support of Putra Grant (GP-IPS/2013/9390100) from Universiti Putra Malaysia and the scholarship of Academic Training Scheme (SLAI) received from Ministry of Education Malaysia.

References

[1] W. H. Enright and M. Hu, "Interpolating Runge-Kutta methods for vanishing delay differential equations," *Computing*, vol. 55, no. 3, pp. 223–236, 1995.

[2] A. Karoui and R. Vaillancourt, "A numerical method for vanishing-lag delay differential equations," *Applied Numerical Mathematics*, vol. 17, no. 4, pp. 383–395, 1995.

[3] H. Yagoub, T. Nguyen-Ba, and R. Vaillancourt, "Variable-step variable-order 3-stage Hermite-Birkhoff-Obrechkoff DDE solver of order 4 to 14," *Applied Mathematics and Computation*, vol. 217, no. 24, pp. 10247–10255, 2011.

[4] H. Hayashi, *Numerical solution of retarded and neutral delay differential equations using continuous Runge-Kutta methods [Ph.D. thesis]*, University of Toronto, Ontario, Canada, 1996.

[5] K. W. Neves and S. Thompson, "Software for the numerical solution of systems of functional-differential equations with state-dependent delays," *Applied Numerical Mathematics*, vol. 9, no. 3–5, pp. 385–401, 1992.

[6] Z. Abdul Majid and M. Suleiman, "Predictor-corrector block iteration method for solving ordinary differential equations," *Sains Malaysiana*, vol. 40, no. 6, pp. 659–664, 2011.

[7] S. Mehrkanoon, Z. A. Majid, and M. Suleiman, "Implementation of 2-point 2-step methods for the solution of first order ODEs," *Applied Mathematical Sciences*, vol. 4, no. 7, pp. 305–316, 2010.

[8] K. H. K. Anuar, K. I. Othman, F. Ishak, Z. B. Ibrahim, and Z. A. Majid, "Development partitioning intervalwise block method for solving ordinary differential equations," *World Academy of Science, Engineering and Technology*, vol. 58, pp. 243–245, 2011.

[9] F. Ishak, Z. A. Majid, and M. Suleiman, "Two point block method in variable step size technique for solving delay differential equations," *Journal of Materials Science and Engineering*, vol. 4, no. 12, pp. 86–90, 2010.

[10] H. C. San, Z. A. Majid, and M. Othman, "Solving delay differential equations using coupled block method," in *Proceedings of the 4th International Conference on Modeling, Simulation and Applied Optimization (ICMSAO '11)*, pp. 1–4, Kuala Lumpur, Malaysia, April 2011.

[11] R. L. Burden and J. D. Faires, *Numerical Analysis*, Thomson Brooks/Cole, 8th edition, 2005.