International Journal of Engineering, Mathematical and Physical Sciences
ISSN: 2517-9934
Vol:5, No:5, 2011

Development of Variable Stepsize Variable Order Block Method in Divided Difference Form for the Numerical Solution of Delay Differential Equations

Fuziyah Ishak, Mohamed B. Suleiman, Zanariah A. Majid and Khairil I. Othman

***Abstract*—**This paper considers the development of a two-point predictor-corrector block method for solving delay differential equations. The formulae are represented in divided difference form and the algorithm is implemented in variable stepsize variable order technique. The block method produces two new values at a single integration step. Numerical results are compared with existing methods and it is evident that the block method performs very well. Stability regions of the block method are also investigated.

***Keywords*—**block method, delay differential equations, predictor-corrector, stability region, variable stepsize variable order.

I. Introduction

In this paper we consider the development of a numerical method for solving systems of first order delay differential equations (DDEs) of the form:

$$\begin{cases} y'(x) = f(x, y(x), y(x-\tau_i)), & x \in [a,b], \tau_i > 0, \\ y(x) = \phi(x), & x \in [\bar{a}, a], \end{cases} \quad (1)$$

where $\phi(x)$ is the initial function and τ_i, for $i = 1,2,\cdots,n,$ where n is an integer such that $n \geq 1$ is the lag function and $\bar{a} = \min_{x \in [a,b]} (x - \tau_i)$. The lag can be constant, time dependent where $\tau_i = \tau_i(x)$ or state dependent, that is $\tau_i = \tau_i(x, y(x))$. The expression $y(x-\tau_i)$ is the solution of the delay term, or simply called the delay term. The function f where $f : [a,b] \times C([\bar{a},b], R^m) \times C([\bar{a},b], R^m) \to R^m$ is continuous and satisfies a Lipschitz condition which guarantees the existence of a unique solution of (1). Here, $C([\bar{a},b], R^m)$ denotes the space of continuous functions mapping $[\bar{a},b]$ into R^m for an integer $m \geq 1$.

Most numerical methods for solving DDEs are adapted from that of numerical methods for ordinary differential equations (ODEs). For some of the earlier work, please refer to [1]-[6]. These methods approximate only one new solution at each integration step. Block methods, however, produce more than one approximate solution at every step. Greater efficiency is achieved because the total number of steps taken can be reduced. Block methods are also the natural candidates for parallel computations because these solutions can be computed simultaneously at every step. Block methods for solving ODEs have been proposed by [7]-[10].

The purpose of this paper is to develop a variable stepsize variable order (VSVO) two-point block method for solving (1). The method produces two new solutions at every step and varies the stepsize and order while achieving the desired accuracy as efficiently as possible. In variable stepsize, the integration coefficients need to be recalculated when the stepsize changes. The recalculation cost can be minimize by representing the formulae in divided difference form because the coefficients are calculated using a simple recursive formula. The performance of the method will be compared with the existing methods SNDDELM in [5] and S2PBDI in [11]. SDDDELM is an all purpose non-block VSVO algorithm where the formulae are represented in modified divided difference form. S2PBDI is a predictor-corrector two-point VSVO block method in divided difference form. The order of the corrector for S2PBDI differs from the order of the method developed here.

The organization of this paper is as follows. In section II, we describe the development of the VSVO predictor-corrector block method. Absolute stability for the two-point predictor-corrector block method is discussed in section III. Numerical results are presented in section IV. Finally section V is the conclusion.

II. Method Development

For simplicity, we consider only single-delay scalar equation of (1). However, the results can easily be extended to systems of equations with multiple delays.

From (1), we let $i = 1$ and $\tau_i \equiv \tau$. Consider a non-uniform grid given by $\bar{a} \leq a = x_0 < \cdots < x_n < x_{n+1} < \cdots \leq b$. We denote the approximation to $y(x)$, the solution of (1) by $y_h(x)$. The predicted value of $y_h(x)$ is denoted as $p_h(x)$. We also denote the expression $f(x_n, y_h(x_n), y_h(\alpha)), \ \alpha = x_n - \tau$ by f_n. Assume that $y_h(x)$ has been computed for $x \in [\bar{a}, x_n]$. The immediate task is to evaluate $y_h(x_{n+1}), y_h(x_{n+2})$, and their corresponding delay terms. The formulae are

F. Ishak and K. I. Othman are with the Mathematics Department, Universiti Teknologi MARA, 40450 Shah Alam, Malaysia (e-mail: fuziyah@tmsk.uitm.edu.my).

M. B. Suleiman and Z. A. Majid are with the Institute for Mathematical Research, Universiti Putra Malaysia, 43400, Serdang, Malaysia.

implemented in PECE mode where P stands for an application of a predictor, E stands for an evaluation of a function f, and C stands for an application of a corrector.

Let $P_{k,n}$ be the interpolating polynomial of degree $k-1$ interpolating f at points

$$(x_n, f_n), (x_{n-1}, f_{n-1}), \ldots, (x_{n-k+1}, f_{n-k+1}).$$

In divided difference form, the interpolating polynomial can be written as

$$P_{k,n}(x) = f[x_n] + \sum_{i=1}^{k-1} f[x_n, x_{n-1}, \ldots, x_{n-i}] \prod_{j=0}^{i-1} (x - x_{n-j}),$$

where

$$f[x_n, x_{n-1}, \ldots, x_{n-k+1}] = \frac{f[x_n . x_{n-1}, \ldots, x_{n-k+2}] - f[x_{n-1}, x_{n-2}, \ldots, x_{n-k+1}]}{x_n - x_{n-k+1}}.$$

Integrating (1) from x_n to x_{n+d}, $d = 1,2$ and replacing f with $P_{k,n}$ yields the following predicted values simultaneously at the grid points,

$$p_h(x_{n+d}) = y_h(x_n) + \int_{x_n}^{x_{n+d}} P(\xi) d\xi$$
$$= y_h(x_n) + \sum_{i=0}^{k-1} f[x_n, x_{n-1}, \ldots, x_{n-i}] \int_{x_n}^{x_{n+d}} c_{n,i}(\xi) d\xi,$$

where ξ is an independent dummy variable and

$$c_{n,i}(x) = \begin{cases} 1, & i = 0, \\ (x - x_n)(x - x_{n-1}) \cdots (x - x_{n-i+1}), & i \geq 1. \end{cases}$$

The predicted values of the derivative are given by

$$p_h'(x_{n+d}) = \sum_{i=0}^{k-1} c_{n,i}(x_{n+d}) f[x_n, x_{n-1}, \cdots x_{n-i}].$$

Let the *t*-fold integral of $c_{n,i}(x)$ be denoted as $c_{n,i}^{(-t)}(x)$. Now define $g_{i,t}(x)$ as follows,

$$g_{i,t}(x) = \begin{cases} c_{n,i}(x), & t = 0, \\ c_{n,i}^{(-t)}(x), & t \geq 1. \end{cases}$$

The formula for $g_{i,t}(x)$ where $t \geq 1, i \geq 1$ can be obtained recursively by the relation

$$g_{i,t}(x) = (x - x_{n-i+1}) g_{i-1,t}(x) - t g_{i-1,t+1}(x).$$

The predicted values at the grid points are given by

$$p_h(x_{n+d}) = y_h(x_n) + \sum_{i=0}^{k-1} g_{i,1}(x_{n+d}) f[x_n, x_{n-1}, \ldots, x_{n-i}],$$

and

$$p_h'(x_{n+d}) = \sum_{i=0}^{k-1} g_{i,0}(x_{n+d}) f[x_n, x_{n-1}, \ldots, x_{n-i}].$$

We now describe the corrector for the first point. Consider the interpolating polynomial $P_{k+1,n+1}$ of degree k that interpolates f at points

$$(x_{n+1}, f_{n+1}^P), (x_n, f_n), (x_{n-1}, f_{n-1}), \ldots, (x_{n-k+1}, f_{n-k+1}),$$

where

$$f_{n+1}^p = f(x_{n+1}, p_h(x_{n+1}), p_h(x_{n+1} - \tau)).$$

Clearly the interpolating polynomial $P_{k+1,n+1}$ can be written as

$$P_{k+1,n+1}(x) = P_{k,n}(x) + c_{n,k}(x) f^p[x_{n+1}, x_n, x_{n-1}, \ldots, x_{n-k+1}].$$

The notation $f^p[x_{n+1}, x_n, x_{n-1}, \ldots, x_{n-k+1}]$ is referred to the divided difference that uses f_{n+1}^p for the points described above. Replacing f in (1) with the polynomial $P_{k+1,n+1}$ and integrating from x_n to x_{n+1}, the first point corrector formula is obtained as,

$$y_h(x_{n+1}) = p_h(x_{n+1}) + g_{k,1}(x_{n+1}) f^p[x_{n+1}, x_n, x_{n-1}, \ldots, x_{n-k+1}].$$

The corrected value of the derivative is given by

$$y_h'(x_{n+1}) = p_h'(x_{n+1}) + g_{k,0}(x_{n+1}) f^p[x_{n+1}, x_n, x_{n-1}, \ldots, x_{n-k+1}].$$

For the second point corrector, consider the interpolating polynomial $P_{k+2,n+2}$ of degree $k+1$ that interpolates f at

$$(x_{n+2}, f_{n+2}^P), (x_{n+1}, f_{n+1}^P), (x_n, f_n), (x_{n-1}, f_{n-1}),$$
$$\ldots, (x_{n-k+1}, f_{n-k+1}),$$

where

$$f_{n+1}^p = f(x_{n+1}, p_h(x_{n+1}), p_h(x_{n+1} - \tau))$$

and

$$f_{n+2}^p = f(x_{n+2}, p_h(x_{n+2}), p_h(x_{n+2} - \tau)).$$

The interpolating polynomial $P_{k+2,n+2}$ can be written as

$$P_{k+2,n+2}(x) = P_{k,n}(x) + c_{n,k}(x) f^p[x_{n+1}, x_n, x_{n-1}, \ldots, x_{n-k+1}]$$
$$+ c_{n+1,k+1}(x) f^p[x_{n+2,} x_{n+1}, x_n, \ldots, x_{n-k+1}].$$

The notation $f^p[x_{n+2,} x_{n+1}, x_n, \ldots, x_{n-k+1}]$ is referred to the divided difference with the interpolations points f_{n+1}^p and f_{n+2}^p. Replacing f in (1) with the polynomial $P_{k+2,n+2}$ and integrating from x_n to x_{n+2}, the formulae for the second point corrector are obtained as

$$y_h(x_{n+2}) = p_h(x_{n+2})$$
$$+ g_{k,1}(x_{n+2}) f^p[x_{n+1}, x_n, x_{n-1}, \ldots, x_{n-k+1}]$$
$$+ (h g_{k,1}(x_{n+2}) - g_{k,2}(x_{n+2})) f^p[x_{n+2}, x_{n+1}, x_n, \ldots, x_{n-k+1}],$$

and

$$y_h'(x_{n+2}) = p_h'(x_{n+2})$$
$$+ g_{k,0}(x_{n+2}) f^p[x_{n+1}, x_n, x_{n-1}, \ldots, x_{n-k+1}]$$
$$+ h g_{k,0}(x_{n+2}) f^p[x_{n+2}, x_{n+1}, x_n, \ldots, x_{n-k+1}].$$

For computational purposes, we let e_d, $d = 1,2$ be defined as

$$e_d = y_h'(x_{n+d}) - p_h'(x_{n+d}).$$

Thus

$$f^p[x_{n+1}, x_n, x_{n-1}, \ldots, x_{n-k+1}] = \frac{e_1}{g_{k,0}(x_{n+1})}$$

and

$$f^{p}[x_{n+2},x_{n+1},x_n,x_{n-1},\ldots,x_{n-k+1}]=\frac{1}{h}\left(\frac{e_2}{g_{k,0}(x_{n+2})}-\frac{e_1}{g_{k,0}(x_{n+1})}\right).$$

Therefore, the corrector formulae for the first point can be written as

$$y_h(x_{n+1}) = p_h(x_{n+1})+\frac{g_{k,1}(x_{n+1})}{g_{k,0}(x_{n+1})}e_1,$$
$$y_h'(x_{n+1}) = p_h'(x_{n+1})+e_1.$$

The corrector formulae for the second point can be written as

$$y_h(x_{n+2}) = p_h(x_{n+2})+\frac{g_{k,2}(x_{n+2})}{hg_{k,0}(x_{n+1})}e_1 + \left(\frac{hg_{k,1}(x_{n+2})-g_{k,2}(x_{n+2})}{hg_{k,0}(x_{n+2})}\right)e_2,$$
$$y_h'(x_{n+2}) = p_h'(x_{n+2})+e_2.$$

Approximation formulae for the delay terms $y_h(x_{n+1}-\tau)$ and $y_h(x_{n+2}-\tau)$ are derived following the earlier discussions for formulae at grid points. For $d=1,2$, let $\alpha_d = x_{n+d}-\tau$ be the delay argument. If $\alpha_d \le a$, the predictor and the corrector are evaluated by using the initial function, that is, $p_h(\alpha_d)=y_h(\alpha_d)=\phi(\alpha_d)$. If $a<\alpha_d\le x_n$, we determine the exact location of α_d by finding the value of j such that $x_j<\alpha_d\le x_{j+1}$. The predictor is given by

$$p_h(\alpha_d)=y_h(x_j)+\sum_{i=0}^{k_{j+1}-1} g_{i,1}(\alpha_d)\,f[x_j,x_{j-1},\ldots,x_{j-i}],$$

and the corrector is given by

$$y_h(\alpha_d)=p_h(\alpha_d)+g_{k_{j+1},1}(\alpha_d)\,f^{p}[x_{j+1},x_j,x_{j-1},\ldots,x_{j-k+1}].$$

If $x_n<\alpha_d\le x_{n+1}$, the predictor is given by

$$p_h(\alpha_d)=y_h(x_n)+\sum_{i=0}^{k-1} g_{i,1}(\alpha_d)\,f[x_n,x_{n-1},\ldots,x_{n-i}],$$

and the corrector is given by

$$y_h(\alpha_d) = p_h(\alpha_d)+g_{k,1}(\alpha_d)\frac{e_1}{g_{k,0}(x_{n+1})}.$$

If $x_{n+1}<\alpha_2\le x_{n+2}$,

$$y_h(\alpha_2) = p_h(\alpha_2)+\frac{g_{k,1}(\alpha_2)}{g_{k,0}(x_{n+1})}e_1 + \left(\frac{(\alpha_2-x_{n+1})g_{k,1}(\alpha_2)-g_{k,2}(\alpha_2)}{h}\right)\times\left(\frac{e_2}{g_{k,0}(x_{n+2})}-\frac{e_1}{g_{k,0}(x_{n+1})}\right).$$

Local errors at the grid points are estimated by comparing the formulae of different orders. The error magnitude of the first point is presumably higher than that of the second point because the approximation at the first point is one order less than the approximation at the second point. By controlling the error at the first point, the method produces better result. We simply refer to the estimated local error at the first point as the local error. From [11], the local error when applying the formulae of order k, denoted by E_k is given by

$$E_k = \frac{-g_{k-1,2}(x_{n+1})}{g_{k,0}(x_{n+1})}e_1. \quad (2)$$

We also calculate local errors for formulae of orders $k-2$, $k-1$, and $k+1$ because these errors will facilitate in the stepsize and order changing strategy. Local errors of adjacent orders E_{k-2}, E_{k-1}, and E_{k+1} can be calculated directly from (2) by exchanging k with $k-2$, $k-1$, and $k+1$ respectively.

This VSVO algorithm is self-starting. A method of order one with a suitable initial stepsize starts the integration. The algorithm varies the order while taking the most optimal stepsize to achieve the desired accuracy as efficiently as possible. A step is accepted if

$$E_k < \mathrm{TOL}\times(A+B\times p_h(x_{n+1}))$$

where TOL is the user specified tolerance and the values of *A* and *B* are either 1 or 0 depending upon whether we use relative, absolute or mixed error test. Whenever a step fails, the new stepsize is reduced by half. New order is selected before the integration step is repeated. When a step is accepted, we select a new order k_{new} based on the following strategy. The order is reduced by one, that is, $k_{\text{new}}=k-1$ if we encounter the following criteria,

- for $k>2$, $\max(|E_{k-1}|,|E_{k-2}|)\le|E_k|$,
- for $k=2$, $|E_{k-1}|\le 0.5|E_k|$,
- if $|E_{k+1}|$ is available, $k>1$ and $|E_{k-1}|\le\min(|E_k|,|E_{k+1}|)$.

We consider raising the order by one only after $k+1$ successful steps at constant stepsize. The new order $k_{\text{new}}=k+1$ if

- for $k=1$, $|E_{k+1}|<0.5|E_k|$,
- for $k>1$, $|E_{k+1}|<|E_k|<\max(|E_{k-1}|,|E_{k-2}|)$.

If none of the above criteria is satisfied, $k_{\text{new}}=k$.

Once the new order is selected, it is renamed k and the corresponding E_k is used in selecting the new stepsize. For a successful step, the new stepsize is given by

$$h_{\text{new}}=\begin{cases}2h, & \text{if } \hat{R}\ge 2,\\ \hat{R}h, & \text{if } 1.6<\hat{R}<2 \text{ or } 0.5\le\hat{R}<0.9,\\ 0.5h, & \text{if } \hat{R}<0.5,\\ h, & \text{otherwise,}\end{cases}$$

where

$$\hat{R}=0.8R,\ R=\left(\frac{\mathrm{TOL}}{\theta_{n+1}|E_k|}\right)^{\frac{1}{k+1}}$$

and

$$\frac{1}{\theta_{n+1}}=A+B\times|y_h(x_{n+1})|.$$

We multiply R with a safety factor of 0.8 in order to give a more conservative estimate for the new stepsize. This practice reduces the number of steps being rejected.

Derivative discontinuities in the solutions of DDEs can be detected whenever the magnitude of the local error is too big as compared with a given tolerance. The algorithm reduces the stepsize and order of the method accordingly in order to reduce the magnitude of local error. The process is repeated until the discontinuity point is included in the grid.

III. Absolute Stability

It is of practical importance to study the behavior of the global error where the numerical solution is expected to behave as the exact solution does when the independent variable x approaches infinity. Many authors including [12]-[15] studied various concepts of stability based on test equations with known stability regions. Common test equations are

$$\begin{aligned} &y'(x) = \lambda y(x) + \mu y(x-\tau), \quad x \geq x_0, \\ &y(x) = \phi(x), \qquad\qquad -\tau \leq x \leq x_0, \end{aligned} \tag{3}$$

and

$$\begin{aligned} &y'(x) = \mu y(x-\tau), \quad x \geq x_0, \\ &y(x) = \phi(x), \qquad -\tau \leq x \leq x_0, \end{aligned} \tag{4}$$

where λ and μ are complex numbers.

We refer to [14] for the definitions of P- and Q-stability regions. Let h be a fixed stepsize such that $x_n = x_0 + nh$ and $mh = \tau,\ m \in I^+$. Let $H_1 = h\lambda$ and $H_2 = h\mu$.

Definition 1: For a fixed stepsize h and $\lambda, \mu \in R$ in (4), 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 (3) vanishes as $x_n \to \infty$.

Definition 2: For a fixed stepsize h, $\mu \in C$ in (4), the region R_Q in the complex H_2-plane is called the Q-stability region if for any $H_2 \in R_Q$, the numerical solution of (4) vanishes as $x_n \to \infty$.

For notational purposes, let $n = 2N$. We define the matrices Y_N^p, Y_N^c, F_N^p and F_N^c as the following,

$$\begin{aligned} Y_N^p &= [y_n^c, y_{n+1}^c, \ldots, y_{n+k-2}^c, y_{n+k-1}^p, y_{n+k}^p, y_{n+k+1}^p]^T, \\ Y_N^c &= [y_n^c, y_{n+1}^c, \ldots, y_{n+k-2}^c, y_{n+k-1}^c, y_{n+k}^c, y_{n+k+1}^c]^T, \\ F_N^p &= [f_n^c, f_{n+1}^c, \ldots, f_{n+k-2}^c, f_{n+k-1}^p, f_{n+k}^p, f_{n+k+1}^p]^T, \\ F_N^c &= [f_n^c, f_{n+1}^c, \ldots, f_{n+k-2}^c, f_{n+k-1}^c, f_{n+k}^c, f_{n+k+1}^c]^T. \end{aligned}$$

For simplicity, we denote the predicted value of $y_h(x_{n+k})$ by y_{n+k}^p and the corrected value by y_{n+k}^c. The predictor-corrector formulae for the two-point block method are given by

$$\begin{aligned} Y_{N+1}^p &= CY_N^c + hDF_N^c, \\ Y_{N+1}^c &= SY_{N+1}^p + hTF_{N+1}^p. \end{aligned} \tag{5}$$

Here, *C, D, S* and *T* are $(k+2)\times(k+2)$ square matrices given by,

$$C = \begin{bmatrix} 0 & 0 & & & \\ \vdots & \vdots & & I_{k\times k} & \\ 0 & 0 & & & \\ 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix},$$

$$D = \begin{bmatrix} 0 & \cdots & & \cdots & & \cdots & 0 \\ \vdots & & & & & & \vdots \\ 0 & \cdots & & \cdots & & \cdots & 0 \\ 0 & 0 & d_{10} & d_{11} & \cdots & d_{1k-2} & d_{1k-1} \\ 0 & 0 & d_{20} & d_{21} & \cdots & d_{2k-2} & d_{2k-1} \end{bmatrix},$$

$$S = \begin{bmatrix} & & & & 0 & 0 \\ & I_{k\times k} & & & \vdots & \vdots \\ 0 & \cdots & 0 & 1 & 0 & 0 \\ 0 & \cdots & 0 & 1 & 0 & 0 \end{bmatrix}, \text{ and}$$

$$T = \begin{bmatrix} 0 & & \cdots & & 0 \\ \vdots & & & & \vdots \\ 0 & & \cdots & & 0 \\ t_{10} & t_{11} & \cdots & t_{1k} & t_{1k+1} \\ t_{20} & t_{21} & \cdots & t_{2k} & t_{2k+1} \end{bmatrix},$$

where $I_{k\div k}$ is the $k \times k$ identity matrix and d_{ij} and t_{ij} are the integration coefficients. The application of (5) to (3) yields

$$\begin{aligned} Y_{N+1}^p &= (C + H_1 D)Y_N^c + H_2 D Y_{N-m}^c, \\ Y_{N+1}^c &= (S + H_1 T)Y_{N+1}^p + H_2 T Y_{N+1-m}^p. \end{aligned}$$

Consequently, the P-stability polynomial of the two-point predictor-corrector block method, $\pi_{2PECE,m}(H_1, H_2; \varsigma)$ is given by

$$\begin{aligned} &\det[I\varsigma^{2m+1} - ((S + H_1 T)(C + H_1 D))\varsigma^{2m} - \\ &((H_2(S + H_1 T)D + H_2 T(C + H_1 D))\varsigma^m - H_2^2 TD], \end{aligned}$$

and the Q-stability polynomial, $\psi_{2PECE,m}(H_2; \varsigma)$ is given by

$$\det[I\varsigma^{2m+1} - SC\varsigma^{2m} - H_2(SD + TC)\varsigma^m - H_2^2 TD],$$

where det[.] is the determinant of a square matrix.

The P- and Q-stability regions are obtained by using boundary locus technique. Inside the regions of absolute stability, the magnitudes of all the roots of the P- and Q-stability polynomials are less than one. As an illustration, the P- and Q-stability regions for $k = 1$ are shown in Fig. 1 and Fig. 2 respectively. The Q-stability regions are symmetric about the real axis and Fig.2 represents the regions in the plane where $\Im(H_2) > 0$.

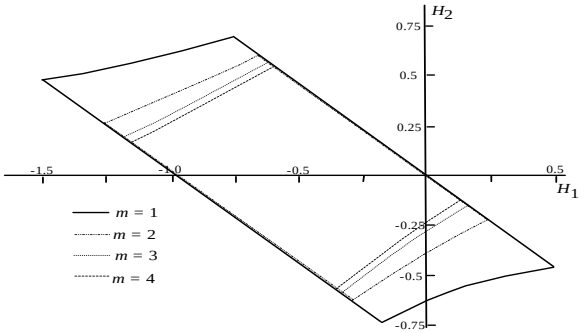

Fig. 1 P-stability regions for the predictor-corrector block method, $k = 1$

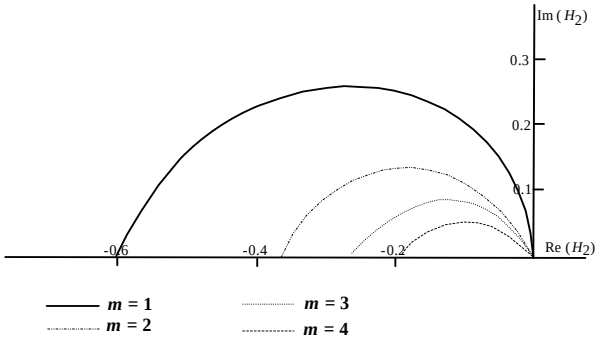

Fig. 2 Q-stability regions for the predictor-corrector block method, $k = 1$

IV. Numerical Results

We test the performance of the block method on various examples from [1]. The performance of the block method is compared with the methods S2PBDI from [11] and SNDDELM from [5]. The examples used are as follows,

Example1:

$$y'(x) = \cos(x)y(y(x)-2), \quad 0 \le x \le 50,$$
$$y(x) = 1, \qquad x \le 0.$$

The exact solution is $y(x) = \sin(x) + 1$.

Example 2:

$$y'(x) = -y(x-1+\exp(-x)) + \sin(x-1+\exp(-x)) + \cos(x), \quad 0 \le x \le 10,$$
$$y(x) = \sin(x), \qquad x \le 0.$$

The exact solution is $y(x) = \sin(x)$.

Example 3:

$$y_1'(x) = -y_1\left(x - \frac{\pi}{2}\right), \quad \frac{\pi}{2} \le x \le 10,$$
$$y_2'(x) = -y_2\left(x - \frac{\pi}{2}\right), \quad \frac{\pi}{2} \le x \le 10,$$
$$y_1(x) = \sin(x), \qquad x \le \frac{\pi}{2},$$
$$y_2(x) = \cos(x), \qquad x \le \frac{\pi}{2}.$$

The exact solution is $y_1(x) = \sin(x)$ and $y_2(x) = \cos(x)$.

Example 4:

$$y'(x) = 1 - y\left(\exp\left(1 - \frac{1}{x}\right)\right), \quad 2 \le x \le 100,$$
$$y(x) = \ln(x), \qquad 0 \le x \le 2.$$

The exact solution is $y(x) = \ln(x)$, $x > 0$.

The error err_i at the grid point for each component is defined as

$$(err_i)_t = \left| \frac{(y_h(x_i))_t - (y(x_i))_t}{A + B(y(x_i))_t} \right|$$

where $(y)_t$ is the t-th component of y and $y(x_i)$ is the exact solution at x_i. A and B may take the values of either 1 or 0 depending upon the kind of error test chosen. In this case, we used mixed error test where $A = 1$ and $B = 1$ as opposed to absolute error test ($A = 1$ and $B = 0$) and relative error test ($A = 0$ and $B = 1$). The maximum error is defined as follows,

$$\text{MAXE} = \max_{1 \le i \le \text{SSTEP}} \left(\max_{1 \le t \le N} (err_i)_t \right),$$

where N is the number of equations in the system and SSTEP is the total number of successful steps. The experiment was executed on Sun Fire V1280 by using one processor. The machine is located at the Institute for Mathematical Research (INSPEM), Universiti Putra Malaysia.

The numerical results are given in Table I – Table IV. The following abbreviations are used to describe the numerical solutions:

TOL	the chosen tolerance,
MTD	method employed,
STP	total number of steps,
FS	total number of failed steps,
MAXE	maximum error,
TIME	time taken in microseconds,
S2PBTI	the algorithm described in the paper,
S2PBDI	the algorithm in [11],
SNDDELM	the algorithm in [5],
2.47518E-02	is equivalent to 2.47518×10^{-2}.

TABLE I
Numerical Results For Example 1

TOL	MTD	STP	FS	MAXE	TIME
10^{-2}	S2PBTI	55	7	4.86496E-01	1725
	S2PBDI	62	4	2.54222E-02	2858
	SNDDELM	81	3	2.44098E-03	9026
10^{-4}	S2PBTI	76	5	1.77338E-04	2374
	S2PBDI	88	3	1.81656E-04	4017
	SNDDELM	120	6	3.10620E-05	14811
10^{-6}	S2PBTI	125	6	5.15951E-07	4346
	S2PBDI	126	5	1.48025E-06	5588
	SNDDELM	136	3	5.12003E-06	16118
10^{-8}	S2PBTI	179	13	1.73155E-08	7137
	S2PBDI	148	0	1.04934E-09	8185
	SNDDELM	180	2	5.01504E-08	21591
10^{-10}	S2PBTI	171	1	1.08162E-11	9841
	S2PBDI	214	7	5.99198E-11	10923
	SNDDELM	282	7	6.55478E-09	35516

TABLE II
NUMERICAL RESULTS FOR EXAMPLE 2

TOL	MTD	STP	FS	MAXE	TIME
10^{-2}	S2PBTI	18	0	3.33076E-02	778
	S2PBDI	21	0	3.05156E-03	1162
	SNDDELM	22	1	3.76223E-03	1446
10^{-4}	S2PBTI	25	0	1.13313E-04	1353
	S2PBDI	30	0	1.04934E-04	1694
	SNDDELM	36	2	3.01735E-04	3788
10^{-6}	S2PBTI	38	0	2.55530E-07	2092
	S2PBDI	66	5	1.95644E-06	3780
	SNDDELM	62	4	3.98644E-06	6260
10^{-8}	S2PBTI	55	1	2.89023E-09	3688
	S2PBDI	62	1	1.01360E-08	4991
	SNDDELM	82	6	3.00713E-09	9078
10^{-10}	S2PBTI	68	0	1.07294E-10	4873
	S2PBDI	63	2	3.67489E-10	7534
	SNDDELM	120	11	9.63660E-09	16576

TABLE III
NUMERICAL RESULTS FOR EXAMPLE 3

TOL	MTD	STP	FS	MAXE	TIME
10^{-2}	S2PBTI	17	0	1.40301E-03	1240
	S2PBDI	21	1	4.25522E-03	1723
	SNDDELM	18	1	2.78831E-03	3409
10^{-4}	S2PBTI	25	0	1.50308E-05	1588
	S2PBDI	28	0	1.10485E-04	1767
	SNDDELM	35	3	2.41004E-04	7288
10^{-6}	S2PBTI	35	0	3.02703E-07	2629
	S2PBDI	40	0	6.56416E-07	3397
	SNDDELM	51	4	5.79068E-06	10458
10^{-8}	S2PBTI	45	0	7.06017E-09	4102
	S2PBDI	61	0	6.23463E-09	6395
	SNDDELM	55	1	4.93816E-09	10646
10^{-10}	S2PBTI	62	0	7.63623E-11	5766
	S2PBDI	74	0	7.69796E-10	8895
	SNDDELM	77	1	1.04197E-09	16172

TABLE IV
NUMERICAL RESULTS FOR EXAMPLE 4

TOL	MTD	STP	FS	MAXE	TIME
10^{-2}	S2PBTI	25	0	3.02890E-03	2043
	S2PBDI	31	0	5.67125E-04	2933
	SNDDELM	29	0	1.30425E-03	1420
10^{-4}	S2PBTI	37	0	6.51784E-06	1522
	S2PBDI	45	0	3.65948E-05	2086
	SNDDELM	54	0	4.58017E-05	4169
10^{-6}	S2PBTI	53	0	9.51416E-07	2537
	S2PBDI	62	0	2.00824E-06	3539
	SNDDELM	86	2	9.49873E-06	8185
10^{-8}	S2PBTI	72	0	1.83465E-08	3484
	S2PBDI	85	0	6.48735E-08	5195
	SNDDELM	128	6	1.58750E-07	18088
10^{-10}	S2PBTI	97	0	8.92104E-11	6513
	S2PBDI	113	0	1.45530E-10	8735
	SNDDELM	167	6	5.97529E-09	25699

From the numerical results, it is evident that S2PBTI achieves the desired accuracy and performs very well when compared to S2PBDI and SNDDELM. Using corrector of one order higher than S2PBDI results in reduction number of total steps and time taken. The predictor-corrector formulae in divided difference form as opposed to the modified divided difference form in SNDDELM do not reduce the performance of the method. Overall results show that total steps taken in S2PBTI are reduced. The simpler formulation together with the reduction in total steps reduces the integration time when compared with S2PBDI and SNDDELM.

V. CONCLUSION

We have described the development of a two-point predictor-corrector block method for solving DDEs. The formulae for the VSVO algorithm in PECE mode are represented in divided difference form. The corrector for the second point uses one extra point resulting from the prediction process. The P- and Q-stability polynomials of the predictor-corrector block method are determined and the stability regions indicate that the method is suitable for solving nonstiff DDEs. From the numerical results we conclude that the two-point block method is efficient, accurate and reliable for solving DDEs.

REFERENCES

[1] A.N. Al-Mutib, *Numerical Methods for Solving Delay Differential Equations*. PhD. Thesis, University of Manchester, 1977.
[2] Z. Jackiewicz, "Variable-step variable-order algorithm for the numerical solution of neutral functional differential equations," *Applied Numer. Math.*, vol. 4, pp. 317-329, 1987.
[3] Z. Jackiewicz and E. Lo, "The numerical solution of neutral functional differential equations by Adams predictor-corrector methods," *Applied Numer. Math*., vol. 8, pp. 477-491, 1991.
[4] L. F. Shampine and S.Thompson, "Solving DDEs in MATLAB," *Applied Numer. Math*., vol. 37, pp. 441-458, 2001.
[5] Z. Jackiewicz and E. Lo, "Numerical solution of neutral functional differential equations by Adams methods in divided difference form," *J. of Comp. and Appl. Math*.,vol. 189, pp.592-605, 2006.
[6] M. B. Suleiman and F.Ishak, "Numerical solution and stability of multistep method for solving delay differential equations (Accepted for publication)," *Japan J. Indust. Appl. Math*., published online 13 October 2010.
[7] L. G. Birta and O. Abou-Rabia, "Parallel block predictor-corrector methods for ODE's," *IEEE Trans. on Computer*, vol. c-36, pp. 299-311, 1987.
[8] A. Makroglou, "Block-by-block method for the numerical solution of Volterra delay integro-differential equations," *Computing*, vol. 30, pp. 49-62, 1983.
[9] L. F. Shampine and H. A. Watts, "Block implicit one-step methods," *Math. Comp*., vol. 23, pp. 731-170, 1969.
[10] B. P. Sommeijer, W. Couzy, and P. J. van der Houwen, "A-stable parallel block methods for ordinary and integro-differential equations," *Applied Numer. Math*., vol. 9, pp. 267-281, 1992.
[11] F. Ishak, M. Suleiman, and Z. Omar, "Two-point predictor-corrector block method for solving delay differential equations," *MATEMATIKA*, vol. 24, no. 2, pp. 131-140, 2008.
[12] V. K. Barwell, "Special stability problems for functional differential equations," *BIT*, vol. 15, pp. 130-135, 1975.
[13] L. F. Wiederholt, "Stability of multistep methods for delay differential equations," *Math. Comp*., vol. 30, pp. 283-290, 1976.
[14] A. N. Al-Mutib, "Stability properties of numerical methods for solving delay differential equations," *J. Comp. Appl. Math*., vol. 10, pp. 71-79, 1984.
[15] Z. Jackiewicz, "Asymptotic stability analysis of θ-methods for functional differential equations," *Numer. Math.*, vol. 43, pp. 389-396, 1984.