MATEMATIKA, 2008, Volume 24, Number 2, 131–140

Two-Point Predictor-Corrector Block Method for Solving Delay Differential Equations

[1]**Fuziyah Ishak,** [2]**Mohamed Suleiman &** [3]**Zurni Omar**

[1]Faculty of Information Technology & Quantitative Sciences
Universiti Teknologi MARA, 40450 Shah Alam, Selangor Darul Ehsan, Malaysia

[2]Mathematics Department, Faculty of Science and Environmental Studies
Universiti Putra Malaysia, 43400 Serdang, Selangor Darul Ehsan, Malaysia

[3]Mathematics Department, Faculty of Quantitative Sciences
Universiti Utara Malaysia, 06010 Sintok, Kedah Darul Aman, Malaysia

e-mail: [1]fuziyah_ishak@yahoo.com, [2]msuleiman@putra.upm.edu.my, [3]zurni@uum.edu.my

Abstract A 2-point variable order variable stepsize block method for the numerical solution of delay differential equations is described. The predictor-corrector formulae using divided difference formulation are implemented to produce two points simultaneously. This algorithm proves to be reliable, accurate and efficient. Furthermore, numerical experiments show that the block method reduces the number of total steps when compared with 1-point sequential method.

Keywords Delay differential equations; variable order; variable stepsize; block method.

1 Introduction

In this paper, we consider the numerical solution of systems of first order delay differential equations (DDEs) of the form:

$$\begin{aligned} &y'(x) = f(x, y(x), y(x-\tau_1), y(x-\tau_2), ..., y(x-\tau_n)),\ x \in [a,b],\ \tau_i > 0 \\ &y(x) = \phi(x),\ x \in [\alpha, a] \end{aligned} \tag{1}$$

where $\phi(x)$ is the initial function, $\tau_i,$ for $i = 1....n$, where n is an integer such that $n \geq 1$ is the lag function and

$$\alpha = \min_{x \in [a,b]} (x - \tau_i).$$

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 function f, where $f : [a,b] \times C([\alpha,b], R^m) \times C([\alpha,b], R^m) \rightarrow R^m$ is continuous and satisfies a Lipschitz condition which guarantees the existence of a unique solution of (1). Here, $C([\alpha,b], R^m)$ denotes the space of continuous functions mapping $[\alpha,b]$ into R^m for an integer $m \geq 1$. 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 Al-Mutib [1], Jackiewicz and Lo [5, 6], Jackiewicz [4] and Shampine and Thompson [10].

Most numerical methods for solving differential equations produce only one new approximation value at each step of integration. Block methods, in particular 2-point block methods, produce two new values simultaneously at each step by using the same back values. Each of these values is calculated independently of each other. This in turn will reduce the cost associated with the prediction, correction and function evaluation at each step as the number of total steps taken can be reduced. Block method can be implemented on sequential computers as well as parallel counterparts. The performance of the block method is further capitalize by implementing concurrent calculation on parallel computers as this will reduce the timing of the calculation as compared with the non-block method performing on sequential machines. Block methods have been proposed by some authors including Shampine and Watts [11], Sommeijer et al. [12], Birta and Abou-Rabia [2] and Makrouglou [8]. These methods were developed to solve either ODEs or Volterra integro-differential equations.

The purpose of this paper is to develop a variable order variable stepsize two-point predictor-corrector block method for solving equation (1). The algorithm will approximate two new values simultaneously at each step of integration. The formulae will be derived in such a way that the calculation of these new values can be performed independently of each other. It is of the authors' further intention that the method will be implemented on parallel machines as better performance can be expected by capitalizing on the capabilities of parallel processors. The parallelization technique using across the method approach will be chosen in order to perform the concurrent evaluations for the two points in the block. The efficiency of the method in terms of the total number of steps will be compared with the performance of the previous non-block algorithm developed by Ishak et al. [3]. The organization of this paper is as follows. In section 2, we derive the divided difference representation of the predictor corrector method for DDEs. In section 3, the calculation of the local error and the stepsize and order changing strategy are discussed. Section 4 presents some numerical results regarding the implementation of the method. Finally, section 5 is the conclusion.

2 Divided-Difference Representation of the Predictor and the Corrector

For reason of simplicity, we consider only single-delay scalar equation. However, the results are also valid for systems of equations with multiple delays. Consider delay differential equation of the form

$$\begin{cases} y'(x) = f(x, y(x), y(x-\tau)), \ x \in [a,b], \tau > 0 \\ y(x) = \phi(x), \ x \in [\alpha, a] \end{cases} \tag{2}$$

where ϕ is a continuous initial function. The non-uniform grid is given by

$$\alpha \leq a = x_0 < \cdots < x_n < x_{n+1} \cdots < x_N = b.$$

We denote the approximation to $y(x)$, where y is the solution of (2) by $y_h(x)$. Also denote the expression $f(x_n, y_h(x_n), y_h(x_n - \tau))$ by f_n. The predicted value of $y_h(x)$ is denoted as $p_h(x)$. Assume that y_h is already computed for $x \in [\alpha, x_n]$.

The immediate task is to evaluate $y_h(x_{n+1})$ and $y_h(x_{n+2})$ simultaneously. The Adams methods are 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. The predictor uses the Adams-Bashforth method of order k while the corrector uses the Adams-Moulton method of order $k+1$.

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

$$x_n, x_{n-1}, \ldots, x_{n-k+1}.$$

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

$$\begin{aligned} P_{k,n}(x) = f[x_n] + (x-x_n)f[x_n, x_{n-1}] + (x-x_n)(x-x_{n-1})f[x_n, x_{n-1}, x_{n-2}] \\ + \cdots + (x-x_n)(x-x_{n-1})\cdots(x-x_{n-k+2})f[x_n, x_{n-1}, \ldots, x_{n-k+1}] \end{aligned}$$

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

$$\begin{aligned} p_h(x_{n+r}) &= y_h(x_n) + \int_{x_n}^{x_{n+r}} P_{k,n}(\xi)\, d\xi, \\ &= y_h(x_n) + \sum_{i=0}^{k-1} f[x_n, x_{n-1}, \ldots, x_{n-i}] \int_{x_n}^{x_{n+r}} p_{n,i}(\xi)\, d\xi \end{aligned}$$

where ξ is an independent dummy variable and

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

The predicted values of the derivative are given by

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

For the corrector values, consider the interpolating polynomial $P_{k+1,n+r}$ of degree k that interpolates f at points, $(x_{n+r}, f^p_{n+r}), (x_n, f_n), (x_{n-1}, f_{n-1}), \ldots, (x_{n-k+1}, f_{n-k+1})$ where

$$f^p_{n+r} = f(x_{n+r}, p_h(x_{n+r}), p_h(x_{n+r} - \tau)).$$

The notation $f^p[x_{n+r}, x_n, \ldots, x_{n+k-1}]$ is referred to the polynomial $P_{k+1,n+r}$ that interpolates f^p_{n+r}. The interpolating polynomial $P_{k+1,n+r}$ can be written as

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

Replacing f in (2) with the polynomial $P_{k+1,n+r}$ and integrating, the corrector formulae for the two points are obtained as follows,

$$y_h(x_{n+r}) = p_h(x_{n+r}) + f^p[x_{n+r}, x_n, \ldots, x_{n-k+1}] \int_{x_n}^{x_{n+r}} p_{n,k}(\xi)\, d\xi$$

and

$$y'_h(x_{n+r}) = p'_h(x_{n+r}) + p_{n,k}(x_{n+r}) f^p[x_{n+r}, x_n, \ldots, x_{n-k+1}].$$

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

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

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

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

With the adoption of the following notation,

$$^{[0]}y_h(x) = p_h(x) \text{ and } ^{[0]}z_h(x) = {}^{[0]}y'_h(x) = p'_h(x),$$

the PECE mode based on Adams methods for the numerical solution of equation (2) is defined by

$$\begin{aligned}
P: &\begin{cases} {}^{[0]}y_h(x_{n+r}) = y_h(x_n) + \sum_{i=0}^{k-1} g_{i,1}(x_{n+r}) f[x_n, x_{n-1}, \ldots, x_{n-i}], \\ {}^{[0]}z_h(x_{n+r}) = \sum_{i=0}^{k-1} g_{i,0}(x_{n+r}) f[x_n, x_{n-1}, \ldots, x_{n-i}], \end{cases} \\
E: &\ {}^{[1]}z_h(x_{n+r}) = {}^{[1]}f(x_{n+r}, {}^{[0]}y_h(x_{n+r}), {}^{[0]}y_h(\alpha_r)), \\
C: &\begin{cases} {}^{[1]}y_h(x_{n+r}) = {}^{[0]}y_h(x_{n+r}) + g_{k,1}(x_{n+r}) f^p[x_{n+r}, x_n, \ldots, x_{n-k+1}] \\ {}^{[1]}z_h(x_{n+r}) = {}^{[0]}z_h(x_{n+r}) + g_{k,0}(x_{n+r}) f^p[x_{n+r}, x_n, \ldots, x_{n-k+1}] \end{cases} \\
E: &\ {}^{[2]}z_h(x_{n+r}) = {}^{[2]}f(x_{n+r}, {}^{[1]}y_h(x_{n+r}), {}^{[1]}y_h(\alpha_r)),
\end{aligned}$$

in which the numerical solution is defined by

$$y_h(x_{n+r}) = {}^{[1]}y_h(x_{n+r}).$$

Let e_r be defined as the difference between subsequent iteration values of the derivative at x_{n+r}. That is,

$$e_r = {}^{[1]}z_h(x_{n+r}) - {}^{[0]}z_h(x_{n+r}).$$

Therefore,

$$e_r = g_{k,0}(x_{n+r}) f^p[x_{n+r}, x_n, \ldots, x_{n-k+1}] \tag{3}$$

From equation (3), the corrector formulae can be rewritten as

$$C: \begin{cases} {}^{[1]}y_h(x_{n+r}) = {}^{[0]}y_h(x_{n+r}) + g_{k,1}(x_{n+r}) \dfrac{e_r}{g_{k,0}(x_{n+r})} \\ {}^{[1]}z_h(x_{n+r}) = {}^{[0]}z_h(x_{n+r}) + g_{k,0}(x_{n+r}) \dfrac{e_r}{g_{k,0}(x_{n+r})} \end{cases} \tag{4}$$

Let $\alpha_r = x_{n+r} - \tau_{n+r},\ r = 1, 2$ be the delay argument. The solution of the delay term is denoted by $y_h(\alpha_r)$. We now describe how the prediction and the correction of the delay term ${}^{[s]}y_h(\alpha_r),\ s = 0, 1$ are being carried out. Also note that $y_h(\alpha_r) = {}^{[1]}y_h(\alpha_r)$. The locations of α_r are sought because the calculation of the delay term depends upon these

locations. The derivation of the predictor-corrector formulae follows from earlier discussion. The calculation is done as follows.

$$ {}^{[0]}y_h(\alpha_r) = \begin{cases} \phi(\alpha_r), \ \alpha_r \le a \\ y_h(x_j) + \sum\limits_{i=0}^{k_{j+1}-1} g_{i,1}(\alpha_r) f[x_j, x_{j-1}, \ldots, x_{j-i}], \ a < \alpha_r \le x_n \\ y_h(x_n) + \sum\limits_{i=0}^{k-1} g_{i,1}(\alpha_r) f[x_n, x_{n-1}, \ldots, x_{n-i}], \ x_n < \alpha_r \le x_{n+r} \end{cases} $$

$$ {}^{[1]}y_h(\alpha_r) = \begin{cases} \phi(\alpha_r), \ \alpha_r \le a \\ {}^{[0]}y_h(\alpha_r) + g_{k_{j+1},1}(\alpha_r) f[x_{j+1}, x_j, \ldots, x_{j-k_{j+1}-1}], \ a < \alpha_r \le x_n \\ {}^{[0]}y_h(\alpha_r) + g_{k,1}(\alpha_r) \dfrac{e_r}{g_{k,0}(x_{n+r})}, \ x_n < \alpha_r \le x_{n+r}. \end{cases} $$

Here j and k_j are integers such that $x_j < \alpha_j \le x_{j+1}$ and k_{j+1} is the order of the method used for advancing the solution from x_j to x_{j+1}.

3 Local Error Estimation and the Strategy of Changing the Order and the Stepsize

A reliable way to control the local error at the grid points is to compare the formulae of different orders. Following the discussion in Suleiman [13], the local error at each grid point $E_{r,k}$ can be estimated as

$$ E_{r,k} \approx y_{n+r}(k+1) - y_{n+r}(k) $$

where $y_{n+r}(k)$ and $y_{n+r}(k+1)$ are the results of $y_h(x_{n+r})$ by using the iterative mode $\mathrm{PEC}_k\mathrm{E}$ and $\mathrm{PEC}_{k+1}\mathrm{E}$ respectively. Let

$$ E_{r,k} = y_{n+r}(k+1) - y_{n+r}(k) \tag{5} $$

denotes the estimated error in $y_{n+r}(k)$ at x_{n+r}. It follows from (4) that

$$ \begin{aligned} y_{n+r}(k) &= p_h(x_{n+r}) + \frac{g_{k-1,1}(x_{n+r})}{g_{k-1,0}(x_{n+r})} \bar{e}_r \\ y_{n+r}(k+1) &= p_h(x_{n+r}) + \frac{g_{k,1}(x_{n+r})}{g_{k,0}(x_{n+r})} \hat{e}_r \end{aligned} $$

where

$$ \bar{e}_r = y'_{n+r}(k) - p'_h(x_{n+r}) $$

and

$$ \hat{e}_r = y'_{n+r}(k+1) - p'_h(x_{n+r}). $$

Now,

$$ \begin{aligned} \bar{e}_r &= g_{k-1,0}(x_{n+r})(x_{n+r} - x_{n-k+1}) f[x_{n+r}, x_n, \ldots, x_{n-k+1}] \\ &= g_{k,0}(x_{n+r}) f[x_{n+r}, x_n, \ldots, x_{n-k+1}] \\ &= \hat{e}_r \\ &= e_r \end{aligned} \tag{6} $$

Using equation (6), we can rewrite (5) as

$$E_{r,k} = -\frac{g_{k-1,2}(x_{n+r})}{g_{k,0}(x_{n+r})} e_r. \tag{7}$$

It is expected that the magnitude of local error at the second point,$E_{2,k}$ is bigger than its counterpart at the first point. This is due to the fact that the prediction of the second point uses the stepsize of $2h$ where $h = x_{n+1} - x_n = x_{n+2} - x_{n+1}$, while the prediction of the first point uses a smaller stepsize h. Therefore, the local error at the second point is controlled. Numerical results also prove that this criterion gives a reliable decision in accepting a step or not.

This variable order variable stepsize algorithm offers a self-starting procedure by starting with the method of order one. The most optimal stepsize is taken while varying the order to achieve the required accuracy as efficiently as possible. Derivative discontinuities in the solutions of DDEs can be detected whenever the magnitude of the local error at the second point is too big as compared with a given tolerance. The algorithm reduces the stepsize and order accordingly in order to reduce the error and thus automatically includes the discontinuity point as the grid point. Errors of adjacent orders $E_{2,k-2}$, $E_{2,k-1}$ and $E_{2,k+1}$ can be calculated directly from formula (7) by exchanging k with $k-2$,$k-1$ and $k+1$respectively. These errors are needed in determining the order for the next integration step. The order and the stepsize changing mechanisms are similar as performed in Ishak et al [3], Omar [9] and Suleiman [13], with the exception that the local error is controlled at the second point.

4 Numerical Results

The accuracy and efficiency of the algorithm are demonstrated by the following examples. Each of the examples is solved on a given interval with the requested tolerance. Since it is impractical to present the numerical solutions at each of the grid points, we present the results in tables consisting of errors and total steps taken as a measurement for accuracy and efficiency of the method respectively. The numerical results are compared with the non-block variable order variable stepsize algorithm introduced in Ishak et al [3].

Example 1. (Al-Mutib [1])

$$\begin{aligned} y'(x) &= \cos(x)\, y\,(y(x) - 2),\ 0 \le x \le 50, \\ y(x) &= 1,\ x \le 0. \end{aligned}$$

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

The error, e_i at the grid point is defined as

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

where $y(x_i)$ 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 use 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). For a specified tolerance, we calculate the maximum and average errors.

The results of the errors together with the number of steps taken to perform the integration in the interval [0, 50] are given in Table 1.

Example 2. (Lin et al. [7])

$$\begin{cases} y'(x) = J_n \mathrm{y}(x) + D_n \mathrm{y}(x-1) + \mathrm{f}(x), & x \geq 0 \\ \mathrm{y}(x) = (1, 1, \ldots, 1)^T, & x \leq 0 \end{cases}$$

where

$$\mathrm{f}(x) = (\sin x, \sin 2x, \ldots, \sin nx)^T,$$

$$J_n = \begin{bmatrix} -1 & 2 & 1 & & \\ 2 & \ddots & \ddots & \ddots & \\ 1 & \ddots & \ddots & \ddots & 1 \\ & \ddots & \ddots & \ddots & 2 \\ & & 1 & 2 & -1 \end{bmatrix} \text{ and } D_n = \begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix}$$

with J_n and D_n are $n \times n$ matrices. In this example, n is taken to be 150 and the problem is solved in the interval of [0, 5].

Exact solution is not available. In this case, the reference solution at the end point for each equation in the system is produced by the method using the tolerance of 10^{-12}. The difference between the reference solution and the approximate solution is used in the calculation of maximum and average errors. The results are given in Table 2.

The following abbreviations are used in describing the solutions:

S1P	1-point variable order variable stepsize algorithm in Ishak et al. [3]
S2PB	2-point predictor-corrector block method
TOL	the chosen tolerance
MAXE	the maximum error
AVERR	the average error
FSTEP	the number of failed steps
STEP	the total number of steps

When the exact solution is available, the errors calculated are defined as

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

where $(y)_t$ is the t-th component of y. A and B may take the values of either 1 or 0 depending upon the kind of error test chosen. In this case, we use 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 and average errors are defined as follows,

$$\mathrm{MAXE} = \max_{1 \leq i \leq \mathrm{SSTEP}} \left(\max_{1 \leq t \leq N} (e_i)_t \right),$$

$$\mathrm{AVE} = \frac{\sum_{i=1}^{\mathrm{SSTEP}} \sum_{t=1}^{N} (e_i)_t}{(R)(N)(\mathrm{SSTEP})}$$

where N is the number of equations in the system, SSTEP is the total number of successful steps and R is the number of point.

When the exact solution is not available, the error calculated refers to the error at the endpoint. The reference solution at the end point for each equation in the system is produced by the method using the tolerance of 10^{-12}. The experiment was executed on Sunfire V1280 by using one processor. The machine is located at the Institute of Mathematical Research (INSPEM), Universiti Putra Malaysia. The numerical results for different values of the tolerance are given in Table 1 and Table 2 for Example 1 and Example 2 respectively.

Table 1: Numerical Results for Example 1

TOL	METHOD	STEP	FSTEP	AVERR	MAXE
10^{-2}	S1P	78	4	1.33840E-02	4.26458E-02
	S2PB	62	4	7.89873E-03	2.54222E-02
10^{-4}	S1P	118	3	8.78369E-05	2.75513E-04
	S2PB	88	3	6.39995E-05	1.81656E-04
10^{-6}	S1P	161	1	1.55361E-06	3.47645E-06
	S2PB	126	5	4.99711E-07	1.48025E-06
10^{-8}	S1P	219	1	4.17149E-09	1.23526E-08
	S2PB	148	0	2.87762E-10	1.04934E-09
10^{-10}	S1P	273	0	5.07972E-11	1.14012E-10
	S2PB	214	7	2.43087E-11	5.99198E-11

Table 2: Numerical Results for Example 2

TOL	METHOD	STEP	FSTEP	AVERR	MAXE
10^{-2}	S1P	101	1	5.34740E-03	1.06864E-02
	S2PB	125	1	4.38894E-02	5.48535E-02
10^{-4}	S1P	269	0	2.84992E-05	5.99010E-05
	S2PB	262	6	8.15218E-05	1.53040E-04
10^{-6}	S1P	555	0	2.75874E-07	7.70910E-07
	S2PB	439	1	7.13090E-07	2.71596E-06
10^{-8}	S1P	936	1	1.48249E-09	8.41883E-09
	S2PB	704	2	1.59668E-08	6.16016E-08
10^{-10}	S1P	1435	3	4.94689E-12	2.01822E-11
	S2PB	1053	4	9.86221E-11	6.95126E-10

For Example 1, the average and maximum errors of S2PB are slightly better than S1P. For Example 2, the errors of S1P are slightly better than that of S2PB. However, both of the methods achieved the desired accuracy and it is clear that the errors are comparable for the given tolerances in both of the methods. The total number of steps for S2PB is less than the total number of steps for S1P, except for tolerance 10^{-2} in Example 2 which is slightly higher. For finer tolerances, S2PB shows greater reduction in the total number

of steps. The number of failed steps in both methods for Example 1 and Example 2 are comparable.

The numerical results clearly indicate that the 2-point predictor-corrector block method performs very well and achieves the required accuracy as compared to the 1-point variable order variable stepsize algorithm. The block method is more efficient as it reduces the number of total steps performed.

5 Conclusion

We have presented a variable order variable stepsize two point block method for the numerical solution of delay differential equations. The performance of the method is compared with the non-block one point algorithm. The method produces two new values simultaneously, thus reduces the cost as the total step decreases. From the numerical results we conclude that this algorithm 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] L. G. Birta & O. Abou-Rabia, *Parallel Block Predictor-Corrector Methods for ODE's,* IEEE Trans. on Computer, C-36 (March 1987), 299-311.

[3] F. Ishak, M. B. Suleiman, F. Ismail & Z. A. Majid, *Variable order variable stepsize algorithm for the numerical solution of delay differential equations in divided difference form,* Proceedings of The 2^{nd} IMT-GT 2006 Regional Conference on Mathematics, Statistics and Applications Vol. II (2006), 46-54.

[4] Z. Jackiewicz, *Variable-Step Variable-Order Algorithm for the Numerical Solution of Neutral Functional Differential Equations,* Applied Numer. Math., 3(1987), 317-329.

[5] Z. Jackiewicz & E. Lo, *The Numerical Solution of Neutral Functional Differential Equations by Adams Predictor-Corrector Methods,* Applied Numer. Math., 8(1991), 477-491.

[6] Z. Jackiewicz & E. Lo, *Numerical Solution of Neutral Functional Differential Equations by Adams Methods in Divided Difference Form,* Journal of Comp. and Appl. Math. 189(2006), 592-605.

[7] F. R. Lin, X. Q. Jin & S. L. Lei, *Strang-Type Preconditioners for Solving Linear Systems from Delay Differential Equations,* BIT Numerical Mathematics, 43(2003), 139-152.

[8] A. Makroglou, *A Block-by-Block Method for the Numerical Solution of Volterra Delay Integro-Differential Equations,* Computing, 30(1983), 49-62.

[9] Z. Omar, *Parallel Block Methods for Solving Higher Order Ordinary Differential Equations Directly,* PhD Thesis, University Putra Malaysia, 1999.

[10] L. F. Shampine & S. Thompson, *Solving DDEs in MATLAB,* Applied Numerical Mathematics, 37(2001), 441-458.

[11] L. F. Shampine & H. A. Watts, *Block Implicit One-Step Methods,* Math. Comp., 23 (1969), 731-170.

[12] B. P. Sommeijer, W. Couzy & P. J. van der Houwen, *A-stable parallel block methods for ordinary and integro-differential equations,* Applied Numerical Mathematics, 9(1992), 267-281.

[13] M. B. Suleiman, *Generalized Multistep Adams and Backward Differentiation Methods for the Solution of Stiff and Non-Stiff Ordinary Differential Equations,* PhD Thesis, University of Manchester, 1979.