Chiang Mai J. Sci. 2012; 39(3) : 502-510
http://it.science.cmu.ac.th/ejournal/
Contributed Paper

2-Point Block Predictor-Corrector of Backward Differentiation Formulas for Solving Second Order Ordinary Differential Equations Directly

Zarina Bibi Ibrahim*[a], Khairil Iskandar Othman [b] and Mohamed Suleiman [c]

[a] Institute for Mathematical Research, Department of Mathematics, Faculty of Science, Universiti Putra Malaysia, 43400 UPM Serdang, Selangor, Malaysia.

[b] Department of Mathematics, Faculty of Computer and Mathematical Sciences, Universiti Technology MARA, 40450 Shah Alam, Selangor, Malaysia.

*Author for correspondence; e-mail: zarinabb@science.upm.edu.my

Received : 28 February 2011
Accepted : 10 January 2012

ABSTRACT

A code based on predictor - corrector 2-point block of Backward Differentiation Formulas using variable step size is developed for solving second order Ordinary Differential Equations (ODEs). The new method solved the second order differential equations directly without reduction to its first order differential equations. Further, the new method allows the differentiation coefficients to be stored and thus avoiding the coefficients to be calculated at each step but robust enough to allow for step size variation. Computational advantages are presented comparing the results obtained by the new method with solver ode15s and ode23s in MATLAB. Numerical results prove that the new developed method is reliable and efficient.

Keywords: block method, directly, ordinary differential equations

1. INTRODUCTION

In this paper, a 2-point block of Backward Differentiation Formulas (BBDF2) is developed for solvingstiff initial value problems (IVPs) for second order Ordinary Differential Equations(ODEs) of the form

$$\begin{cases} y'' = f(t, y, y'), t \in [t_0, t_{end}] \\ y(t_0) = y_0, y'(t_0) = y'_0 \end{cases} \quad (1)$$

Such systems often occur in mechanical systems without dissipation, satellite tracking and celestial mechanics [1]. Many problems of the form (1) are solved by reducing it to an equivalent system of first order equations and then solved it using first order method. These reductions to first order methods involve overhead which are computationally expensive in terms of execution times and total number of steps used. Block method for numerical solution of (1) directly, had been proposed by several researchers such as [1,2, 4-6]. A 2-point and 3-point explicit block methods for solving higher order ODEs directly are described in [6]. Two point block method designed for two processors for solving directly non stiff large system of higher order ODEs has been developed by [5].

The general theory of block method for solving (1) is given in [1]. A block method will compute a set of new approximations simultaneously at several distinct points on the *x*-axis in the block. Though there has been much effort and research on solving second order differential equations of the form (1) directly, not much attention is given on solving stiff second order differential equations directly. Most of the work is focused on solving nonstiff second order differential equations. Hence, the purpose of this paper is to develop a variable step size block differentiation method known as 2-point block of Backward Differentiation Formulas (BBDF2) method for solving Eq. (1) which are stiff *i.e.* equations with widely separated time constants. These type of equations are encountered in many areas of applied mathematics *e.g.* in reactor calculations, circuit analysis and chemical kinetics. The problem in (1) is said to be stiff whenever the eigen values λ_i are such that $Re\,\lambda_i < 0$ and $\max_i|\text{Re}\,\lambda_i| \gg \min_i|\text{Re}\,\lambda_i|$.

When solving (1) numerically, the focus will be on achieving high accuracy with reduction in the computational cost. These are done by storing all the differentiation coefficients in order to prevent repetitive computations of the differentiation coefficients, apart from the advantage of the direct method in not enlarging the second order system. In the following section, the general theory on backward differentiation formulas is summarized briefly. Section 3 discussed the computations of the coefficients of the new method and the implementation of the method using Newton Iteration. Finally, in Section 4, numerical tests on second order differential equations which are stiff are performed.

2. General theory on backward differentiation formulas

The Backward Differentiation Formula (BDF) is an implicit multistep method which is based on interpolating the solution points y_{n+1-i} rather than the derivatives $y'_{n+1-i} = f(t_{n+1-i}, y_{n+1-i})$ where $i = 1,\ldots, k$.

The BDF formula is obtained by considering the *k*th degree polynomial P_{k+1} which satisfiesthe conditions:

$$P_{k+1}(t_{n+1-i}) = y_{n+1-i} \tag{2a}$$

$$P'_{k+1}(t_{n+1}) = f(t_{n+1}, y_{n+1}) \tag{2b}$$

Equation (2b) can be expressed by

$$y_{n+1} = -\sum_{i=1}^{k} \alpha^c_{k-i}\, y_{n+1-i} + h_n \beta^c_k f(t_{n+1}, y_{n+1}) \tag{3}$$

which is known as the corrector BDF with order *k*. α_k and Equation (3) is explicit in the solution y_{n-k} since it uses only the past values $y_n, y_{n-1}, y_{n-2},\ldots, y_{n-k}$, but is implicit in the one derivative, dy_{n+1}/dt. The predictor–corrector scheme for a BDF method has the form:

Predict (P): $y^p_{n+1} = -\sum_{i=1}^{k} \alpha^p_{k-i}\, y_{n+1-i}$

Evaluate (E): $y'_{n+1} = f\left(t_{n+1}, y^p_{n+1}\right)$

Correct (C): $= y_{n+1} y^p_{n+1} - \sum_{i=1}^{k} \bar{\alpha}^c_{k-i}\, y_{n+1-i} + h_n \beta^c_k y'_{n+1}$

Evaluate (E): $y'_{n+1} = f(t_{n+1}, y_{n+1})$

with $\bar{\alpha} = \alpha^c - \alpha^p$. The global error of the method at a given time t_n

$$e_n = y(t_n) - y_n$$

3. Formulation of 2-Point Block Backward Differentiation Formulas (BBDF2)

In this section, we shall derive the coefficients for the BBDF2 method to approximate the values for y_{n+1} and y_{n+2} at step t_{n+1} and t_{n+2} simultaneously. We use y_n to denote the generated approximation to the solution $y(t)$ at $t = t_n$ and the evaluation of f at (t_n, y_n), is denoted by $f(t_n, y_n)$. The derivation of the coefficients for the Direct Differentiation method is based on the derivation introduced in [8,9].

(i) The interval $[a,b]$ is divided into series of blocks with each block containing two points. In Figure 1, two new values of y_{n+1} and y_{n+2} are computed simultaneously in a block using the same back values.

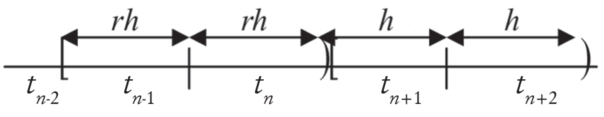

Figure 1. 2-point block method.

From Figure 1, the computed block has step size h and the previous block has step size rh where r is the step size ratio for variable step size implementation.

(ii) For the BBDF2, the values considered were $r = 1$, $r = 2$, and $r = 5/8$ which corresponds respectively to constant step size, half the step size and increment of the step size by a factor of 1.6, i.e. $r = 5/8$. We do not consider doubling the step size, *i.e.* $r = 0.5$ due to zero instability.

(iii) For the corrector formulas, consider the Lagrange polynomial corresponding to the set of interpolation points $\{t_{n-2},\ldots,t_{n+2}\}$ with $s = t - t_{n+1}/h$ or $t = t_{n+1} + sh$. Let $P_{k,n}$ be the interpolating polynomial of degree $k\text{-}1$ interpolating f at points (t_n, f_n), (t_{n-1}, f_{n-1}),..., (t_{n-k+1}, f_{n-k+1}). The formula is obtained by replacing f in (1) with the polynomial $P_{k+1,n+1}$.

(iv) The polynomial is then differentiated twice at $t = t_{n+1}$. The coefficients for y_{n+1} are obtained by substituting $s = 0$.

(v) Similarly, the coefficients for the second point, y_{n+2} is obtained by differentiating twice the resulting Lagrange polynomial at $t = t_{n+2}$ and substituting $s = 1$ with various step ratio $r = 1, 2,$ and $r = 5/8$.

The corresponding pairs of formulas of BBDF2 with constant step size ($r = 1$), halved the step size ($r = 2$) and increasing the step size by 1.6 h are as follows:

for $r = 1$

$$\left.\begin{aligned}
hy'_{n+1} &= -\frac{1}{12}y_{n-2} + \frac{1}{2}y_{n-1} - \frac{3}{2}y_n + \frac{5}{6}y_{n+1} + \frac{1}{4}y_{n+2} \\
hy'_{n+2} &= \frac{1}{4}y_{n-2} - \frac{4}{3}y_{n-1} + 3y_n - 4y_{n+1} + \frac{25}{12}y_{n+2} \\
y_{n+1} &= -\frac{1}{20}y_{n-2} + \frac{1}{5}y_{n-1} + \frac{3}{10}y_n + \frac{11}{20}y_{n+2} - \frac{3}{5}h^2 f_{n+1} \\
y_{n+2} &= -\frac{11}{35}y_{n-2} + \frac{8}{5}y_{n-1} - \frac{114}{35}y_n + \frac{104}{35}y_{n+2} + \frac{12}{35}h^2 f_{n+2}
\end{aligned}\right\} \qquad (3)$$

for $r = 2$

$$\left.\begin{aligned}
hy'_{n+1} &= -\frac{1}{80}y_{n-2} + \frac{5}{48}y_{n-1} - \frac{15}{16}y_n + \frac{8}{15}y_{n+1} + \frac{5}{16}y_{n+2} \\
hy'_{n+2} &= -\frac{1}{30}y_{n-2} - \frac{1}{4}y_{n-1} + \frac{3}{2}y_n - \frac{16}{5}y_{n+1} + \frac{23}{12}y_{n+2} \\
y_{n+1} &= -\frac{1}{224}y_{n-2} + \frac{5}{224}y_{n-1} + \frac{15}{32}y_n + \frac{115}{224}y_{n+2} - \frac{15}{28}h^2 f_{n+1} \\
y_{n+2} &= -\frac{1}{20}y_{n-2} + \frac{5}{14}y_{n-1} - \frac{51}{28}y_n + \frac{88}{35}y_{n+2} + \frac{3}{7}h^2 f_{n+2}
\end{aligned}\right\} \qquad (4)$$

for $r = 5/8$

$$\left.\begin{aligned}
hy'_{n+1} &= -\frac{208}{825}y_{n-2} + \frac{3072}{2275}y_{n-1} - \frac{117}{50}y_n + \frac{279}{52}y_{n+1} + \frac{3}{14}y_{n+2} \\
hy'_{n+2} &= -\frac{224}{275}y_{n-2} + \frac{2048}{525}y_{n-1} + \frac{273}{50}y_n - \frac{14}{3}y_{n+1} + \frac{3}{14}y_{n+2} \\
y_{n+1} &= -\frac{512}{2125}y_{n-2} + \frac{12288}{14875}y_{n-1} - \frac{819}{4250}y_n + \frac{723}{1190}y_{n+2} - \frac{117}{170}h^2 f_{n+1} \\
y_{n+2} &= -\frac{70784}{67575}y_{n-2} + \frac{96256}{22525}y_{n-1} - \frac{125853}{22525}y_n + \frac{9086}{2703}y_{n+2} + \frac{273}{901}h^2 f_{n+2}
\end{aligned}\right\} \qquad (5)$$

Note that the approximation to y_{n+1} is computed using past values y_{n-2}, y_{n-1} and y_n as well as predicted value for f_{n+1} and future value y_{n+2}. For this reason, we say that the formulas are fully implicit. Another characteristic feature of the BBDF2 is that each application of the formulas generates a block of *m* new solution values along the *t* axis on each step and hence each block is considered as a unit calculation. The general *k*-order BBDF2 for solving problem (1) in general form is

$$\sum_{j=0}^{k} \alpha_j y_{n+j} = h\sum_{j=0}^{k} \beta_j y'_{n+j} + h^2 \sum_{j=0}^{k} \gamma_j f_{n+j}$$

where $\alpha_i, \beta_i, \gamma_i$ are the coefficients of the method in (3), (4) and (5).

3.1 Implementation of the Method

In this section, we derive the iteration process by applying Newton-type iteration to the direct differentiation methods. The notation i is introduced for specifying the iteration. Therefore, $y_{n+1}^{(i+1)}$ will denote the i^t iterative value of y_{n+1} and $e_{n+1}^{(i+1)} = y_{n+1}^{(i+1)} - y_{n+1}^{(i)}$ will denote the difference between the ith and $(i+1)$th iterative values of y_{n+1}.

First point formula in equation (3) can be written as

$$hy'^{(i)}_{n+1} = \beta_1 y_{n+1}^{(i)} + \beta_2 y_{n+2}^{(i)} + G_{n1} \qquad (6)$$

$$hy'^{(i+1)}_{n+1} = \beta_1 y_{n+1}^{(i+1)} + \beta_2 y_{n+2}^{(i+1)} + G_{n1} \qquad (7)$$

where G_{n1} are the back values.

Subtract the equation in (6) from (7), we have

$$he'^{(i+1)}_{n+1} = \beta_1 e_{n+1}^{(i+1)} + \beta_2 e_{n+2}^{(i+1)} \qquad (8)$$

Similar process apply to the second point formula in equation (3) which give the following

$$he'^{(i+1)}_{n+1} = \hat{\beta}_1 e_{n+1}^{(i+1)} + \hat{\beta}_2 e_{n+2}^{(i+1)} \qquad (9)$$

(8) and (9) can be written in matrix for

$$\hat{E}' = \begin{bmatrix} e'_{n+1} \\ e'_{n+2} \end{bmatrix}^{(i+1)} = \frac{1}{h}\begin{bmatrix} \beta_1 & \beta_2 \\ \hat{\beta}_1 & \hat{\beta}_2 \end{bmatrix}\begin{bmatrix} e_{n+1} \\ e_{n+2} \end{bmatrix}^{(i+1)} \qquad (10)$$

Next, the Newton iteration for the third point in the formula are derived using the first principle, which is by Taylor expansion.

$$y_{n+1}^{(i+1)} = \theta_1 y_{n+2}^{(i+1)} + \alpha_1 h^2 f_{n+1}^{(i+1)} + D_{n1} \quad (11)$$

$$y_{n+1}^{(i+1)} = \theta_1 y_{n+2}^{(i+1)} + \alpha_1 h^2 f\left(y_{n+1}^{(i+1)}, {y'}_{n+1}^{(i+1)}\right) + D_{n1} (12)$$

Substitute $y_{n+1}^{(i+1)} = y_{n+1}^{(i)} + e_{n+1}^{(i+1)}$ and ${y'}_{n+1}^{(i+1)} = {y'}_{n+1}^{(i)} + {e'}_{n+1}^{(i+1)}$ into (12). Thus,

$$y_{n+1}^{(i)} + e_{n+1}^{(i+1)} = \theta_1\left(y_{n+2}^{(i)} + e_{n+2}^{(i+1)}\right) + \alpha_1 h^2 f\left(y_{n+1}^{(i)} + e_{n+1}^{(i+1)}, {y'}_{n+1}^{(i)} + {e'}_{n+1}^{(i+1)}\right) + D_{n1} (13)$$

Expanding $f\left(y_{n+1}^{(i)} + e_{n+1}^{(i+1)}, {y'}_{n+1}^{(i)} + {e'}_{n+1}^{(i+1)}\right)$ in (13) by Taylor expansion, will generate

$$y_{n+1}^{(i)} + e_{n+1}^{(i+1)} = \\ \theta_1\left(y_{n+2}^{(i)} + e_{n+2}^{(i+1)}\right) + \alpha_1 h^2\left[f\left(y_{n+1}^{(i)}, {y'}_{n+1}^{(i)}\right) + \left(\frac{\partial f_{n+1}}{\partial y_{n+1}}\right) e_{n+1}^{(i+1)} + \left(\frac{\partial f_{n+1}}{\partial {y'}_{n+1}}\right) {e'}_{n+1}^{(i+1)}\right] + D_{n1} \quad (14)$$

Substitute (8)into (14), and do some simplifications yield

$$\left[1 - \alpha_1 h^2\left(\frac{\partial f_{n+1}}{\partial y_{n+1}}\right) - \alpha_1\beta_1 h\left(\frac{\partial f_{n+1}}{\partial {y'}_{n+1}}\right)\right] e_{n+1}^{(i+1)} - \left[\theta_1 + \alpha_1\beta_2 h\left(\frac{\partial f_{n+1}}{\partial {y'}_{n+1}}\right)\right] e_{n+2}^{(i+1)} \\ = -y_{n+1}^{(i)} + \theta_1 y_{n+2}^{(i)} + \alpha_1 h^2 f\left(y_{n+1}^{(i)}, {y'}_{n+1}^{(i)}\right) + D_{n1} \qquad (15)$$

Similarly, Newton iteration was applied to the point of y_{n+2} and gives the following

$$-\left[\theta_2 + \alpha_2\hat{\beta}_1 h\left(\frac{\partial f_{n+2}}{\partial {y'}_{n+2}}\right)\right] e_{n+1}^{(i+1)} + \left[1 - \alpha_2 h^2\left(\frac{\partial f_{n+2}}{\partial y_{n+2}}\right) - \alpha_2\hat{\beta}_2 h\left(\frac{\partial f_{n+2}}{\partial {y'}_{n+2}}\right)\right] e_{n+2}^{(i+1)} \\ = -y_{n+2}^{(i)} + \theta_2 y_{n+1}^{(i)} + \alpha_2 h^2 f\left(y_{n+2}^{(i)}, {y'}_{n+2}^{(i)}\right) + D_{n2} (16)$$

The corresponding linear system to be solved in matrix form is $AE = B$. Generally,

$$A = \begin{bmatrix} 1 - \alpha_1 h^2\left(\frac{\partial f_{n+1}}{\partial y_{n+1}}\right) - \alpha_1\beta_1 h\left(\frac{\partial f_{n+1}}{\partial {y'}_{n+1}}\right) & -\theta_1 - \alpha_1\beta_2 h\left(\frac{\partial f_{n+1}}{\partial {y'}_{n+1}}\right) \\ -\theta_2 - \alpha_2\hat{\beta}_1 h\left(\frac{\partial f_{n+2}}{\partial {y'}_{n+2}}\right) & 1 - \alpha_2 h^2\left(\frac{\partial f_{n+2}}{\partial y_{n+2}}\right) - \alpha_2\hat{\beta}_2 h\left(\frac{\partial f_{n+2}}{\partial {y'}_{n+2}}\right) \end{bmatrix}$$

$$B = \begin{bmatrix} -y_{n+1}^{(i)} + \theta_1 y_{n+2}^{(i)} + \alpha_1 h^2 f\left(y_{n+1}^{(i)}, y_{n+1}^{\prime(i)}\right) + D_{n1} \\ \theta_2 y_{n+1}^{(i)} - y_{n+2}^{(i)} + \alpha_2 h^2 f\left(y_{n+2}^{(i)}, y_{n+2}^{\prime(i)}\right) + D_{n2} \end{bmatrix} \text{and}$$

$$E = \begin{bmatrix} e_{n+1} \\ e_{n+2} \end{bmatrix}^{(i+1)}$$

in which $\alpha_j, \theta_j, \beta_j, \overline{\beta_j}$ are the coefficients of the formulas in equation (3),(4), and (5).Calculate ${e'}_{n+1}^{(i+1)}$ and ${e'}_{n+2}^{(i+1)}$ from the value *E* as in equation (10).
Therefore $Y_{n+1,n+2}^{(i+1)} = Y_{n+1,n+2}^{(i)} + E_{n+1,n+2}^{(i+1)}$ and ${Y'}_{n+1,n+2}^{(i+1)} = Y_{n+1,n+2}^{\prime(i)} + {E'}_{n+1,n+2}^{(i+1)}$.

The following computations is carried out to obtain the approximations:

(i) Compute $Y^{(i)}_{n+1,n+2}$ and $Y'^{(i)}_{n+1,n+2}$ using the predictor formula.

(ii) Solved $E^{(i+1)}_{n+1,n+2}$ and followed by $E'^{(i+1)}_{n+1,n+2}$ for the corresponding linear system.

(iii) Computed the corrected value of $Y^{(i)}_{n+1,n+2}$ and $Y'^{(i)}_{n+1,n+2}$ with $E^{(i+1)}_{n+1,n+2}$ and $E'^{(i+1)}_{n+1,n+2}$ respectively.

3.2 Step Size Selection

In order to control the global error, a step size is selected in such a way that the global error increment is approximately of the size of a given tolerance TOL. On any given step, the user will imposed an error tolerance limit, TOL at each step. Following the discussion in [7], an estimation of local truncation error is obtained by comparing the derived corrector formula of order $(k+1)$ at second point, and the corrector formula of order k. Let LTE be the estimate of local truncation error at the second point in the block. Then, we have

$$\text{LTE} \simeq y^{(k+1)}_{n+2} - y^{(k)}_{n+2} \qquad (17)$$

where $y^{(k+1)}_{n+2}$ is the $(k+1)$th order method and $y^{(k)}_{n+2}$ is the kth order method. Let k_{old} and h_{new} denote the current step size and the next step size to be used respectively. The step size adjustment is controlled in the following way:

(i) To check the success of a step the estimated error is checked against a given tolerance. If $\|\text{LTE}\| < \text{TOL}$, the required accuracy has been obtained. Therefore, the approximate solution y_{n+1}, y_{n+2} at t_{n+1}, t_{n+2} is accepted and the step is considered successful. Then, the next step size is increased by 1.6 to gain computation speed, *i.e.* r is 5/8. The step size increment is given by

$$h_{new} = ch_{old}\,(\text{TOL/LTE})^{1/p+1}$$

and if $h_{new} \geq \sigma h_{old}$ then $h_{new} = \sigma h_{old}$ where c is the safety factor such that $0 < c < 1$, p is the order of the method, h_{old} is the step size from previous block and $\sigma = (\text{TOL/LTE})^{1/p+1}$. The safety factor c is to reduce the number of failure step.

(ii) If $\|\text{LTE}\| \geq \text{TOL}$, the step is considered fail since the required accuracy is not obtained. The approximate solution y_{n+1}, y_{n+2} at t_{n+1}, t_{n+2} is rejected and the step is repeated with decrease step size, in our code this is done by halving the current step size. In this case, the step ratio r is 2.

4. PROBLEM TESTED

In this section, we solved several model problems to validate the efficiency of the new direct differentiation method. The results are compared with the differential equations solver in Matlab. These codes are

ode15s:	Variable order solver based on Backward Differentiation Formulas, BDFs (Gear's method).
ode23s:	Based on a Modified Rosenbrock formula of order 2.

Problem: Harmonic oscillator in circuit theory (RLC circuits) and in physics of particles

$$my'' + K_d y' + K_s y = 0$$

m = mass, K_s = spring constant, K_d =

coefficient damping. Associated first order system is

$$y'_1 = y_2$$
$$y'_2 = -\frac{K_s}{m}y_1 - \frac{K_d}{m}y_2$$

The ode15s, ode23s and BBDF2 code were applied to the following test problems.

Problem 1:

$y'' = -10000y - 100y'$(18)

with initial values $y(0) = -3$, $y'(0) = 0$, and range of integration is $0 \le t \le 15$.

Exact solution: $y(t) = -e^{-50t}\left[3cos\left(50\sqrt{3}\right)t + \sqrt{3}sin\left(50\sqrt{3}\right)t\right]$

Equation (18) when reduced to first order ODEs: $y'_1 = y_2$

$y'_2 = -10000y_1 - 100y_2$

with initial values $y_1(0) = -3$, $y_2(0) = 0$, and range of integration is $0 \le t \le 15$.

Exact solution:

$y_1(t) = -e^{-50t}\left[3cos\left(50\sqrt{3}\right)t + \sqrt{3}sin\left(50\sqrt{3}\right)t\right]$

$y_2(t) = 200\sqrt{3}e^{-50t}sin\left(50\sqrt{3}\right)t$

Problem 2:

$y'' = -3y - 4y'$(19)

with initial values $y(0) = 2, y^{'(0)} = -12$ and range of integration is $0 \le t \le 15$.

Exact solution: $y(t) = -3e^{-t} + 5e^{-3t}$

Equation (19) when reduced to first order ODEs: $y_1' = y_2$

$y_2' = -3y_1 - 4y_2$

with initial values $y_1(0) = 2, y_2(0) = -12$

Exact solution: $y_1(t) = -3e^{-t} + 5e^{-3t}$

$y_2(t) = 3e^{-t} - 15e^{-3t}$

The test set was solved with tolerances 10^{-2}, 10^{-4} and 10^{-6}. The comparison between the direct method and reduction method in terms of maximum error and total number of integration steps are tabulated for the two problems in Table 1 and 2. The following notations are used in Table 1 and 2:

ode15s:	Approximate solution using variable order solver based on Backward Differentiation Formulas, BDFs (Gear's method).
ode23s:	Approximate solution based on a Modified Rosenbrock formula of order 2.
BBDF2:	Approximate solution using BBDF2
TOL:	Tolerance
TS:	Total number of steps to complete the integration in a given range.
MXE:	Magnitude of the maximum error of the computed solution.

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

$$(e_i)_n = \left|\frac{(y_i)_n - \left(y(t_i)\right)_n}{\omega + \mu\left(y(t_i)\right)_n}\right|$$

where
y_i : the approximate solution
$y(t_i)$: the exact solution at t_i
$\omega = 1, \mu = 0$: absolute error test
$\omega = 1, \mu = 1$: mixed error test
$\omega = 0, \mu = 1$: relative error test

The maximum error is defined as follows

$$\text{MAXE} = \max_{1 \le i \le SSTEP} (\max_{1 \le n \le N} (e_i)_n)$$

where N is the number of equations in the system and SSTEP is the total number of successful steps. The code is written in C programming language and executed on Intel Pentium IV processor.The tables below give the numerical results for problem 1 and 2.

Table 1. Comparison between ode15s, ode 23s and DBBDF methods for solving Problem 1.

TOL	Method	TS	MXE
10^{-2}	ode15s	82	3.3906E+00
	ode23s	49	2.3708E+00
	BBDF2	40	2.4753E-03
10^{-4}	ode15s	143	0.0512E+00
	ode23s	169	0.1183E+00
	BBDF2	79	1.6352E-04
10^{-6}	ode15s	253	4.6632E-04
	ode23s	720	0.0056E+00
	BBDF2	205	8.1226E-06
10^{-8}	ode15s	472	1.5202E-05
	ode23s	3266	2.5880E-04
	BBDF2	577	3.4128E-07

Table 2. Comparison between ode15s, ode 23s and DBBDF methods for solving Problem 2.

TOL	Method	TS	MXE
10^{-2}	ode15s	34	0.0601E+00
	ode23s	23	0.0305E+00
	BBDF2	27	2.97862E-03
10^{-4}	ode15s	72	0.0011E+00
	ode23s	75	0.0014E+00
	BBDF2	58	2.00190E-04
10^{-6}	ode15s	136	1.8709E-05
	ode23s	326	6.5386E-05
	BBDF2	152	6.99359E-06
10^{-8}	ode15s	256	2.1855E-07
	ode23s	1495	3.0307E-06
	BBDF2	421	2.50427E-07

The numerical results presented in Table 1-2 clearly indicate it is obvious that for all tolerance, the method BBDF2 required less number of total steps compared to ode15s and ode23s when solving the same problems.These results are expected since the BBDF2 method would approximate the solution at two points simultaneously. It is observed that the accuracy is either maintained or better when using BBDF2.

5. CONCLUSION

In this paper, we have developed a code of variable step size based on predictor - corrector 2-point block Backward Differentiation Formulas which can be applied to any stiff second order differential equations. The formulas representation reduces the computational cost since the coefficients of the methods need not be recomputed at every step when the stepsize varying. We have shown the efficiency of the developed direct block method in terms of total steps and improving the accuracy over the reduction to first order method for solving second order differential equations.

ACKNOWLEDGMENTS

The authors wish to thank the anonymous referees for their careful reading of the manuscript. This work is partially supported by Institute of Mathematical Research, Universiti Putra Malaysia under Fundamental Research Grant Scheme (project code 01-04-10-888FR).

REFERENCES

[1] Fatunla S.O., Block methods for second order ODEs, *Int. J. Comput. Math.*, 1991; **40**: 55-63.

[2] Ibrahim Z.B., Othman K.I. and Suleiman M.B., Implicit R-point block backward differentiation formula for solving first order stiff ODEs, *Int. J. Appl. Math. Comput.*, **186**: 558-565.

[3] Lambert J.D., Numerical Methods for Ordinary Differential Systems-The Initial Value Problem, 1991 (New York: Wiley).

[4] Majid Z. and Suleiman M.B., 1-point implicit code of Adams Moulton type to solve first order ordinary differential equations, *Chiang Mai J. Sci.*, 2006; **33(2)**: 153-159.

[5] Majid Z. and Suleiman M.B., Parallel direct integration variable step block method for solving large system of higher order ordinary differential equations, *Int. J. Comput. Math. Sci.*, 2009; **3(3)**: 119-123.

[6] Omar Z. and Suleiman M.B., Designing parallel algorithms for solving higher order ordinary differential equations directly on a shared memory parallel computer architecture, *Chiang Mai J. Sci.*, 2006; **33(1)**: 9-21.

[7] Senu N., Suleiman M. and Ismail F., An embedded explicit Runge-Kutta-Nystrom method for solving oscillatory problems, *PhysicaScripta*, 2009; **80**: 1-7.

[8] Suleiman M.B., Generalised Multistep Adams and Backward Differentiation Methods for the Solution of Stiff and Non-Stiff Ordinary Differential Equations. Ph.D. Thesis. University of Manchester, 1979.

[9] Suleiman M.B., Solving higher order ODEs directly by the direct integration method, *J. Appl. Math. Comput.*, 1989; **33**: 197-219.