Iranian Journal of Mathematical Sciences and Informatics
Vol. 3, No. 1 (2008), pp 63-76
DOI: 10.7508/ijmsi.2008.01.006

Some implementation aspects of the general linear methods withinherent Runge-Kutta stability

P. Mokhtary and S. Mohammad Hosseini [1]
Department of Mathematics, Tarbiat Modares University
P.O.Box: 14115-137, Tehran, Iran

Email: payam_mokhtare@yahoo.com
Email: hossei_m@modares.ac.ir

Abstract. In this paper we try to put different practical aspects of the general linear methods discussed in the papers [1,6,7] under one algorithm to show more details of its implementation. With a proposed initial step size strategy this algorithm shows a better performance in some problems. To illustrate the efficiency of the method we consider some standard test problems and report more useful details of step size and order changes, and number of rejected and accepted steps along with relative global errors.

Keywords: General linear methods; Variable step size; Inherent Runge-Kutta stability; Error estimation.

2000 Mathematics subject classification: 65L05.

1. introduction

To design an efficient algorithm for the numerical solution of ordinary differential equation

$$\begin{cases} y'(x) = f(y(x)), \quad x \in [x_0, X] \\ y(x_0) = y_0, \quad y \in \mathbb{R}^m \end{cases} \tag{1}$$

[1]Correspondence author

Received 26 October 2008; Accepted 26 November 2008

it is necessary to deal with two major issues. The first one is the selection of the formula used to advance the computations. Many methods are available for this purpose such as Runge-Kutta, linear multi step or general linear methods. The second important issue is the efficient implementation of these formulas. This paper is concerned with the second issue for general linear methods with inherent Runge-Kutta stability(IRKS). These formulas have been introduced by Butcher [3] and are written as

$$\begin{cases} Y = h(A \otimes I)F(Y) + (U \otimes I)y^{[n-1]} \\ y^{[n]} = h(B \otimes I)F(Y) + (V \otimes I)y^{[n-1]}, \end{cases} \tag{2}$$

where I is the identity matrix of dimension m. $h = x_n - x_{n-1}$ the step size, and $F(Y) = [f(Y_1), ..., f(Y_s)]^T$. The stages Y_i approximate the solution $y(x_{n-1} + c_i h)$ at nodal points c_i and Nordsieck vector $y^{[n]}$, $y_i^{[n]} \approx h^{i-1} y^{(i-1)}(x_n), i = 1, ..., s$. The order of the internal stages is denoted by q and the order of the output approximation is denoted by p and $s = p + 1$. Since Runge-Kutta methods have good stability properties it would be desirable to obtain general linear methods with the same stability regions as some equivalent Runge-Kutta methods. It is a complicated task to determine the conditions on the method in order to ensure Runge-Kutta stability in its most general sense. Butcher and Wright, [2,6](2002), have given a sufficient condition to ensure the general linear method to have Runge-Kutta stability. This leads to a condition known as inherent Runge-Kutta stability(IRKS). The IRKS conditions relate the coefficient matrices of the method with a doubly companion matrix X to satisfy

$$BA = XB \qquad BU \equiv XV - VX \qquad \sigma(V) = \{0, 1\}.$$

Huang, [7](2005), implemented general linear methods for stiff ordinary differential equation with a fixed order using a fortran code.
Butcher and Podhaisky, [1](2006), investigated the special estimation of local truncation errors for methods in variable order mode. In order to give the details of implementation we need the following information that are presented in the following sections. In section 2 a brief review on the topic and basic definitions and facts related to IRKS is given. In section 3 the iteration scheme for evaluation of stage values using some variant of Newton iteration is introduced. In section 4 we describe starting procedures for constructing the initial Nordsieck vector. Section 5 is devoted to some computable estimates of the local truncation errors. In section 6 strategies for changing step size and order is discussed. In section 7 we talk about a logical initial step size formula. Finally,in section 8, the results of some numerical experiments are presented and discussed.

2. Methods with inherent Runge-Kutta stability

Considering the *scalar* linear equation $y' = qy$, the method (2) reduces to the matrix recursion

$$y^{[n]} = M(z)y^{[n-1]}, \qquad z = hq,$$

with the stability matrix $M(z) = V + zB(I - zA)^{-1}U$. The method is said to have Runge-Kutta stability if $M(z)$ has only one non-zero eigenvalue $R(z)$, where $R(z)$ is Runge-Kutta stability matrix. This property was introduced in [6] and is a desirable property because a method with this structure forces the computed results to behave like those produced by a one-step method. Furthermore, by confining ourselves to methods with this property, it is possible to draw upon the wealth of existing knowledge of stable Runge-Kutta methods. Since the eigenvalue condition is very difficult to handle, Butcher and Wright have introduced the concept of inherent Runge-Kutta stability [6]. Details of the procedure to construct GLMS with inherent Runge-Kutta stability can be found in [2,6].

3. Iteration scheme

Huang and Butcher [7] introduced an iteration scheme to solve stage values. The stages can be expressed by the following equation

$$Y_i - \lambda h f(Y_i) = \sum_{j=1}^{i-1} a_{ij} h f(Y_j) + \sum_{j=1}^{s} u_{ij} y_j^{[n-1]}, \quad i = 1, 2, ..., s. \tag{3}$$

Applying the Newton method here, one then solves an iterative linear system in which the iteration matrix is

$$I - h\lambda J = LU$$

where the matrices L and U are its LU factors and J is the jacobian matrix. In the Newton iteration process, one needs a starting guess (vector) $Y^{[0]}$ as an approximation to Y. In [7] there were discussed three different stage predictors to estimate the value of $Y^{[0]}$ that are explained here.

3.1. **Taylor expansion predictor.** We use here the known values from the previous step without any additional computational cost. For the Taylor expansion the higher order derivatives are obtained from the Nordsieck vector $y^{[n-1]}$ of the previous step. Therefore, for i^{th} stage, we have

$$
\begin{aligned}
Y_i &= y(x_{n-1}+c_ih)+O(h^{p+1})\\
&= y(x_{n-1})+\sum_{j=1}^{p}\frac{c_i^j}{j!}h^j y^{(j)}(x_{n-1})+O(h^{p+1})\\
&= \sum_{j=0}^{p}\frac{c_i^j}{j!}y_{j+1}^{[n-1]}+O(h^{p+1})\\
&\approx \sum_{j=0}^{p}\frac{c_i^j}{j!}y_{j+1}^{[n-1]}.
\end{aligned}
\tag{4}
$$

3.2. **Newton interpolation predictor.** To improve the accuracy of the prediction, an alternative stage predictor uses the Newton interpolation to find values of $Y_i^{[0]}$. Newton interpolation has the following general formula:

$$\phi(t)\approx\phi(t_1)+(t-t_1)\phi(t_1,t_2)+(t-t_1)(t-t_2)\phi(t_1,t_2,t_3),$$

where $\phi(.)$, $\phi(.,.)$, and $\phi(.,.,.)$ are defined recursively by divided difference formula.

We first compute Y_1, Y_2 from Taylor expansion predictor. Then $Y_3^{[0]}$ is computed from Y_1, Y_2 and $hf(Y_2)$ that are already obtained from previous two stages. This means that we can find $\phi(c_3)$, using the given values $\phi(c_1), \phi(c_2), \phi'(c_2)$. This leads to the general form

$$Y_i^{[0]} = Y_{i-2} + c_i\ h\ f(Y_{i-1}). \tag{5}$$

3.3. **Hermite interpolation predictor.** Another predictor is based on Hermite interpolation formula. In this predictor, we use Y_{i-2}, $hf(Y_{i-2})$, Y_{i-1} and $hf(Y_{i-1})(i>2)$ to predict the value of $Y_i^{[0]}$. For the first two stages the Taylor expansion predictor is used and for 3rd and more one uses

$$Y_i^{[0]} = aY_{i-2} + bhf(Y_{i-2}) + cY_{i-1} + dhf(Y_{i-1}). \tag{6}$$

where a, b, c, d can be obtained by solving the following system of equation

$$
\begin{gathered}
a+c=1,\\
ac_1+b+cc_2+d=c_3,\\
a\frac{c_1^2}{2!}+bc_1+c\frac{c_2^2}{2!}+dc_2=\frac{c_3^2}{2!},\\
a\frac{c_1^3}{3!}+b\frac{c_1^2}{2!}+c\frac{c_2^3}{3!}+d\frac{c_2^2}{2!}=\frac{c_3^2}{3!}.
\end{gathered}
$$

that is obtained by expanding both sides of the equation (4) about x_n. The numerical experiments for these three predictors will be presented in section 8.

4. Starting procedure

For the IRKS methods it is necessary to approximate the initial Nordsieck vector

$$y^{[0]} = \begin{bmatrix} y(x_0) \\ hy'(x_0) \\ h^2y''(x_0) \\ \vdots \\ h^py^{(p)}(x_0) \end{bmatrix} + O(h^{p+1}).$$

Huang [7] used special starting procedure based on modified singly implicit Runge-Kutta methods with $p+1$. Huang obtained matrix coefficients by solving some linear systems, that in this algorithm matrix A is lower triangular but in [6] Butcher and Wright obtained an algorithm for starting procedure that have abstract form with matrix relations and the same as that of the general linear method in which the matrix A is full. In this algorithm we have

$$U = ee_1^T, \qquad V = e_1e_1^T.$$

The coefficients of the method are

$$\begin{aligned} U &= C - ACK \\ V &= I - BCK \\ AC &= CJ \\ BC &= J, \end{aligned} \tag{7}$$

where $K = [0 \;\; e_1 \;\; e_2 \;\; \cdots \;\; e_{p-1} \;\; e_p]$ and $J = K', C = [1 \;\; c \;\; \frac{c^2}{2!} \;\; \cdots \;\; \frac{c^{p-1}}{(p+1)!} \;\; \frac{c^p}{p!} \;]$. The numerical results for these two algorithms don't differ basically. But as in the Wright's algorithm there is an explicit formula for matrix coefficients we use this formula for our starting procedure.

5. Error estimation

Suppose that C_p is the error constant for a method of order p. Then variable step size requires the estimation of $h^{p+1}y^{(p+1)}(x_n)$ so that $E_n = \| C_ph^{p+1}y^{(p+1)}(x_n) \|$ can be calculated as an approximation to the local truncation error. Wright [6] uses special form for local error estimate that is

$$h^{p+1}y^{(p+1)}(x_n) = \phi^T hf(Y^{[n]}), \tag{8}$$

where ϕ is the solution of the following linear system

$$\phi^T C = e_{p+1}.$$

But Butcher and Podhaisky consider the local error estimate based on a linear combination of the form

$$\delta_0 y_2^{[n-1]} + \sum_{i=1}^{p+1} \delta_i h F_i \,, \tag{9}$$

where $\delta_0 = -\sum_{i=1}^{p+1} \delta_i$. The calculation details of $\delta_i \quad , i = 0, 1, ..., p+1$ can be found in [1]. Since we want to implement the method in variable order mode then we must estimate $h^{p+2} y^{(p+2)}$. To estimate $h^{p+2} y^{(p+2)}$ Wright in [6] and Butcher in [1] use the various difference between E_n and a suitably scaled value of E_{n-1}. Details of local estimators for variable order mode can be found in [1,6]. The numerical experiments for these two error estimators will be presented in section 8.

6. Changing step size and order control

To obtain an effective implementation of any numerical method, step size change is a necessary requirement. Let $h_n = x_n - x_{n-1}$ denote the step size of step n. The new Nordsieck vector corresponding to step size $h_{n+1} = r h_n$ can be obtained from $y^{[n]}$ just by multiplication with a diagonal matrix, $(D(r) \otimes I) y^{[n]}$ with $D(r) = diag(1, r, ..., r^p)$. This is equivalent to replace B by $D(r)B$ and V by $D(r)V$ in the method. Butcher and Podhaisky [1] use this formula for step size and order changing:
while we have a estimates for the leading error term $h_n^{p+1} C_p y^{(p+1)}(x_n)$, after n^{th} step using order p, the new step size $h_{n+1,q}$ to continue the integration with order $q \in \{p-1, p, p+1\}$ is computed by the following formula, using the error estimation (9) for $q = p$ and $q = p+1$, see Butcher [1]:

$$h_{n+1,q} = \max\left(0.5,\ \min(2, 0.85\left(\frac{s-1}{s} \parallel h_{n,p}^{q+1} C_q y_{est}^{(q+1)} \parallel_{wrms}\right)^{-1/(q+1)})\right) h_{n,p}, \tag{10}$$

where

$$\parallel x \parallel_{wrms} = \sqrt{\sum_{i=1}^{m} \left(\frac{x_i}{Atol + Rtol \times abs(y_{1,i}^{[n]})}\right)^2}. \tag{11}$$

that Atol and Rtol respectively were absolute and relative tolerances.
Huang [7] uses another step size formula, using the error estimation (8) for $q = p$ and $q = p+1$, that is written as

$$h_{n+1,q} = \min\left(2,\ \max(0.9\,(\frac{Tolerance}{\parallel h_{n,p}^{q+1} C_q y_{est}^{(p+1)} \parallel})^{(1/(q+1))})\right) h_{n,p}\ , \tag{12}$$

where $y_{est}^{(q+1)}$ denotes the estimate to $y^{(q+1)}(x_n)$ which is the last component of the Nordsieck vector $y_p^{[n]}$ for $q = p-1$. Infinity norm or 2-norm can be used

in Huang formula.
In the Huang's step size formula to have an optimal step size we require the norm of local error to be approximately equal to the given Tolerance while in the Butcher's step size approach, at each step the i^{th} component of the error, E, of the solution must satisfy

$$|E(i)| \leq \max(Rtoll \times |y(i)|, Atoll),$$

where $Rtoll$, the Tolerance of relative accuracy, controls the number of correct digits in the approximate solution and $Atoll$, the tolerance for the absolute error, controls the absolute error in the approximate solution. Roughly speaking, this means that $Rtoll$ correct digits in all the solution components except those smaller than thresholds $Atoll(i)$ is requested.
For order changing the proper step size h_{n+1} is not chosen as the largest of $h_{n+1,p-1}, h_{n+1,p}, h_{n+1,p+1}$, but rather that one which minimizes the computational cost. The standard strategy for the order change is to choose the order q is such a way that step size $h_{n+1,d}$ per unit of computational cost is maximized,i.e.,

$$\frac{h_{n+1,q}}{\omega_d} \rightarrow \max$$

where $q \in p-1, p, p+1$ and ω_d is computational cost. Computational cost depends on the size and complexity of the differential equation system. The number of Newton iterations and general nature of the system have significant effect. For IRKS methods the computational cost is often measured by $s = p+1$. But they have mistakenly assumed that each stage has the same cost and then for order changing consider the ratios

$$\frac{h_{n+1,p-1}}{s-1} \quad , \quad \frac{h_{n+1,p}}{s} \quad , \quad \frac{h_{n+1,p+1}}{s+1}.$$

So, in order to balance the effects of computational costs the safety factors $\delta_{p-1}, \delta_p, \delta_{p+1}$ are included. We then put $\delta_p = \frac{s-1}{s}$ and the new order is chosen to maximize

$$\frac{(s-2)h_{n+1,p-1}}{(s-1)^2}, \qquad \alpha\frac{(s-1)h_{n+1,p}}{s^2}, \qquad \text{and} \quad \frac{s\ h_{n+1,p+1}}{(s+1)^2},$$

where $\alpha = 1.2$ is a safety factor to avoid frequent order changes.
The numerical experiments for these two step sizes chosen will be presented in section 8.

7. Initial step size

In any efficient implementation a bad initial choice for h should be quickly repaired by the step size control. Nevertheless, when this happens too often and when the choices are too bad, much computing time can be wasted. We take up an idea of Gladwell, Shampine and Brankin (1987)[4] which is based on the hypothesis that

$$localerror \approx Ch^{p+1}y^{(p+1)}(x_0).$$

Since $y^{(p+1)}(x_0)$ is unknown we shall replace it by approximations of the first and second derivatives of the solution. The resulting algorithm is then follows:

a) *evaluate the function value* $f(x_0, y_0)$ *at the initial point. Then put* $d_0 = \| y_0 \|$ *and* $d_1 = \| f(x_0, y_0) \|$, *where the norm is the same as (11),*

b) *As a first guess for the step size let*

$$h_0 = 0.01 \ (d_0/d_1).$$

So that the increment of an explicit Euler step is small compared to the size of the initial value. If either d_0 or d_1 is smaller than 10^{-5} we put $h_0 = 10^{-6}$.

c) *Perform one explicit Euler step,* $y_1 = y_0 + h_0 f(x_0, y_0)$ *, and compute* $f(x_0 + h_0, y_1)$,

d) *Compute* $d_2 = \| f(x_0 + h_0, y_1) - f(x_0, y_0) \|/h_0$, *as an estimate of the second derivative of the solution. The norm being the same as in* **a)**

e) *Compute a step size* h_1 *from the relation*

$$h_1^{p+1}.\max(d_1, d_2) = 0.01.$$

if $max(d_1, d_2) \leq 10^{-15}$ *we put* $h_1 = max(10^{-6}, h_0\ 10^{-3})$.

f) *Finally to use some scales of* $h_0,\ h_1$ *we propose as initial step size*

$$h = min(100\ h_0, h_1). \tag{13}$$

Note that this algorithm does not try to provide a sharp satisfactory initial step size in each problem, but usually gives a reasonably good guess for it or at least avoids a very bad choice. The Numerical experiments for this initial step size will be presented in section 8.

8. Numerical experiments

In this section we consider some numerical examples and apply IRKS method using different ideas mentioned above to show their numerical merits. Clearly, an efficient implementation would be emerged by putting together the well behaved strategies of doing different jobs in an algorithm. We used here methods with free parameters assigned as follows: (see[1,5,6])
For the method of **order** 1 we have chosen

$$\lambda = 0.3, \quad c_1 = [\lambda \ \ 1],$$

for the method of **order** 2 we have chosen

$$\lambda = \frac{4}{9}\ ,\ c = [\frac{1}{3} \quad \frac{2}{3} \quad 1]\ ,\ \ \beta_1 = 0,$$

for the method of **order** 3 we have chosen

$$\lambda = \frac{9}{40}\ ,\ c = [\frac{1}{4} \quad \frac{1}{2} \quad \frac{3}{4} \quad 1]\ ,\ \beta_1 = 0\ ,\ \beta_2 = 0\ ,\ T = I,$$

and for the method of **order** 4 we have chosen

$$\lambda = \frac{2}{7}\ ,\ c = [-0.63 \quad 0.67 \quad 0.95 \quad -0.82 \ \ 1]\ ,\ \beta_1 = 0\ ,\ \beta_2 = 0\ ,\ \beta_3 = 0\ ,\ T = [e_2 \ \ e_1 \ \ e_3].$$

The test problems that we consider are

Bruss [4] : The(nonstiff) Brusselator equation

$$y_1' = 1 + y_1^2 y_2 - 4y_1,$$
$$y_2' = 3y_1 - y_1^2 y_2,$$

with initial values $y(0) = [1.5, 3]^T$ and $x \in [0, 20]$.

Hires [5] : A stiff ODE with $m = 8$ equations.

Oreg [5] : The Oregonator system,

$$y_1' = 77.27(y_2 + y_1(1 - 8.375 \times 10^{-6} y_1 - y_2)),$$
$$y_2' = \frac{1}{77.27}(y_3 - (1 + y_1)y_2),$$
$$y_3' = 0.161(y_1 - y_3),$$

with $y(0) = [1, 2, 3]^T$ and $x \in [0, 30]$.

VPOL [4] : The Vanderpol system,

$$y_1^{'} = y_2$$
$$y_2^{'} = 10^6((1 - y_1^2)y_2 - y_1)$$

with initial vector $y(0) = [2, 0]^T$ and $x \in [0, 2]$.
In the following we show the performance of constant step size against variable step size, initial step size, stage predictors, local error estimators (8) and (9), step size choosing strategies (12) and (10). At the end some comparison results are also presented for variable order method.

8.1. **Constant step size.** To understand how IRKS methods perform, we first use constant (fixed) step size. The test problem used here is the Vpol problem

Table 1: The error and total steps versus step size by a method of order 3

Tolerance=1e-5	total steps	relative global error
variable step size	3863	1.5295e-005
constant step size	nstep=100	NAN
	nstep=1000	2.3031e+256
	nstep=10000	1.1480e+038

This confirms the need of variable step size algorithms for solving ordinary differential equations for an efficient computation.

8.2. **Effect of the initial step size.** In a variable step size algorithm the choice of an appropriate initial step size h_0 has much affect on the total number of required steps. We investigate the effect of our proposed initial step size (13) on the Hires problem, using a method of order 4 with different tolerances.

Table 2: The effects of initial step size, h_0, for a method of order 4

Tolerance	h_0	Total steps	reject steps	function evaluation
1e-5	1e-4	706	48	3530
	1e-3	709	39	3545
	1.04e-2 **by** *(13)*	701	32	3505
1e-10	1e-4	590	21	2950
	1e-5	578	19	2890
	1.60e-3 **by** *(13)*	591	21	2955

It is seen that the smaller the tolerance is, The smaller the initial stepsize should be with less rejected steps and less function evaluation, therefore we choose a slightly larger initial stepsize for bigger tolerance.

8.3. **Performance of stage predictors.** Huang [7] proposed three predictors discussed in chapter 3. For an order 2 method, we only have three stages, the Taylor expansion predictor is good enough. For the methods of orders $3, 4$, we try all three predictors on the Hires problem [5]. Here "Tay-expan" stands for Taylor expansion, "Newt-interp" stands for Newton interpolation, and "Herm-interp" stands for Hermite interpolation.

Table 3: Results for an order 3 method

Tolerance	Predictor	Total steps	Reject steps
1e-5	Tay-expan	904	12
	Newt-interp	888	5
	Herm-interp	891	6
1e-8	Tay-expan	1042	3
	Newt-interp	899	1
	Herm-interp	897	0

Table 4: Results for an order 4 method

Tolerance	Predictor	Total steps	Reject steps
1e-5	Tay-expan	705	33
	Newt-interp	701	32
	Herm-interp	701	32
1e-8	Tay-expan	751	26
	Newt-interp	664	49
	Herm-interp	653	39

Numerical results show that when the Tolerance is bigger, the second predictor performs slightly better than the other two predictors in terms of the total and reject numbers for the methods of orders $3, 4$. When Tolerance is small, the third predictor performs better with a fewer total steps and a reasonable number of rejects.

8.4. **Effect of the error estimate and step size chosen approaches.** In sections 5,6 we discussed about Wright's and Butcher's schemes for the local error estimates and Huang's and Butcher's schemes for variable step size approaches respectively. In this section we apply these schemes to Hires and Oreg problems[5]:

Table 5: Results for an order 3 on Hires problem

Tolerance	step size - error estimator	total steps	reject steps	relative global error
1e-5	(10)- (9)	888	5	7.900e-03
	(12)- (8)	888	5	1.450e-02
1e-8	(10)- (9)	897	0	7.311e-06
	(12)- (8)	906	1	1.421e-05

Table 6: Results for a method of order 3 on Oreg problem

Tolerance	step size - error estimator	total steps	reject steps	relative global error
1e-6	(10)- (9)	1400	6	8.515e-06
	(12)- (8)	2315	12	1.153e-06
1e-8	(10)- (9)	1783	1	2.805e-07
	(12)-(8)	4973	9	4.034e-08

The numerical results show that Butcher's scheme for the local error estimator with Butcher's step size scheme for variable step size approach performs better with fewer total and reject numbers of steps and works better that Wright's scheme for the local error estimate with Huang's step size scheme for variable step size approach. Then according to the above experiments one could suggest the following simple algorithm for implementation of GLMs:
1: use the smaller initial stepsize for small tolerance, and slightly larger initial stepsize for bigger tolerance,
2: use Newton interpolation predictor for a relatively large Tolerance and Hermite interpolation predictor for a relatively small Tolerance,
3: use Wright's starting procedure to approximate the initial Nordsieck vector,
4: use Butcher's local error estimate scheme to estimate $h^{p+1}y^{(p+1)}$,
5: use Butcher and Podhaisky's scheme for variable step size and variable Order approach.

8.5. **Variable order results.** In section 6 we discussed about variable order approach. We in this section test this approach on Bruss, Hires, and Oreg problems [4,5] and give order changing figures

Table 7: Results with Tolerance $= 1e-6$

Test problem	total steps	reject steps	relative global error
Bruss	158	7	4.857e-04
Hires	874	9	4.000e-03
Oreg	620	67	7.138e-05

9. Conclusions

This paper showed different practical aspects of the IRKS methods discussed in the papers [1,6,7] under one algorithm with a proposed initial step size algorithm that performed well on some classical test problems.

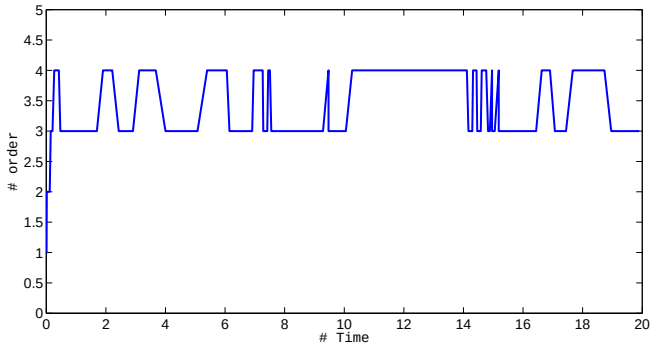

FIGURE 1. Order changing - Tolerance $= 1e-6$ - Bruss Problem

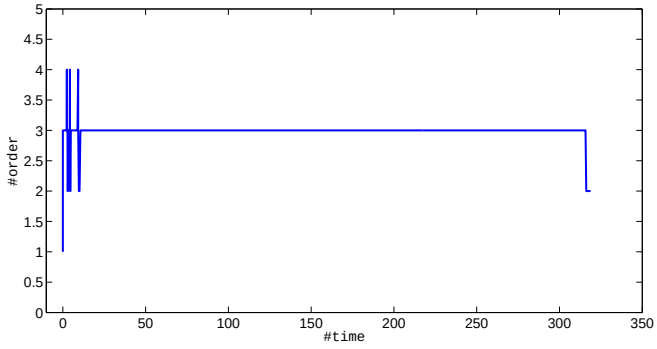

FIGURE 2. Variable order - Tolerance $= 1e-6$ - Hires Problem

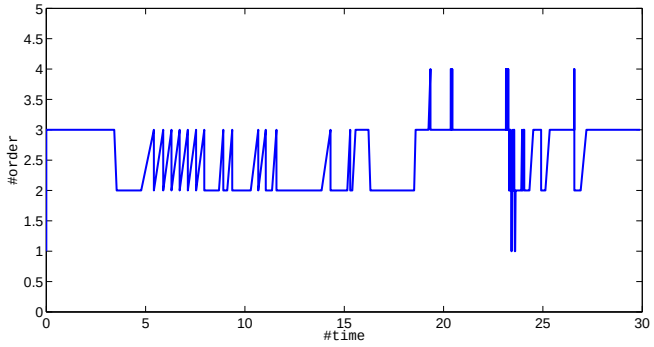

FIGURE 3. Variable order - Tolerance $= 1e-6$ - Oreg Problem

References

[1] J.C. Butcher, H. Podhaisky, On error estimation in general linear methods for stiff ODEs, Appl. Numer. Math., **56** (2006), 345-357.

[2] J.C. Butcher, W. Wright, The construction of general linear methods, *BIT*, **43** (2003) 695-721.

[3] J.C. Butcher, Numerical Methods for Ordinary Differential Equation, Wiley, New York, 2003.

[4] E. Hairer, S.P. Nørsett, G. Wanner, Solving Ordinary Differential Equations I, Springer, Berlin,1993.

[5] E. Hairer, G. Wanner, Solving Ordinary Differential Equations II, Springer, Berlin,1996.

[6] W. Wright, General linear Methods with Inherent Runge-Kutta Stability, PhD thesis,The University of Auckland, 2002.

[7] S.Y. Huang, Implementation of General Linear Methods for Stiff Ordinary Differential Equations, PhD thesis,The university of Auckland, 2005.