UNIVERSITÀ DEGLI STUDI DI BARI

DIPARTIMENTO INTERUNIVERSITARIO DI MATEMATICA

Solving Ordinary Differential Equations by Generalized Adams Methods: Properties and Implementation Techniques

F. Iavernaro, F. Mazzia

Report n. 24/97

Address:
Via E. Orabona 4, I-70125 Bari (Italy)
Fax: +39 80 5460612
URL: http://www.dm.uniba.it

Solving Ordinary Differential Equations by Generalized Adams Methods: properties and implementation techniques. *

F. Iavernaro, F. Mazzia ‡

Abstract

Generalized Adams Methods of order 3,5,7 and 9 are used to find numerical solutions of Initial Value Problems. The effectiveness of these methods for the treatment of stiff problems is shown on the basis of their attractive properties and an efficient technique to deal with the algebraic nonlinear systems representing the discrete counterpart of the continuous problem. Numerical examples are also presented in which an experimental code based on these methods is compared with two well known codes for ODEs. The numerical results are quite satisfactory and suggest that these methods may have a useful role in the solution of stiff ODEs.

Key words. linear multistep formulas, boundary value methods, runge-kutta methods, stiff initial value problems

AMS(MOS) subject classifications. 65L05, 65L20

1 Introduction

During the last years there has been a great deal of interest in deriving efficient methods for the numerical solution of stiff initial value problems of the form

$$\begin{cases} \mathbf{y}' = \mathbf{f}(t,\mathbf{y}), \qquad t \geq t_0, \qquad \mathbf{f} : \mathbb{R} \times H \to H, \\ \mathbf{y}(t_0) = \mathbf{y}_0. \end{cases} \tag{1}$$

where $H = \mathbb{R}^r$. This interest is focused usually on deriving codes that can be competitive with existing ones or that can be implemented on parallel machines. It is possible to classify general purpose codes for ODEs into two classes according to the methods implemented, that is, linear multistep or Runge-Kutta methods.

*Work supported by MURST (40% project) and partially by the European Community (Marie Curie - Training and Mobility of Researchers, Contract N^o ERBFMBICT961643).

†Dipartimento di Matematica, Università di Bari. Via Orabona, 4. I-70125 Bari (Italy).

‡E–mail : labor@alphamath.dm.uniba.it, Fax: +39 80 5442722

There are a lot of sequential codes designed to handle stiff ODEs. Historically the most well known code is LSODE, based on Backward Differentiation Formulae (BDFs), which has been used for a long time in spite of its well known disadvantages, due to lack of stability of high order BDFs [16]. Other important codes in the same class are: DASSL and its variants, based on BDFs for the numerical solution of the wider class of stiff differential-algebraic equations [4]; MEBDF, based on extended BDFs, A-stable formulae up to order 4 and A(α)-stable formulae up to order 9 [12, 14]; VODE and its modification, based on BDF variable coefficient formulae [5].

Concerning Runge-Kutta methods one of the most efficient codes is RADAU5, a code based on Radau-type formulae of order 5 [14]. This code is usually very robust and efficient on difficult problems because of its good stability properties. The DESI code based on diagonally extended singly implicit RK formulae is the first code based on RK that adopts an order variation strategy [10]. Indeed, a disadvantage of codes based on RK-formulas is that an order variation strategy is a hard task to handle because of the difficulty of having a convenient representation of the local truncation errors.

The aim of this paper is to present an experimental general-purpose code based on Boundary Value Methods (BVMs), a class of methods for ODEs that has been succesfully used for the solution of initial and boundary value ODE problems (see [7]). A BVM may be applied to the entire time interval under consideration or sequentially in adjacent subintervals giving rise to block-BVMs. As shown in [17], these latter may be seen to belong to the wider class of General Linear Methods (GLMs) and indeed the theory on GLMs has been exploited to derive results on convergence and order conditions.

Our interest in this paper is devoted to a particular class of block-BVMs, namely the Generalized Adams Methods (GAMs) which have been implemented in a code for the solution of IVPs. The definition and the general properties of GAMs and block-BVMs may be found in [7], while in [17] convercence and stability results have been studied.

Block-BVMs have the dual property to be defined by a set of linear multistep methods (LMMs) and to belong to the class of Runge-Kutta methods. As a consequence they possess features of both LM and RK formulae and on attempting to costruct a code based on BVMs, the peculiarities of both methods may be taken into consideration. With RK-formulae they share good stability properties for high order methods, a stepsize variation strategy typical of one-step schemes (which is simpler than those used for LMMs), and suffer from a high computational effort per step. On the other hand they have the same representation of the local truncation error as LM-formulae, and this allows one to perform a simple technique for varying the order while integrating the IVP.

In this paper we wish to describe an experimental variable step, variable order code, that has been realized using the A-stable, high order block-GAMs. Moreover we evaluate its performance on a set of challenging stiff test problems and compare this performance with that of some standard widely used codes.

In the next section we briefly describe the GAMs and we specify the formulae used in the code. Section 3 is devoted to the resolution techniques of the nonlinear systems arising from the application of a GAM, while in section 4 we exploit the simple structure of the local truncation error to derive an estimation of the error and an efficient order changing rule. In section 5 we report some numerical tests to show the effectiveness of the code.

2 The Generalized Adams Methods

The Generalized Adams Methods of order 3,5,7 and 9 may be identified by pairs $(A_s^e, B_s^e),\ s \in I = \{3,5,7,9\}$ of $N_s \times N_s + 1$ matrices used as linear operators $H^{N_s+1} \to H^{N_s}$. The integration procedure is typical of a standard one-step implicit formula with the difference that now the method need not to be constant while advancing in time. At step n an integer s is chosen in the set I by a suitable criteria and the method (A_s^e, B_s^e) is used to approximate the true solution, say $\hat{\mathbf{y}}(t)$, on the time block $W_n = [t^{(n-1)}, t^{(n)}]$. More precisely, if we want to advance the solution starting at the time $t^{(n-1)}$, we need to fix a stepsize h_n and therefore define W_n by means of the uniform mesh $t^{(n-1)} = t_0^{(n)} < t_1^{(n)} < \ldots < t_{N_s}^{(n)} = t^{(n)},\ \ t_i^{(n)} = t_{i-1}^{(n)} + h_n$. Finally we use the fixed GAM to obtain on that mesh a numerical solution represented by the vector $\mathbf{y}^{(n)} = [\mathbf{y}_1^{(n)}, \mathbf{y}_2^{(n)}, \ldots, \mathbf{y}_{N_s}^{(n)}]^T$, where we do not include $\mathbf{y}_0^{(n)}$ since we assume that it coincides with the last component of $\mathbf{y}^{(n-1)}$ which has been previously computed.

Denoting by $\mathbf{a}_s$ and $\mathbf{b}_s$ the first columns of A_s^e and B_s^e respectively and performing the partitions

$$A_s^e = [\mathbf{a}_s, A_s],$$
$$B_s^e = [\mathbf{b}_s, B_s],$$

the integration of (1) then requires the solution at each step of the following nonlinear problem:

$$A_s \mathbf{y}^{(n)} - h_n B_s F(\mathbf{y}^{(n)}) = \boldsymbol{\delta}_s^{(n)}, \qquad s \in I, \tag{2}$$

where $F(\mathbf{y}^{(n)}) = [\mathbf{f}(t_1^{(n)}, \mathbf{y}_1^{(n)}), \mathbf{f}(t_2^{(n)}, \mathbf{y}_2^{(n)}), \ldots, \mathbf{f}(t_{N_s}^{(n)}, \mathbf{y}_{N_s}^{(n)})]^T$ and $\boldsymbol{\delta}_s^{(n)} = -\mathbf{a}_s \mathbf{y}_0^{(n)} + h_n \mathbf{b}_s^{(n)} \mathbf{f}(t_0^{(n)}, \mathbf{y}_0^{(n)})$ contains the known terms that involved the previous computed solution. An equivalent but more compact and sometimes useful way to describe a general GAM is to directly use in (2) the 'extended' matrices A_s^e and B_s^e, that is

$$A_s^e \mathbf{y}^{e(n)} - h_n B_s^e F(\mathbf{y}^{e(n)}) = \mathbf{0}, \tag{3}$$

where now $\mathbf{y}^{e(n)} = [\mathbf{y}_0^{(n)}, \mathbf{y}_1^{(n)}, \ldots, \mathbf{y}_{N_s}^{(n)}]^T$ includes the known solution at $t_0^{(n)}$ and the function F has an extra argument.

We now examine in more detail how the coefficients of the matrices A^e and B^e are defined for the class of GAMs. To simplify the notation, we omit the indices n and s in the above expressions. The i-th component of the system (3) is really a k-step linear method with $k_1^{(i)}$ initial and $k_2^{(i)}$ final conditions, $k = k_1^{(i)} + k_2^{(i)}$, which may be written as

$$\mathbf{y}_i - \mathbf{y}_{i-1} = h \sum_{j=-k_1^{(i)}}^{k_2^{(i)}} \beta_{ij} \mathbf{f}_{i+j}. \tag{4}$$

This formula computes $\mathbf{y}_i$ by means of approximations to the solution at $k_1^{(i)}$ left points and $k_2^{(i)}$ right points. The coefficients β_{ij} are uniquely determined by imposing the condition that the formula has maximum order $p = k+1$, while if the index ν is defined as

$$\nu = \begin{cases} (k+1)/2 & \text{for odd k}, \\ k/2 & \text{for even k}, \end{cases}$$

the integers $k_1^{(i)}$ (and therefore $k_2^{(i)} = k - k_1^{(i)}$) are defined as follows:

$$k_1^{(i)} = \begin{cases} i & \text{for } i = 1, \dots, \nu - 1, \\ \nu & \text{for } i = \nu, \dots, N - \nu\ , \\ i - N + 2\nu & \text{for } i = N - \nu + 1, \dots, N. \end{cases} \tag{5}$$

We use the GAMs obtained in correspondence with an even number of steps and therefore with odd order. As we see from (5), whatever the dimension $N \geq k$ of the matrices A and B each GAM consists of a finite number of different formulae: the first $\nu - 1$ formulae are called initial methods while the last ν are called final methods. Between the initial and final methods we find the so called basic method which is repeated N-k+1 times. Properties of the basic method are passed to the underlying block-BVM when the dimension N is large enough.

For reasons that will be clear below we have fixed the following dimensions for the four selected GAMs: $N_3 = 4$, $N_5 = 6$, $N_7 = 8$, $N_9 = 9$. In [17] the coefficients β_{ij} of the initial, basic and final methods are reported while it is easily seen that the matrices A_s are bidiagonal with 1 and -1 as diagonal and lower-diagonal entries respectively. The authors also showed that all considered GAMs are A-stable. With reference to equation (3), if $\hat{\mathbf{y}}^e = [\hat{\mathbf{y}}(t_0), \hat{\mathbf{y}}(t_1), \dots, \hat{\mathbf{y}}(t_n)]^T$ is the true solution on the fixed mesh, the local truncation error is defined as

$$\boldsymbol{\tau}(h) \equiv A^e \hat{\mathbf{y}}^e - hB^e F(\hat{\mathbf{y}}^e),$$

and has components

$$\boldsymbol{\tau}_i(h) = c_i \hat{\mathbf{y}}^{(p+1)}(\xi_i) h^{p+1}, \qquad i = 1, \dots, N \tag{6}$$

where $\xi_i \in [t_0, t_N]$ and c_i are the error constants of the method. In the following we will make use of the following norm on the vectors $\mathbf{y} \in H^N$

$$||\mathbf{y}|| = \max_{1 \leq i \leq N} ||\mathbf{y}_i||_2.$$

3 The solution of the nonlinear systems

Many implicit schemes for the ODE problem (1) require the solution of systems of algebraic equations of the form (2). For example when an implicit s-stage Runge-Kutta method is used on problem (1), the associated discrete problem for the computation of the internal stages assumes the form:

$$S(\mathbf{y}^{(n)}) \equiv \mathbf{y}^{(n)} - h_n C F(\mathbf{y}^{(n)}) - \boldsymbol{\delta}^{(n)} = \mathbf{0}, \tag{7}$$

where C is an $N \times N$ matrix used as a linear operator mapping $H^N \rightarrow H^N$.

The two nonlinear problems (2) and (7) can be solved by exactly the same technique if A is nonsingular, even if we prefer to keep the matrices A and B separated in order to have a simple representation of the local error.

One of the most successful ways of solving the problem (7) is the modified Newton's iteration

$$(I - hCD)\Delta\mathbf{y}^{k+1} = -S(\mathbf{y}^k), \qquad i = 1, 2, \ldots, \tag{8}$$

where D is a block diagonal matrix containing in each block the Jacobian J_0 of the function $\mathbf{f}$ evaluated at (t_0, y_0). The use of the modified iteration has the obvious advantage of requiring that the matrix $(I - hCD)$ must be factored only once, to perform the iteration (8). In this case the leading term in the arithmetic complexity is $\frac{2}{3}(N \cdot r)^3$ flops, where we count as one *flop* one of the four basic *fl*oating point *op*erations with real quantities. In [9] this cost is reduced by transforming the matrix C into its Jordan form. This lowers the arithmetic complexity to the range $[\frac{2}{3}N \cdot r^3, \frac{8}{3}N \cdot r^3]$ flops, depending on the eigenvalues of C. A further modification of this procedure is that used in the implementation of the code RADAU5 [14], where complex systems are solved by writing them as two real systems.

A different approach is to use an *inner* or *secondary* iterative procedure (such as SOR or splitting methods) for each *outer* or *primary* iteration (8). In general only a fixed number (possibly one) of inner iterations are performed so that one can describe the entire process by means of a single iteration scheme of the form

$$\mathbf{y}^{k+1} = G(\mathbf{y}^k), \qquad G : \Omega \subset H^N \longrightarrow H^N \tag{9}$$

which, starting from an initial guess $\mathbf{y}^0$, generates a sequence $\{\mathbf{y}^k\}$ that is supposed to converge to the solution of (7), say $\mathbf{y}^*$.

Some of such inner-outer iteration schemes were recently proposed by van der Houwen and de Swart [20, 21], where they essentially considered the splitting $C = L + R$ where the matrix L is obtained by the LU factorization of the matrix C, with unit diagonal entries in the matrix U and R is the remainder in the decomposition. With this particular choice of the splitting matrices L and R, no restriction on the stepsize is imposed by the convergence conditions of the inner iteration scheme in the presence of stiff components of the solution. Indeed the dependence of the convergence properties on the problem represents a very important question to be faced when an iterative procedure is adopted for solving the underlying linear systems. Applying the method to the test problem $y' = \lambda y$, the study of this dependence leads to the definition of the region of convergence (see [20]) as the following subset of the complex plane:

$$\Gamma = \left\{ q = h\lambda : \mathbf{y}^k \rightarrow \mathbf{y}^* \text{ as } k \rightarrow \infty \right\}.$$

If $\mathcal{S}$ is the region of A-stability of the implicit method, a desirable property to be required is that $\mathcal{S} \cap \mathbb{C}^- \subset \Gamma \cap \mathbb{C}^-$, since in this case the stepsize may be restricted only by the stability requirements and therefore the general properties of the method are preserved.

The roles of the primary and secondary iterations may be exchanged; that is, one can think of first performing a nonlinear splitting to obtain a convenient structure of the system which is then solved by an appropriate method for nonlinear equations. The most famous nonlinear iterative processes are of course the extensions to nonlinear systems of the best known iterative methods, namely Jacobi, Gauss-Seidel and SOR methods. Convergence results of these schemes may be found in [19].

In our implementation we solve the systems (2) by means of a nonlinear block-splitting process. We consider the general nonlinear $N \cdot r$ problem

$$A\mathbf{y} - hBF(\mathbf{y}) = \boldsymbol{\delta}, \tag{10}$$

where the pair (A, B) identifies a GAM of order s=3,5,7,9, and $\mathbf{y}^*$ denotes its solution (for existence and uniqueness results of the solution of (10) see for example [13]). We then consider the following decomposition of the matrix B:

$$B = L + R \tag{11}$$

where L is chosen to be lower triangular. Consequently the splitting (11) defines the nonlinear iterative process

$$A\mathbf{y}^{k+1} - hLF(\mathbf{y}^{k+1}) = \boldsymbol{\delta} + h(B - L)F(\mathbf{y}^k). \tag{12}$$

Equation (12) still represents a nonlinear system with the property that the i-th unknown $\mathbf{y}_i^{k+1}$ only depends on $\mathbf{y}_j^{k+1}$, $j = 1, \ldots, i-1$, which may be previously computed. Hence we have to solve N systems of dimension r instead of a system of dimension $N \cdot r$.

The i-th block-component of (12) assumes the form

$$-\mathbf{y}_{i-1}^{k+1} + \mathbf{y}_i^{k+1} - hl_{ii}\mathbf{f}(\mathbf{y}_i^{k+1}) - h\sum_{j=1}^{i-1} l_{ij}\mathbf{f}(\mathbf{y}_j^{k+1}) = \boldsymbol{\delta}_i + h\sum_{j=1}^{N}(b_{ij} - l_{ij})\mathbf{f}(\mathbf{y}_j^k). \tag{13}$$

where to simplify the notation we assume $\mathbf{f}(\mathbf{y}_i^k) = \mathbf{f}(t_i, \mathbf{y}_i^k)$.

We obtain an approximation of the unknown $\mathbf{y}_i^{k+1}$ by means of Newton's method. In general it may be not convenient to iterate this inner procecedure to convergence and indeed in our application we found that a single inner iteration performs better. Hence we adopted a one-step splitting-Newton process that can be put in the form (9) where the mapping G has components $\mathbf{g}_i$, $i = 1, \ldots, N$ which are conveniently defined recursively by

$$\begin{aligned}\mathbf{g}_i(\mathbf{y}) = \;& \mathbf{y}_i - (I_r - hl_{ii}J(\mathbf{y}_i))^{-1} \\ & \left(-\mathbf{g}_{i-1}(\mathbf{y}) + \mathbf{y}_i - hl_{ii}\mathbf{f}(\mathbf{y}_i) - h\sum_{j=1}^{i-1} l_{ij}\mathbf{f}(\mathbf{g}_j(\mathbf{y})) - \boldsymbol{\delta}_i - h\sum_{j=1}^{N}(b_{ij} - l_{ij})\mathbf{f}(\mathbf{y}_j)\right),\end{aligned} \tag{14}$$

where I_r is the identity matrix of size r, J is the Jacobian matrix of $\mathbf{f}$ and we made the assumption that $\mathbf{g}_0(\mathbf{y}) = 0$. In the following theorem a convergence result of this scheme is reported (for simplicity in the notation) in the case where the problem (1) is scalar; the generalization to vector problems is straightforward. We observe that each component of $\mathbf{y}^*$ will converge to y_0 when $h \to 0$. Hence it is possible to fix $\bar{h} > 0$ and an open set $\Omega \subset \mathbb{R}^N$ containing $\mathbf{y}^*$ for each $h < \bar{h}$. Furthermore $\rho(M)$ will denote the spectral radius of a square matrix M.

THEOREM 3.1 *Suppose f is continuously differentiable. Then there exist $h_0 > 0$ and an open set $S \subset \Omega$ where, for each $h < h_0$, the mapping G is well defined and $\mathbf{y}^*$ is a point of attraction for the iterations $\mathbf{y}^{k+1} = G(\mathbf{y}^k)$.*

Proof From the regularity of f it follows that an open set $\tilde{S} \subset \Omega$ and $\tilde{h}_0 > 0$ exist such that $\mathbf{y}^* \in \tilde{S}$ for each $h < \tilde{h}_0$ and

$$1 - \tilde{h}_0 l_{ii} f'(y_i) > \eta > 0 \qquad \forall \mathbf{y} \in \tilde{S}. \tag{15}$$

An easy induction proves that g_i is continuously differentiable in $\tilde{S}$ for $h < \tilde{h}_0$ and so is G. We now observe that the matrix

$$M(h, \mathbf{y}^*) = h(A - hLD^*)^{-1}(B - L)D^*, \tag{16}$$

where D^* is the diagonal matrix having $f'(y_i^*)$ as its i-th diagonal entry, is well defined for $h < \tilde{h}_0$ since, for such values of h, the conditions (15) imply the invertibility of $(A - hLD^*)$. Considering that $M(h, \mathbf{y}^*) \to 0$ as $h \to 0$, the assertion will follow if we prove that $G'(\mathbf{y}^*) = M(h, \mathbf{y}^*)$. In this case in fact, one can define $h_0 > 0$ such that $0 < h_0 < \tilde{h}_0$ and

$$\rho(M(h, \mathbf{y}^*)) < 1 \qquad \forall h < h_0 \tag{17}$$

and $S \subset \tilde{S}$ is an open set in which $\mathbf{y}^*$ is attractive.

Since $G(\mathbf{y}^*) = \mathbf{y}^*$ the expression (14) may be recast as

$$\begin{aligned} (1 - hl_{ii}f'(y_i))(g_i(\mathbf{y}) - g_i(\mathbf{y}^*)) &= (1 - hl_{ii}f'(y_i))(y_i - y_i^*) \\ +g_{i-1}(\mathbf{y}) - y_i + hl_{ii}f(y_i) + h\sum_{j=1}^{i-1} l_{ij} f(g_j(\mathbf{y})) &+ \delta_i + h\sum_{j=1}^{N}(b_{ij} - l_{ij})f(y_j), \end{aligned} \tag{18}$$

We now consider the following expansions in neighborhoods of y_j^* and $\mathbf{y}$:

$$\begin{aligned} f(y_j) &= f(y_j^*) + f'(y_j^*)(y_j - y_j^*) + O((y_j - y_j^*)^2), \\ f'(y_i) &= f'(y_i^*) + O(y_i - y_i^*), \\ g_j(\mathbf{y}) &= g_j(\mathbf{y}^*) + O(||\mathbf{y} - \mathbf{y}^*||), \end{aligned}$$

and consequently

$$f(g_j(\mathbf{y})) = f(g_j(\mathbf{y}^*)) + f'(g_j(\mathbf{y}^*))(g_j(\mathbf{y}) - g_j(\mathbf{y}^*)) + O(||\mathbf{y} - \mathbf{y}^*||^2).$$

Taking into account that $g_j(\mathbf{y}^*) = y_j^*$, substituting these expressions in (18) and simplifying yields

$$-(g_{i-1}(\mathbf{y}) - g_{i-1}(\mathbf{y}^*)) + (g_i(\mathbf{y}) - g_i(\mathbf{y}^*)) - h\sum_{j=1}^{i} l_{ij} f'(y_j^*)(g_j(\mathbf{y}) - g_j(\mathbf{y}^*)) =$$
$$y_{i-1}^* + y_i^* + \delta_i + h\sum_{j=1}^{N} b_{ij} f(y_j^*) + h\sum_{j=1}^{N}(b_{ij} - l_{ij}) f'(y_j^*)(y_j - y_j^*) + O(||\mathbf{y} - \mathbf{y}^*||^2).$$

Now the sum of the first four terms in the right hand side is zero, since it represents the i-th component of system (10) and $\mathbf{y}^*$ is its solution. In matrix notation these N equalities may be rewritten as

$$(A - hLD^*)(G(\mathbf{y}) - G(\mathbf{y}^*)) = h(B - L)D^*(\mathbf{y} - \mathbf{y}^*) + Q(\mathbf{y})$$

where $Q(\mathbf{y})$ is a vector function whose components are $O(||\mathbf{y} - \mathbf{y}^*||^2)$. From (16) we have

$$\begin{aligned}\lim_{\mathbf{y}\to\mathbf{y}^*} & \frac{||G(\mathbf{y}) - G(\mathbf{y}^*) - M(h, \mathbf{y}^*)(\mathbf{y} - \mathbf{y}^*)||}{||\mathbf{y} - \mathbf{y}^*||} \\ & = \lim_{\mathbf{y}\to\mathbf{y}^*} \frac{||(A - hLD^*)^{-1} Q(\mathbf{y})||}{||\mathbf{y} - \mathbf{y}^*||} \\ & = 0,\end{aligned}$$

and this completes the proof. □

Obviously in the general case the condition (17) is not sufficient to assure the convergence of any sequence $\{\mathbf{y}^k\}$ until the starting point $\mathbf{y}^0$ is close enough to the solution $\mathbf{y}^*$. This is not the case of linear autonomous problems for which the convergence properties have a global character. Applying the scheme to the test problem $y' = \lambda y$, the matrix M assumes the form

$$M(q) = q(A - qL)^{-1}(B - L), \qquad q = h\lambda \tag{19}$$

and the region of convergence is described by

$$\Gamma = \{q \in \mathbb{C} : \rho(M(q)) < 1\}.$$

Since the considered GAMs are all A-stable we are interested in choosing the matrices L in order to obtain A-convergence. A necessary condition for A-convergence is that the diagonal entries of L must be positive. In this case the function $\rho(M(q))$ turns out to be analytic and symmetric with respect to the real axis in $\mathbb{C}^-$ and A-convergence is equivalent to the requirement

$$\rho^* := \max_{x\in\mathbb{R}^+} \rho(M(ix)) \le 1, \tag{20}$$

where, as usual, i is the imaginary unit. We also consider a relaxation of this definition. The method is said to be $A(\alpha)$-*convergent* if

$$\rho^*(\alpha) := \max_{arg(q)=\pi-\alpha} \rho(M(q)) < 1. \tag{21}$$

Another desirable property (defined in ([20])) to be mentioned is that

$$\rho(q) \rightarrow 0, \qquad \text{as} \quad q \rightarrow \infty. \tag{22}$$

In this case fast convergence is guaranteed in correspondence to large values of $|q|$ in the left half complex plane and consequently the iteration is very suited for stiff problems. At infinity the iteration matrix (19) assumes the form

$$M(\infty) = L^{-1}B - I, \tag{23}$$

and hence, choosing L as the lower triangular matrix obtained by the LU factorization of B one has $\rho(M(\infty)) = 0$.

Considering the equations (13) and the way they are solved, we observe that, for large r, the leading term in the arithmetic complexity of the procedure is $\frac{2}{3}N \cdot r^3$ flops plus N Jacobian evaluations. This is due to the factorization of the $r \times r$ matrices

$$I_r - hl_{ii}J_i, \qquad i = 1, \ldots, N,$$

where J_i is the Jacobian of $\mathbf{f}$ evaluated at $(t_i, \mathbf{y}_i)$. If all the entries l_{ii} are equal and the modified Newton's method is used with $J_i = J_0, i = 1, \ldots, N$, then only one of the above factorizations is needed. The possibility of having a matrix L with constant diagonal entries and simultaneously preserving property (22) was proved in [1], where the authors found that there exists a matrix similar to the matrix C in (7) which has the LU factorization with equal leading diagonal element on the matrix L. This approch is in general efficient on sequential computers if the *inner* and *outer* iteration are used, but has the disadvantage to transform by similarity the linear system of the modified Newton's iteration, and the nonlinear block splitting process (12) cannot be applied.

Looking at (23), all the possible ways of choosing a matrix L satisfying property (22) and defining a splitting for the matrix B are obtained by means of a factorization of the matrix B of the form $B = LU$ where U is a real matrix having all eigenvalues equal to 1.

We are interested in finding matrices L with constant diagonal elements satisfying this factorization: a necessary condition is that L has constant diagonal entries $l_{ii} = \sqrt[s]{\det(B)}$. We found the other entries of L using minimization techniques over the quantities $\rho^*_\infty = \rho(M(\infty))$ and the norm $\|(L^{-1}B)^s\|^{(1/s)}$ whose importance is specified in the sequel.

To show the behaviour of the iterative scheme we compare the properties of the amplification matrix $M_c(q)$ obtained using L with constant diagonal entries and the matrix $M_v(q)$ obtained by performing the LU factorization of the matrix B.

In Table 1 we report the parameters defined in (20), (21) for the matrix M_c and the corresponding parameters for the matrix M_v.

By comparing the values on each row of the table, we see that for high order, although the splittings defined by M_c are not A-convergent, they remain A(α)-convergent with α close

Table 1: parameters (20) and (21) for the matrices M_c and M_v corresponding to GAMs.

		ρ^*		$\rho^*(\pi/2.14)$		$\rho^*(\pi/2.64)$	
Order	s	M_v	M_c	M_v	M_c	M_v	M_c
3	4	0.24	0.26	0.22	0.23	0.16	0.18
5	6	0.50	0.59	0.44	0.53	0.32	0.42
7	8	0.78	2.02	0.70	0.90	0.51	0.70
9	9	1.04	1.36	0.92	1.22	0.68	0.94

to $\pi/2$.

However, the parameter ρ^* is significant to describe only the asymptotic behaviour of powers of the matrices $M_c(q)$ and $M_v(q)$. In place of the parameter (20) and (21), which does not describe the behaviour of the amplification matrices when the number of iterations is small, the following factors can be considered [21] (hereafter, the norm used is the 2-norm):

- the *averaged amplification factors*

$$\begin{aligned} \rho^*_\mu &= \max_{x\in \mathbb{R}^+} \sqrt[\mu]{\|M_c^\mu(\mathrm{i}\,x)\|} \\ \rho^*_\mu(\alpha) &= \max_{arg(q)=\pi-\alpha} \sqrt[\mu]{\|M_c^\mu(q)\|} \qquad \mu = 1,2,\ldots \end{aligned} \tag{24}$$

  which are always greater than ρ^*, $\rho^*(\alpha)$ and approximate ρ^*, $\rho^*(\alpha)$ as $\mu \to \infty$.

- the *nonstiff averaged amplification factor*,

$$\tilde{\rho}_\mu = \sqrt[\mu]{\|(A^{-1}(B-L))^\mu\|}, \tag{25}$$

  which is useful when $q \approx 0$ since in this case $M_c(q) \approx qA^{-1}(B-L)$.

- the *stiff averaged amplification factor*

$$\rho^{(\infty)}_\mu = \lim_{x\to\infty} \sqrt[\mu]{\|M_c^\mu(\mathrm{i}\,x)\|}, \tag{26}$$

  which may not be zero, for finite μ.

The same parameters are defined for $M_v(q)$. In Table 2 we list the above parameters for the GAMs considered in Table 1, for both matrices M_c and M_v. In all cases, it turns out that the iterative procedure defined by M_c has a better nonstiff average amplification factor and a worse stiff average amplification factor than for M_v. But the possibility of performing only one LU-factorization during the iteration allows us to have good results, especially with small tolerances.

Table 2: parameters (24), (25) and (26).

		ρ_1^*		ρ_3^*		ρ_5^*		ρ_9^*	
Order	s	M_v	M_c	M_v	M_c	M_v	M_c	M_v	M_c
3	4	0.26	0.60	0.25	0.35	0.25	0.30	0.24	0.28
5	6	0.76	1.41	0.62	0.96	0.57	0.79	0.54	0.69
7	8	2.16	2.65	1.10	1.53	0.95	1.29	0.87	1.14
9	9	7.84	5.36	1.91	2.20	1.48	1.73	1.25	1.55
		$\rho^*(\pi/2.64)_1$		$\rho^*(\pi/2.64)_3$		$\rho^*(\pi/2.64)_5$		$\rho^*(\pi/2.64)_9$	
Order	s	M_v	M_c	M_v	M_c	M_v	M_c	M_v	M_c
3	4	0.18	0.55	0.16	0.26	0.16	0.21	0.16	0.20
5	6	0.48	1.40	0.35	0.73	0.34	0.55	0.33	0.48
7	8	1.33	2.66	0.64	1.38	0.58	0.92	0.54	0.80
9	9	4.73	5.38	1.18	2.12	0.93	1.32	0.80	1.11
		$\tilde{\rho}_1$		$\tilde{\rho}_2$		$\tilde{\rho}_5$		$\tilde{\rho}_9$	
Order	s	M_v	M_c	M_v	M_c	M_v	M_c	M_v	M_c
3	4	0.35	0.41	0.33	0.31	0.32	0.27	0.31	0.23
5	6	0.76	1.41	0.62	0.96	0.57	0.79	0.64	0.34
7	8	3.14	2.60	1.72	0.73	1.34	0.59	1.08	0.48
9	9	7.20	6.54	2.84	1.17	2.04	0.85	1.53	0.63
		$\rho_1^{(\infty)}$		$\rho_3^{(\infty)}$		$\rho_5^{(\infty)}$		$\rho_9^{(\infty)}$	
Order	s	M_v	M_c	M_v	M_c	M_v	M_c	M_v	M_c
3	4	0.12	0.55	0.12	0.19	0	0.02	0	0.01
5	6	0.45	1.41	0.26	0.71	0.17	0.37	0	0.12
7	8	1.05	2.66	0.52	1.38	0.34	0.81	0	0.31
9	9	2.35	5.38	0.95	2.12	0.54	1.25	0	0.46

4 The stepsize and order variation

Suppose a step has been performed by the order s GAM (A_s^e, B_s^e) and the solution $\mathbf{y}_s$ computed. Without going into technical details we want now to give an account of how the order variation and stepsize selection strategies have been implemented.

First we need to evaluate an approximation, say $\mathbf{e}_s$, of the error $\hat{\mathbf{y}} - \mathbf{y}_s$ and then we may accept the computed solution if the criteria

$$E_s := \max_{1\le j\le N_s} \left(\frac{1}{r} \sum_{i=1}^{r} \left(\frac{|e_{j,i}^s|}{atol + rtol|y_{0,i}|} \right)^2 \right)^{1/2} < 1 \tag{27}$$

is satisfied; *atol* and *rtol* denote the absolute and relative tolerances while $e_{ji}^s,\ i = 1,\dots,r$ are the elements of the j-th component of $\mathbf{e}_s$. If this same method was used to proceed in the integration, the new stepsize (h_{new}) would be defined in terms of the old one (h_{old}) by the rule

$$h_{new} = \frac{h_{old}}{(sf_s E_s)^{\frac{1}{s}}} \tag{28}$$

where sf_s is a suitably chosen safety factor. Since we want to include the possibility of changing the order we need to repeat the above procedure for two GAMs, of order lower and greater than s respectively, applied on the same mesh. We therefore consider the GAM

(A_r^e, B_r^e) of order $r = s - 2$ (which makes sense only if $s > 3$) and dimension $N_r = N_s$ and the GAM (A_q^e, B_q^e) of order q defined as $q = s + 2$ if $s < 9$ and dimension $N_q = N_s$. This choice for the dimension N_q is possible because N_s coincides with the minimum admissible dimension for the order q GAM. We must now wonder what would the errors $\mathbf{e}_r$ and $\mathbf{e}_q$ have been if these two GAMs were performed on the same mesh and consequently what stepsize h_{new} would they have predicted. Obviously for the successive time step we cannot prefer the GAM that predicted the maximum stepsize because at least two important factors must be taken into consideration: the cost associated with the implementation of each method which is an increasing function of the dimension and therefore of the order, and the fact that at parity of the stepsize higher order GAMs cover a wider time interval. Hence once the quantities E_r and E_q defined as in (27) have been evaluated, the index p is chosen in the set $\{r, s, q\}$ in such a way that

$$\frac{C_p}{(sf_pE_p)^{\frac{1}{p}}} = \max\left\{\frac{C_r}{(sf_rE_r)^{\frac{1}{r}}}, \frac{C_s}{(sf_sE_s)^{\frac{1}{s}}}, \frac{C_q}{(sf_qE_q)^{\frac{1}{q}}}\right\}$$

where C_r, C_s and C_q are the cost factors and take into account the above mentioned remarks (see also [10] where a similar approach was adopted). The order p GAM is then selected to continue the integration procedure, a numerical solution $\mathbf{y}_p$ is derived and the whole process is then repeated identifying s with p. In the case when $s = 3$ only the higher order method is considered and likewise when $s = 9$ the order may possibly only decrease.

The technique does not differ, in substance, from any standard strategy that performs the order variation and indeed the delicate question to be faced is how to obtain a cheap evaluation of the errors $\mathbf{e}_s, \mathbf{e}_q$ and $\mathbf{e}_r$: the rest of this section is devoted to the resolution of this problem.

The use of the three GAMs on the current mesh leads to the three systems

$$A^e\mathbf{y}_s^e - hB_s^eF(\mathbf{y}_s^e) = \mathbf{0}, \tag{29.1}$$
$$A^e\mathbf{y}_r^e - hB_r^eF(\mathbf{y}_r^e) = \mathbf{0}, \tag{29.2}$$
$$A^e\mathbf{y}_q^e - hB_q^eF(\mathbf{y}_q^e) = \mathbf{0}, \tag{29.3}$$

where we excluded the subscript on the matrix A^e since it is invariant among the three methods. If $\tilde{\mathbf{y}}_s,\ \tilde{\mathbf{y}}_q,\ \tilde{\mathbf{y}}_r \in H^{N_s}$ are such that

$$||\hat{\mathbf{y}} - \tilde{\mathbf{y}}_s|| = o(h^{s+1}), \quad ||\hat{\mathbf{y}} - \tilde{\mathbf{y}}_q|| = o(h^{q+1}), \quad ||\hat{\mathbf{y}} - \tilde{\mathbf{y}}_r|| = o(h^{r+1}),$$

we estimate the errors $\hat{\mathbf{y}} - \mathbf{y}_s,\ \hat{\mathbf{y}} - \mathbf{y}_q,\ \hat{\mathbf{y}} - \mathbf{y}_r$, by means of the quantities

$$\mathbf{e}_s = \tilde{\mathbf{y}}_s - \mathbf{y}_s, \qquad \mathbf{e}_q = \tilde{\mathbf{y}}_q - \mathbf{y}_q, \qquad \mathbf{e}_r = \tilde{\mathbf{y}}_r - \mathbf{y}_r.$$

The derivation of these estimations is obtained by means of a deferred correction-like procedure. We need first to mention how, starting from the known solution $\mathbf{y}_s^e$ of (29.1), it is possible to obtain, at low cost, approximations of the vectors of local truncation errors $\boldsymbol{\tau}_s,\ \boldsymbol{\tau}_r$ and $\boldsymbol{\tau}_q$ (see also [7]). We consider the identities

$$A^e \hat{\mathbf{y}}^e - hB_s^e F(\hat{\mathbf{y}}^e) = \boldsymbol{\tau}_s, \tag{30.1}$$
$$A^e \hat{\mathbf{y}}^e - hB_r^e F(\hat{\mathbf{y}}^e) = \boldsymbol{\tau}_r, \tag{30.2}$$
$$A^e \hat{\mathbf{y}}^e - hB_q^e F(\hat{\mathbf{y}}^e) = \boldsymbol{\tau}_q. \tag{30.3}$$

Subtracting (30.1) from (30.2) and (30.3) we obtain

$$\begin{aligned} \boldsymbol{\tau}_r - \boldsymbol{\tau}_s &= -h(B_r^e - B_s^e)F(\hat{\mathbf{y}}^e), \\ \boldsymbol{\tau}_q - \boldsymbol{\tau}_s &= -h(B_q^e - B_s^e)F(\hat{\mathbf{y}}^e). \end{aligned} \tag{31}$$

Inserting the expansion

$$F(\hat{\mathbf{y}}^e) = F(\mathbf{y}_s^e) + O(||\hat{\mathbf{y}}^e - \mathbf{y}_s^e||) = F(\mathbf{y}_s^e) + O(h^{s+1})$$

in (31) yields

$$\begin{aligned} \boldsymbol{\tau}_r - h(B_s^e - B_r^e)F(\mathbf{y}_s^e) &= \boldsymbol{\tau}_s + o(h^{s+1}), \\ \boldsymbol{\tau}_s - h(B_q^e - B_s^e)F(\mathbf{y}_s^e) &= \boldsymbol{\tau}_q + o(h^{s+1}). \end{aligned} \tag{32}$$

Hence the vectors $\mathbf{d}_r = h(B_s^e - B_r^e)F(\mathbf{y}_s^e)$ and $\mathbf{d}_s = h(B_q^e - B_s^e)F(\mathbf{y}_s^e)$ represent approximations of $\boldsymbol{\tau}_r$ and $\boldsymbol{\tau}_s$ of order greater than $r+1$ and $s+1$ respectively, since from (32) one has

$$||\boldsymbol{\tau}_r - \mathbf{d}_r|| = O(h^{s+1}) = o(h^{r+1}), \qquad ||\boldsymbol{\tau}_s - \mathbf{d}_s|| = o(h^{s+1}).$$

The determination of a vector $\mathbf{d}_q$ that approximates $\boldsymbol{\tau}_q$ only makes sense when $s < 9$. Looking at the expressions of the vectors $\boldsymbol{\tau}_s$ and $\boldsymbol{\tau}_q$, that is

$$\begin{aligned} \boldsymbol{\tau}_s &= h^{s+1}[c_{s,1}\hat{\mathbf{y}}^{(s+1)}(\boldsymbol{\xi}_1), c_{s,2}\hat{\mathbf{y}}^{(s+1)}(\boldsymbol{\xi}_2), \dots, c_{s,N_s}\hat{\mathbf{y}}^{(s+1)}(\boldsymbol{\xi}_{N_s})]^T, \\ \boldsymbol{\tau}_q &= h^{s+3}[c_{q,1}\hat{\mathbf{y}}^{(s+3)}(\boldsymbol{\eta}_1), c_{q,2}\hat{\mathbf{y}}^{(s+3)}(\boldsymbol{\eta}_2), \dots, c_{q,N_s}\hat{\mathbf{y}}^{(s+3)}(\boldsymbol{\eta}_{N_s})]^T, \end{aligned}$$

we can obtain $\mathbf{d}_q$ by means of second order divided differences. More precisely if $\tilde{\mathbf{d}}_s \in H^{N_s}$ has components $\tilde{\mathbf{d}}_{s,i} = \mathbf{d}_{s,i}/c_{s,i}$, the vector $\mathbf{d}_q$ is defined in terms of its components as follows:

$$\begin{cases} \mathbf{d}_{q,1} = (\tilde{\mathbf{d}}_{s,3} - 2\tilde{\mathbf{d}}_{s,2} + \tilde{\mathbf{d}}_{s,1})c_{q,1}, \\ \mathbf{d}_{q,i} = (\tilde{\mathbf{d}}_{s,i+1} - 2\tilde{\mathbf{d}}_{s,i} + \tilde{\mathbf{d}}_{s,i-1})c_{q,i}, \qquad i = 2, \dots, N_s - 1, \\ \mathbf{d}_{q,N_s} = (\tilde{\mathbf{d}}_{s,N_s} - 2\tilde{\mathbf{d}}_{s,N_s-1} + \tilde{\mathbf{d}}_{s,N_s-2})c_{q,N_s}. \end{cases}$$

It is easily seen that $||\boldsymbol{\tau}_q - \mathbf{d}_q|| = o(h^{q+1})$ and this approximation is particularly good if $\hat{\mathbf{y}}^{(s+1)}(t)$ is a smooth function on $[t_0, t_{N_s}]$.

4.1 Determination of $\mathbf{e}_s$

We start by considering the system

$$A^e \mathbf{y}^e - hB_s^e F(\mathbf{y}^e) = \mathbf{d}_s \tag{33}$$

and prove that its solution has an order of convergence to $\hat{\mathbf{y}}$ greater than $s+1$ (here $\hat{\mathbf{y}}$ will denote the true solution vector passing through $\mathbf{y}_{s,0}$ so that $||\hat{\mathbf{y}}^e - \mathbf{y}_s^e|| = ||\hat{\mathbf{y}} - \mathbf{y}_s||$). Subtracting (33) from (30.1) we obtain

$$A^e(\hat{\mathbf{y}}^e - \mathbf{y}^e) = hB_s^e(F(\hat{\mathbf{y}}^e) - F(\mathbf{y}^e)) + \boldsymbol{\tau}_s - \mathbf{d}_s.$$

If λ is the Lipschitz constant of $\mathbf{f}$ and $h\lambda$ is sufficiently small, it follows that

$$||\hat{\mathbf{y}} - \mathbf{y}|| \leq \frac{||A^{-1}||\ ||\boldsymbol{\tau}_s - \mathbf{d}_s||}{1 - h\lambda||A^{-1}B_s||} = o(h^{s+1}) \tag{34}$$

We now get an approximate solution of the system (33) by performing one iteration of the modified Newton scheme with initial approximation $\mathbf{y}_s$

$$(A - hLD)\mathbf{e}_s = -A^e\mathbf{y}_s + hB_s^eF(\mathbf{y}_s) + \mathbf{d}_s \tag{35}$$

where L can be the same matrix used in the splitting-Newton iteration or for simplicity its diagonal part, D is a block diagonal matrix containing in each block the Jacobian of $\mathbf{f}$ evaluated at $\mathbf{y}_{s,0}$ so that no extra factorization is needed.

Since $\mathbf{y}_s$ is solution of (30.1), the system (35) becomes

$$(A - hLD)\mathbf{e}_s = \mathbf{d}_s \tag{36}$$

and for sufficiently small h

$$\|\mathbf{e}_s\| \leq \frac{||A^{-1}||\ ||\mathbf{d}_s||}{1 - h\lambda||A^{-1}L||} = O(h^{s+1}). \tag{37}$$

We then set $\tilde{\mathbf{y}}_s = \mathbf{e}_s + \mathbf{y}_s$ and prove that this single modified Newton iteration is sufficient to guarantee that $||\hat{\mathbf{y}} - \tilde{\mathbf{y}}_s|| = o(h^{s+1})$. Subtracting (29.1) from (33) we obtain:

$$A(\mathbf{y} - \mathbf{y}_s) - hB_s(F(\mathbf{y}) - F(\mathbf{y}_s)) = \mathbf{d}_s. \tag{38}$$

Repeating the same argument used to derive (34), we see that

$$||\mathbf{y} - \mathbf{y}_s|| = O(h^{s+1}); \tag{39}$$

furthermore we insert in (38) the expansion

$$F(\mathbf{y}) - F(\mathbf{y}_s) = D_s(\mathbf{y} - \mathbf{y}_s) + O(||\mathbf{y} - \mathbf{y}_s||^2) = D_s(\mathbf{y} - \mathbf{y}_s) + o(h^{s+1}),$$

where D_s is the Jacobian of F evaluated at $\mathbf{y}_s$. Consequently (38) becomes

$$(A - hB_sD_s)(\mathbf{y} - \mathbf{y}_s) = \mathbf{d}_s + o(h^{s+1}). \tag{40}$$

Subtracting (36) from (40) and simplifying we obtain

$$A(\mathbf{y} - \tilde{\mathbf{y}}_s) = h\left(B_s D_s(\mathbf{y} - \mathbf{y}_s) - LD(\tilde{\mathbf{y}}_s - \mathbf{y}_s)\right) + o(h^{s+1}).$$

Now using (37) and (39) we get $||\mathbf{y} - \tilde{\mathbf{y}}_s|| = o(h^{s+1})$. Finally we achieve the desired result using this last expression and (34) in the bound

$$||\hat{\mathbf{y}} - \tilde{\mathbf{y}}_s|| \leq ||\hat{\mathbf{y}} - \mathbf{y}|| + ||\mathbf{y} - \tilde{\mathbf{y}}_s||.$$

4.2 Determination of $\mathbf{e}_q$ and $\mathbf{e}_r$

We want to determine the vector $\mathbf{e}_q$ with an analogous procedure applied to the system

$$A^e\mathbf{y}^e - hB_q^eF(\mathbf{y}^e) = \mathbf{d}_q, \tag{41}$$

even if the solution $\mathbf{y}_q$ of (29.3) is not known.

We can get an approximate solution of the system (41) by performing one iteration of the modified Newton scheme with initial approximation $\mathbf{y}_q$

$$(A - hLD)\mathbf{e}_q = -A^e\mathbf{y}_q + hB_q^eF(\mathbf{y}_q) + \mathbf{d}_q$$

where L and D are the same matrices used to find $\mathbf{e}_s$. Since $\mathbf{y}_q$ is solution of (29.3), the previous system is

$$(A - hLD)\mathbf{e}_q = \mathbf{d}_q$$

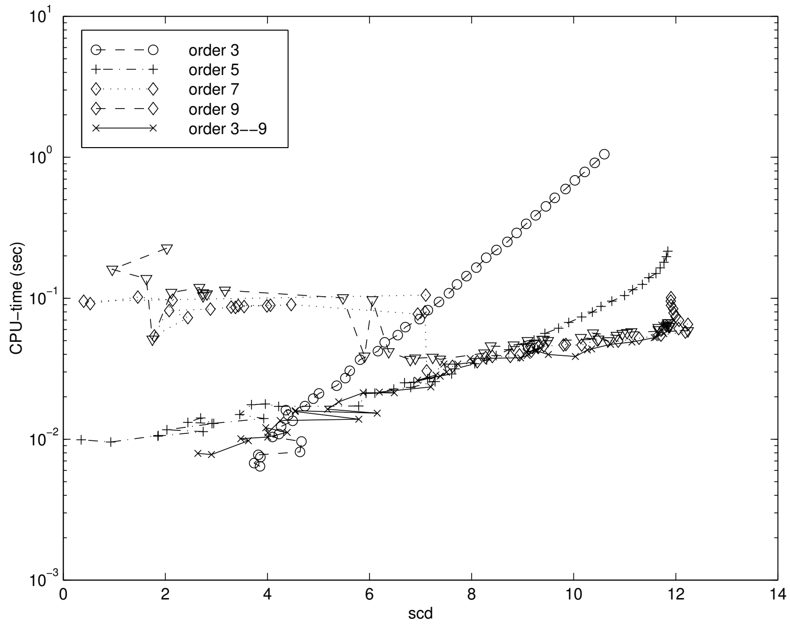
which contains only known terms. We set $\tilde{\mathbf{y}}_q = \mathbf{y}_q + \mathbf{e}_q$ and observe that arguments similar to those used in the above subsection may be considered to prove that $||\hat{\mathbf{y}} - \tilde{\mathbf{y}}_q|| = o(h^{q+1})$.

The technique used to derive $\mathbf{e}_r$ is the analogue of that used to find $\mathbf{e}_q$ provided that system (33) is substituted by

$$A^e\mathbf{y}^e - hB_r^eF(\mathbf{y}^e) = \mathbf{d}_r. \tag{42}$$

To show the effectiveness of the order changing rule we performed five runs of the code on a test problem, namely the Van der Pol problem with parameter $\mu = 1000$ (see the next section). For the first four runs the order has been kept constant and equal to 3,5,7 and 9 respectively, while it may change during the last run. In figure (1) we give a diagram in which the CPU time in seconds needed for the execution of each run is plotted against the effective precision attained by the numerical solution at the end point (scd is the number of significant correct digits). We see that, as expected, low order GAMs are the fastest when low accuracy is required and become the slowest if we need high precision. This behaviour justifies the introduction of a strategy for varying the order. The curve associated to the variable order run is almost everywhere below the others hence proving the efficiency of the analyzed technique.

Figure 1: Van der Pol problem: variable versus fixed order



5 Test problems and numerical results

In this section we present some numerical results to demonstrate the performance of the GAM code on stiff test problems. The GAM code was written in Fortran 77 using double precision. For comparison purposes we also list the results obtained using RADAU5 [15] and MEBDFDAE [11]. The runs were performed on a SunOS 5.5.1 Generic sun4u sparc SUNW Ultra-2 workstation, compiled with optimization f77 -O4. We ran all the ODE test problems reported in [18] and some of the test problems reported in [14]. The codes were run using default input parameters and we changed the input tolerances and the initial stepsize; for MEBDFDAE we use maximum order equal to 8.

We report here some of the most interesting results that show the difference between the tested codes. We note that any numerical comparison is necessarily problem dependent and any conclusion is highly dependent on exactly how we measure the quality of the code. The list of the problems discussed is: RINGMOD (the ring modulator problem with parameter $c_s = 2.0 \times 10^{-12}$ which originates from electrical circuit analysis and presents solution components oscillating at very different frequencies [18]), POLLUTION (a stiff system of 20 nonlinear differential equations [18]), BEAM (the nonlinear elastic beam equation of dimension 80 [14]), ROBER (the Robertson problem represented by a stiff system of 3 nonlinear differential equations [14]), CHEMAKZO (the Chemical Akzo Nobel problem of 6 nonlinear components [18]) and VDPOL (the Van der Pol problem of dimension 2 with parameter $\mu = 10^3$ [14]).

For each numerical example we report the work precision diagram; that is, a range of input tolerances and a range of initial stepsizes were used to produce a plot of the resulting

scd values (the minimum number of significant digits in the components of the numerical solution at the endpoint) against the *CPU* time in seconds needed for the run (see figures 2-7). The format of these diagrams is as in [14]. Naturally each diagram strongly depends on the input tolerances and on the default parameters of the considered codes; the variation of a parameter may considerably change the output. For all problems we fixed $atol = rtol$ and $h_0 = rtol$ while the values of $rtol$ were chosen as follows: $10^{-(3+m/4)}$, $m = 0, \ldots, 40$, in RINGMOD; $10^{-(3+m/4)}$, $m = 0, \ldots, 44$, in POLLUTION; $10^{-(2+m/4)}$, $m = 0, \ldots, 16$, in BEAM; $10^{-(5+m/4)}$, $m = 0, \ldots, 36$, in ROBER; $10^{-(7+m/4)}$, $m = 0, \ldots, 28$, in CHEMAKZO for RADAU5 and MEBDFDAE, and $10^{-(4+m/4)}$, $m = 0, \ldots, 40$, for GAM; $10^{-(2+m/4)}$, $m = 0, \ldots, 48$, in VDPOL.
Since timings on most computers may depend on other processes, the minimum of the CPU times of 100 runs is listed.

We found that the three codes solved all problems efficiently at almost all the tolerances. In term of CPU time it is really impossible to decide which code is the fastest. Regarding the number of LU factorizations, we saw that in general GAM and MEBDFDAE take very few, whereas GAM requires a very large number of function evaluations due to the splitting Newton iterations for the solution of the nonlinear systems. In RINGMOD the codes MEBDFDAE and GAM achieve very similar accuracy with a significantly lower total cost than RADAU5 while the converse is true in ROBER. In POLLUTION and CHEMAKZO the behaviour of the three codes is very similar although RADAU5 stagnates when high precision is required. In the rest of the examples the general observation is that RADAU5 performs better at moderate tolerances while if high accuracy is required MEBDFDAE and GAM become more competitive; this behaviour is probably due to the high order methods used inside GAM and MEBDFDAE.

6 Conclusions

We compared the GAM code with two well known codes for ODEs, RADAU5 and MEBDFDAE. The methods inside all the three codes have good stability properties, and the behavior on some difficult test problems is good. The numerical results obtained by the Generalized Adams Methods are very encouraging; and since GAM uses larger stepsizes than the other codes, hence computing the solution on wider intervals, we expect that a more efficient technique for the solution of the nonlinear systems may considerably improve the efficiency of the code. It is our intention to investigate the properties of the nonlinear systems deriving from the application of GAMs and also to apply the same techniques to other classes of block-BVMs with different stability properties. In particular we will consider the Extended Trapezoidal Rules (ETRs), used for the solution of Hamiltonian problems and the Generalized Backward Differentiation Formulae (GBDFs), used for the solution of Differential Algebraic Equations [7]. This will allow a further comparison with RADAU5 and MEBDFDAE both of which can solve linearly implicit DAEs with index up to 3. The possibility of extending the code for parallel machines will also be considered in the imminent future.

7 Acknowledgments

The authors are very grateful to Prof. D.Trigiante for his helpful comments, suggestions and constructive criticism during the preparation of the GAM code and of this paper. They also whish to thank Dr. A.McCoy for his contribution.

References

[1] P. Amodio, L. Brugnano, *A note on the efficient implementation of implicit methods for ODEs*, Journal of Computational and Applied Mathematics, (to appear).

[2] P. Amodio, F. Mazzia, *A Boundary Value approach to the numerical solution of IVP by multistep methods*, J. of Difference Equations and Applications, vol. 1, 1995, pp. 353-367.

[3] P. Amodio, F. Mazzia,*Boundary Value Methods based on Adams-type methods*, Applied Numerical Mathematics, vol. 18, 1995, pp. 23-35.

[4] K. Brenan, S. Campbell, L. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations.* North-Holland, New York-Amsterdam-London, 1989.

[5] P. N. Brown, G. D. Byrne, A. C. Hindmarsh. *VODE: A variable coefficient ODE solver*, SIAM J. Sci. Stat. Comput., 10, 1989, pp. 1038-1051.

[6] L. Brugnano, D. Trigiante, *High order multistep methods for Boundary Value Problems*, Appl. Numer. Math. , vol. 18, 1995, pp.79-94.

[7] L. Brugnano, D. Trigiante. *Solving Differential Problems by Multistep Initial and Boundary Value Methods*, Gordon & Breach, (in press).

[8] L. Brugnano, D. Trigiante. *On the Potentiality of Sequential and Parallel Codes Based on Extended Trapezoidal Rules (ETRs)*, Appl. Numer. Math. (to appear).

[9] J. C. Butcher. *On the Implementation of Implicit Runge-Kutta Methods*, BIT, 16, 1976, pp. 237-240.

[10] J. C. Butcher, J. R. Cash, M. T. Diamantakis, *DESI Methods for Stiff Initial-Value Problems* ACM Transactions on Mathematical Software, Vol. 22, No. 4, 1996, pp 401-422.

[11] J. R. Cash, *MEBDFDAE* April 1997. Available via WWW at URL http://www.ma.ic.ac.uk/~jcash/IVP_software/finaldae/mebdfdae.f.

[12] J. R. Cash, S. Considine, *An MEBDF Code for Stiff Initial Value Problems*, ACM Transaction on Mathematical Software, Vol 18, No. 2, 1992, pp. 142-155.

[13] G. Di Lena, F. Iavernaro, F. Mazzia, *On the unique solvability of the systems arising from Boundary Value Methods,* Nonlinear Studies, Vol. 4, No 1, 1997, pp 1-12.

[14] E. Hairer, G. Wanner. *Solving Ordinary Differential Equations II. Stiff and Differential–Algebraic Equations*, Springer Series in Computational Mathematics, vol. 14, Springer-Verlag, Berlin, 1996.

[15] E. Hairer, G. Wanner. *RADAU5*, July 1996. Available via WWW at URL ftp://ftp.unige.ch/pub/doc/math/stiff/radau5.f.

[16] A. C. Hindmarsh, *ODEPACK, A systematized collection of ODE solvers*, in Scientific Computing, R. S. Stepleman et al. (Eds.), North Holland, Amsterdam, 1983, pp. 55-64.

[17] F. Iavernaro, F. Mazzia, *Block-Boundary Value Methods for the solution of Ordinary Differential Equations*, submitted.

[18] W. M. Lioen, J. J. B. de Swart, W. A. van der Veen *Test Set for IVP Solvers*, CWI, Department of Mathematics, Amsterdam, Report NM-R9615, 1996.

[19] J. M. Ortega, W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, NewYork, 1970.

[20] P. J. van der Houwen, J. J. B. de Swart. *Triangularly Implicit Iteration Methods for ODE-IVP Solvers*, SIAM J. Sci. Comput., 18, 1997, pp. 41-55.

[21] P. J. van der Houwen, J. J. B. de Swart. *Parallel Linear System Solvers for Runge-Kutta Methods*, Adv. Comput. Math., 7,1-2, 1997, pp. 157-181.

Figure 2: RINGMOD problem

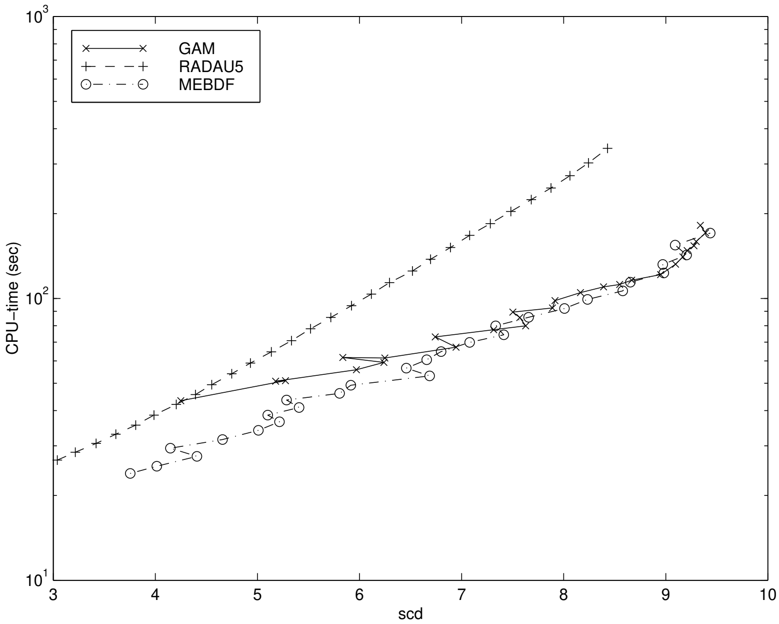


Figure 3: POLLUTION problem

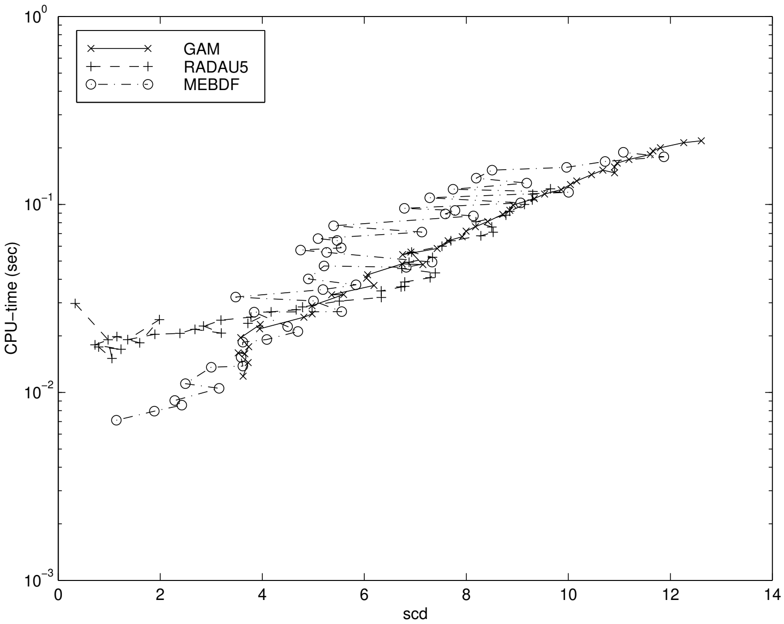


Figure 4: BEAM problem

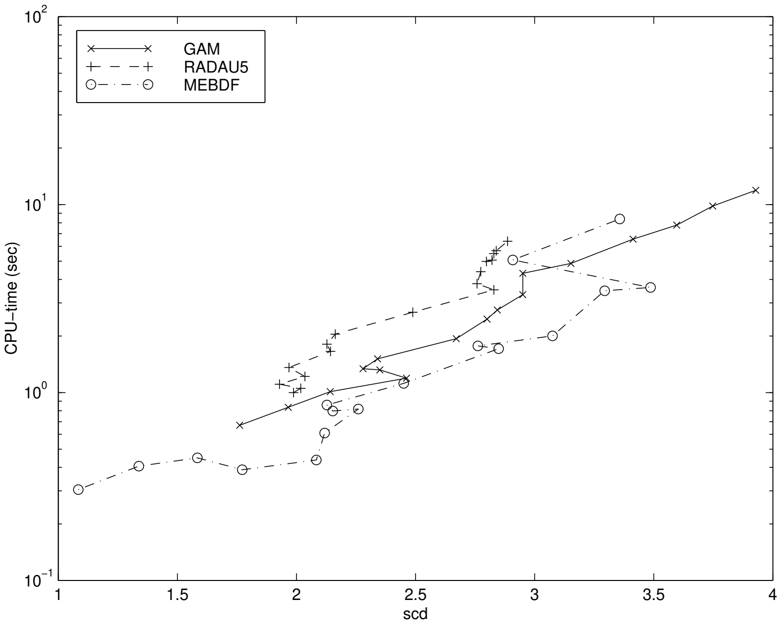


Figure 5: ROBER problem

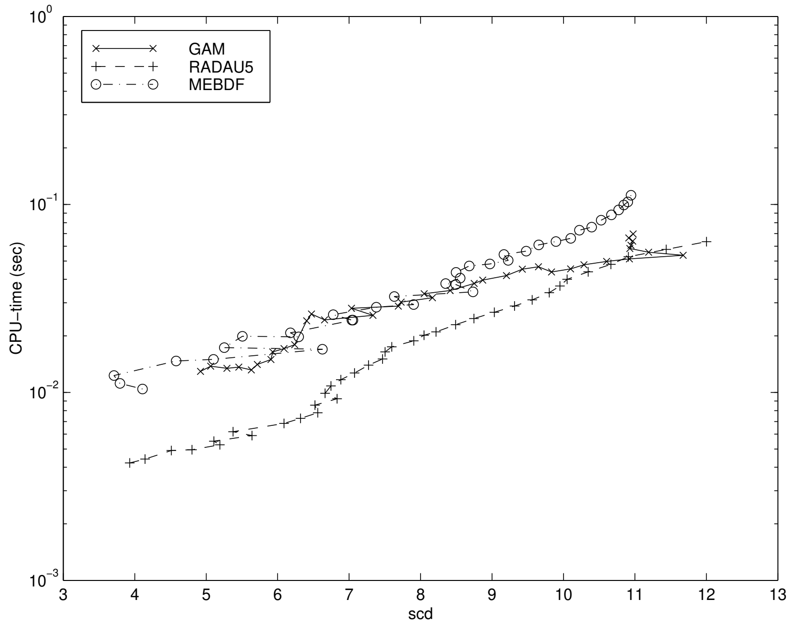


Figure 6: CHEMICAL AKZO NOBEL problem

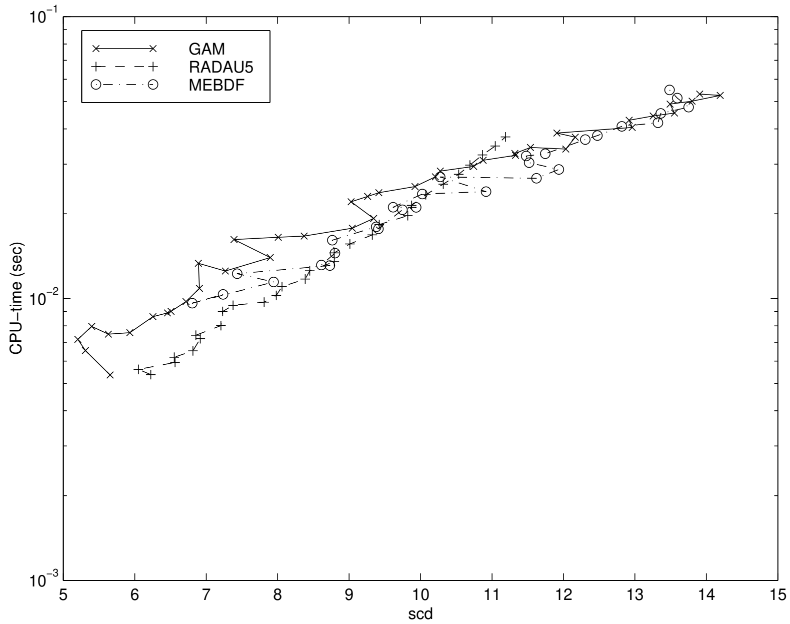


Figure 7: VAN DER POL problem