

Blended Implicit Methods for the Numerical Solution of DAE Problems*

Luigi Brugnano[†] Cecilia Magherini[‡] Filippo Mugnai[§]

Dipartimento di Matematica “U. Dini”
Viale Morgagni 67/A
50134 Firenze, Italy

Abstract

Recently, a new approach for solving the discrete problems, generated by the application of *block implicit methods* for the numerical solution of initial value problems for ODEs, has been devised [5, 6, 9]. This approach is based on the so called *blended* implementation of the methods, giving corresponding *blended implicit methods*. The latter have been implemented in the computational code BiM [7]. Blended implicit methods are here extended to handle the numerical solution of DAE problems, resulting in a straightforward generalization of the basic approach.

Keywords: Ordinary Differential Equations, Stiff Problems, Differential Algebraic Equations, Numerical Methods for ODEs, Numerical Methods for DAEs, Iterative Solution of Algebraic Systems, Blended Implicit Methods.

MSC: 65L05, 65L06, 65L80, 65H10.

1 Introduction

It is widely known that the use of implicit numerical methods is customary when solving *stiff* initial value problems for ODEs,

$$y' = f(t, y), \quad t \in [t_0, T], \quad y(t_0) = y_0 \in \mathbb{R}^m. \quad (1)$$

As a matter of fact, the use of explicit methods is not advisable, because of their weak stability properties. Moreover, usual linear multistep formulae (LMF), when used as initial value methods, do not permit the use of high order formulae,

*Research supported by I.N.d.A.M.

[†]brugnano@math.unifi.it

[‡]magherini@math.unifi.it

[§]f.mugnai@libero.it

due to the well-known *second Dahlquist barrier*. As a consequence, the use of *block implicit methods* [6, 9, 19] is often considered. Such methods generate, at each step of numerical integration, a discrete problem in the form

$$A \otimes I_m \mathbf{y}_n - h_n B \otimes I_m \mathbf{f}_n = \boldsymbol{\eta}_n, \quad n = 0, 1, \dots \quad (2)$$

In the previous formula, I_m denotes the identity matrix of size m , the matrices $A, B \in \mathbb{R}^{r \times r}$ define the method, h_n is the current stepsize, the block vectors

$$\mathbf{y}_n = (y_{n1}, \dots, y_{nr})^T, \quad \mathbf{f}_n = (f_{n1}, \dots, f_{nr})^T,$$

$$f_{nj} = f(t_{nj}, y_{nj}), \quad y_{nj} \approx y(t_{nj}), \quad t_{nj} = t_n + c_j h_n, \quad j = 1, \dots, r,$$

contain the discrete solution, and the vector $\boldsymbol{\eta}_n$ only depends on already known quantities. Hereafter, the two matrices A and B are assumed to be nonsingular, so that the method is an implicit one. Methods falling in this category are the majority of implicit Runge-Kutta methods, a number of General Linear Methods [11, 14, 15], and, more recently, block Boundary Value Methods (BVMs) [8].

The key problem, when using methods in the form (2), is that, at each step of numerical integration, one has to solve a (generally nonlinear) system of rm algebraic equations, in order to compute the discrete (local) solution \mathbf{y}_n . As a matter of fact, the straight application of the simplified Newton iteration would require the factorization of the $rm \times rm$ matrix

$$A \otimes I_m - h_n B \otimes J_n,$$

where J_n denotes the Jacobian matrix of $f(t, y)$ evaluated at the last recently known point. The cost for this is usually too high for this approach to be of practical interest. Consequently, many attempts have been done, across the years, in order to get rid of this drawback. The first successful approach is due to Butcher [10], consisting, in this setting, in having the two matrices A and B essentially diagonalized by the same similarity transformation. By neglecting, for the sake of simplicity, the cost for functional evaluations (which are very problem dependent), such an approach reduces the complexity of the problem from $O((rm)^3)$ to $O(rm^3)$ *flops* (where, as usual, we count as one *flop*, one of the basic algebraic binary operations), and it is successfully implemented in the computational codes RADAU5 and RADAU [15]. An alternative approach consists in defining suitable linear and/or nonlinear splittings [1, 16, 17], i.e., in looking for suitably simple structured matrices A^* and B^* , thus solving the sequence of simpler problems

$$\begin{aligned} A^* \otimes I_m \mathbf{y}_n^{(i)} - h_n B^* \otimes I_m \mathbf{f}_n^{(i)} = \\ (A^* - A) \otimes I_m \mathbf{y}_n^{(i-1)} - h_n (B^* - B) \otimes I_m \mathbf{f}_n^{(i-1)} + \boldsymbol{\eta}_n, \quad i = 1, 2, \dots, \end{aligned} \quad (3)$$

with an obvious meaning of the used notation. Nevertheless, appropriate convergence properties are required for the iteration (3), in order not to nullify the stability properties of the underlying block method. This approach, which has very good potentialities (a reduction of the complexity to $O(m^3)$ *flops* is,

in principle, the target), is implemented in the computational code GAM [18]. However, it is definitely not an easy task to derive suitably simple splitting matrices A^* and B^* defining corresponding iterations having satisfactorily convergence properties.

This task has been recently made much simpler by introducing the *blended implementation* of the block implicit method (2) (we also speak about a *blended implicit method*, in such a case). The basic idea, on which such schemes rely, consists in deriving a numerical method as the combination (*blending*) of two suitable component ones. This approach, at first defined for block BVMs [5], has then been deepened in [6, 9], thus leading to the release of the code BiM [7], which is based on blended implicit methods. In this paper we extend this approach, defined for solving problem (1), to the case of linearly implicit DAE problems, that is, problems in the more general form

$$Ky' = f(t, y), \quad t \in [t_0, T], \quad y(t_0) = y_0 \in \mathbb{R}^m, \quad (4)$$

where the *mass matrix* $K \in \mathbb{R}^{m \times m}$ may be singular. Such problems, indeed, are of great importance in the applications (see, e.g., [4, 13]). For the eventual computation of y_0 , we refer, for example, to [2, 3].

The paper is organized as follows: in Section 2 the main facts about blended implicit methods are recalled for completeness and later reference; the extension of the approach to DAE problems is then described in Section 3; some numerical tests, obtained by using a modification of the code BiM, are reported in Section 4; finally, some concluding remarks are reported in Section 5.

2 Blended Implicit Methods

In order to conveniently introduce *blended implicit methods*, and to carry out a corresponding linear convergence analysis (according to [16, 17]), it is customary to consider the application of the block method (2) to the usual test equation

$$y' = \mu y, \quad y(t_0) = y_0, \quad \operatorname{Re}(\mu) \leq 0. \quad (5)$$

Moreover, for sake of simplicity, let us consider the very first application of the method, since the same arguments apply to each subsequent step of integration. This allows us to avoid, hereafter, the subscript n denoting the integration step. As a consequence, the discrete problem generated by the application of method (2) to problem (5) results in the following system of linear equations,

$$(A - qB) \mathbf{y} = \boldsymbol{\eta}, \quad (6)$$

where, as usual, we have set $q = h\mu$. We shall here consider the so called *type I methods* as described in [6], which are the methods implemented in the code BiM [7]. In more detail, we observe that the nonsingularity of both matrices A and B implies that the discrete problem (6) is equivalent to the following two equations, where $C = A^{-1}B$, $\gamma > 0$, and I is the $r \times r$ identity matrix,

$$(I - qC) \mathbf{y} = A^{-1} \boldsymbol{\eta} \equiv \boldsymbol{\eta}_1, \quad (7)$$

$$\gamma (C^{-1} - qI) \mathbf{y} = \gamma B^{-1} \boldsymbol{\eta} \equiv \boldsymbol{\eta}_2. \quad (8)$$

Let now introduce the *weighting function*

$$\theta(q) = (I - q\gamma I)^{-1}. \quad (9)$$

We observe that:

- $\theta(q)$ is analytical for $q \in \mathbb{C}^-$;
- $\theta(0) = I$;
- $\theta(q) \rightarrow O$, the zero matrix, as $q \rightarrow \infty$.

We then define the *blended implicit method* based on the block method (2), as the method generating the following discrete problem,

$$M(q)\mathbf{y} \equiv (A(q) - qB(q))\mathbf{y} = \hat{\boldsymbol{\eta}}(q), \quad (10)$$

where

$$\begin{aligned} A(q) &= \theta(q)I + (I - \theta(q))\gamma C^{-1}, \\ B(q) &= \theta(q)C + (I - \theta(q))\gamma I, \\ \hat{\boldsymbol{\eta}}(q) &= \theta(q)\boldsymbol{\eta}_1 + (I - \theta(q))\boldsymbol{\eta}_2, \end{aligned}$$

which is still equivalent to (6). We observe that

- $M(q) = I + O(q)$, when $q \approx 0$;
- $M(q) = q(\gamma I + O(q^{-1}))$, as $q \rightarrow \infty$.

This naturally induces the splitting matrix

$$N(q) = I - q\gamma I \equiv \theta(q)^{-1}, \quad (11)$$

since it coincides with $M(q)$ at $q = 0$ and at ∞ , and the corresponding *blended iteration* associated to the blended method (10),

$$N(q)\mathbf{y}^{(i)} = (N(q) - M(q))\mathbf{y}^{(i-1)} + \hat{\boldsymbol{\eta}}(q), \quad i = 1, 2, \dots \quad (12)$$

The iteration will be convergent if and only if the spectral radius, say $\rho(q)$, of the iteration matrix,

$$Z(q) = I - N(q)^{-1}M(q),$$

is not larger than 1. From the previous arguments, we observe that, whatever the choice of the parameter $\gamma > 0$, one has that

- $\rho(0) = 0$,
- $\rho(q) \rightarrow 0$, as $q \rightarrow \infty$.

According to [16, 17], and by using the maximum modulus principle, the iteration (12) is said to be *L-convergent* if the *maximum amplification factor*,

$$\rho_\gamma^* = \max_{x>0} \rho(ix), \quad (13)$$

is smaller than 1, where, as usual, i denotes the imaginary unit. It is worth to emphasize that an L -convergent iteration is highly desirable, if the underlying block method is L -stable. In the above definition, the subscript γ does denote the dependence of this factor from the choice of the positive parameter γ . For the methods implemented in the code BiM, and more in general for all block methods such that:

- the i -th equation of the discrete problem is defined by a r -step LMF of order (at least) r ,
- the spectrum of the matrix $C = A^{-1}B$ coincides (up to a variable scaling) with the reciprocal polynomial at the denominator of the (s, r) Padé approximation to the exponential, $s = r - 2, r - 1, r$,

the following result holds true [6].

Theorem 1 *The choice*

$$\gamma = \min_{\lambda \in \sigma(C)} |\lambda|$$

is the one that minimizes the value of the maximum amplification factor (13).

As matter of fact, for all practical values of r one obtains L -convergent iterations, which are associated to corresponding L -stable methods, when $s < r$. By the way, we observe that Theorem 1 applies to Runge-Kutta Radau IIA methods.

We end this section by observing that, when the blended method is applied to the more general problem (1), then the iteration (10)–(12) becomes

$$\begin{aligned} \delta^{(i)} &= N^{-1} \left(\theta \left((I - \gamma C^{-1}) \otimes I_m \mathbf{y}^{(i-1)} - h(C - \gamma I) \otimes I_m \mathbf{f}^{(i-1)} \right) \right. \\ &\quad \left. + \gamma \left(C^{-1} \otimes I_m \mathbf{y}^{(i-1)} - hI \otimes I_m \mathbf{f}^{(i-1)} \right) - \hat{\boldsymbol{\eta}} \right), \\ \mathbf{y}^{(i)} &= \mathbf{y}^{(i-1)} - \delta^{(i)}, \quad i = 1, 2, \dots, \end{aligned} \quad (14)$$

where, by setting J the Jacobian of $f(t, y)$ evaluated at (t_0, y_0) ,

$$N^{-1} \equiv \theta = I \otimes \Omega^{-1}, \quad \Omega = I_m - h\gamma J. \quad (15)$$

As a consequence, if ℓ iterations are required to obtain convergence, the overall leading cost, to carry out the iteration (14), amounts to:

- 1 Jacobian evaluation,
- 1 factorization for the $m \times m$ matrix Ω ,
- $r\ell$ function evaluations, and
- $2r\ell$ system solvings with the factors of Ω .

For more details, we refer to [7]. An additional, remarkable, feature of the iteration (14)–(15) is the block diagonal structure of the splitting matrix N .

For later reference, we observe that, in the case of the linear ODE

$$y' = Jy + g(t),$$

the iteration (14) induces the iteration matrix

$$Z = C^{-1}(C - \gamma I)^2 \otimes hJ(I_m - h\gamma J)^{-2}, \quad (16)$$

for which the previous linear convergence analysis, summarized by the result of Theorem 1, applies.

3 Extension to DAE problems

We now extend blended implicit methods for solving DAE problems in the form (4). Since we are interested in a linear analysis of convergence for the blended iteration, we confine ourselves to the case of linear equations in the form

$$Ky' = Jy + g(t), \quad (17)$$

where K and J are constant matrices and $g(t)$ is a vector valued function. The following notions (see, for example, [4, 13]) are briefly recalled, for the sake of completeness. First of all, the *matrix pencil*

$$\mu K - J, \quad (18)$$

is associated to equation (17). The pencil is said to be *regular* if its determinant is not identically 0, as a function of μ . Moreover, equation (17) is solvable if and only if the pencil (18) is regular. This will be, therefore, assumed in the sequel. In such a case, the pencil (18) can be cast into its *Kronecker canonical form*,

$$PKQ = \begin{pmatrix} I_d & \\ & H \end{pmatrix} \equiv \hat{K}, \quad PJQ = \begin{pmatrix} G & \\ & I_a \end{pmatrix} \equiv \hat{J}, \quad (19)$$

where P and Q are nonsingular $m \times m$ matrices, $d + a = m$, I_d and I_a are the identity matrices of dimension d and a , respectively, $G \in \mathbb{R}^{d \times d}$, and,

$$H = \begin{pmatrix} H_1 & & \\ & \ddots & \\ & & H_k \end{pmatrix} \in \mathbb{R}^{a \times a}, \quad (20)$$

where

$$H_i = \begin{pmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & 0 \end{pmatrix} \in \mathbb{R}^{\nu_i \times \nu_i}, \quad i = 1, \dots, k.$$

Consequently,

$$H^\nu = O, \quad (21)$$

where the integer

$$\nu = \max_i \nu_i, \quad (22)$$

is known as the *index (of nilpotency)* of the DAE. It is also known that the higher the index, the more difficult the problem.

By means of the Kronecker normal form (19), the problem decouples as follows:

$$x' = Gx + \varphi(t), \quad Hz' = z + \psi(t), \quad (23)$$

where

$$Q^{-1}y \equiv \begin{pmatrix} x \\ z \end{pmatrix}, \quad Pg(t) \equiv \begin{pmatrix} \varphi(t) \\ \psi(t) \end{pmatrix}.$$

With this premise and notations, let us generalize the blended iteration (9)–(12), (14)–(15) for solving the DAE (17). The only differences with what seen in Section 2 amount to the weighting function,

$$\theta = I \otimes K\Omega^{-1}, \quad \Omega = K - h\gamma J, \quad (24)$$

and the splitting matrix,

$$N = I \otimes \Omega, \quad (25)$$

in place of (15). We observe that we get exactly the formulae previously seen in Section 2, in the case $K = I_m$, i.e., when (17) reduces to an ODE.

By taking into account (17), (24)–(25), it is not difficult to see that the blended iteration (14) now becomes,

$$\begin{aligned} \delta^{(i)} &= N^{-1} \left(\theta \left((I - \gamma C^{-1}) \otimes K - h(C - \gamma I) \otimes J \right) \mathbf{y}^{(i-1)} \right. \\ &\quad \left. + \gamma \left(C^{-1} \otimes K - hI \otimes J \right) \mathbf{y}^{(i-1)} - \hat{\boldsymbol{\eta}} \right), \\ \mathbf{y}^{(i)} &= \mathbf{y}^{(i-1)} - \delta^{(i)}, \quad i = 1, 2, \dots \end{aligned} \quad (26)$$

The corresponding iteration matrix is then given by

$$\begin{aligned} Z &= I \otimes I_m - N^{-1} \left(\theta \left((I - \gamma C^{-1}) \otimes K - h(C - \gamma I) \otimes J \right) \right. \\ &\quad \left. + \gamma \left(C^{-1} \otimes K - hI \otimes J \right) \right). \end{aligned} \quad (27)$$

Let now denote by \hat{Z} the matrix obtained from Z with the following formal substitutions (see (19) and (24)–(25)):

$$K \leftarrow \hat{K}, \quad J \leftarrow \hat{J}. \quad (28)$$

The following result then holds true.

Lemma 1 $\hat{Z} = (I \otimes Q^{-1}) Z (I \otimes Q)$.

Proof Let us consider the Kronecker normal form (19) and define

$$\hat{\Omega} = P\Omega Q, \quad \hat{N} = (I \otimes P)N(I \otimes Q), \quad \hat{\theta} = (I \otimes P)\theta(I \otimes P^{-1}). \quad (29)$$

Moreover, one verifies that:

$$\begin{aligned} I \otimes \hat{K} - hC \otimes \hat{J} &= (I \otimes P) (I \otimes K - hC \otimes J) (I \otimes Q), \\ C^{-1} \otimes \hat{K} - hI \otimes \hat{J} &= (I \otimes P) (C^{-1} \otimes K - hI \otimes J) (I \otimes Q). \end{aligned}$$

Thus, by substituting in the corresponding formula (27)-(28) for \hat{Z} , we obtain:

$$\begin{aligned}\hat{Z} &= I \otimes I_m - \hat{N}^{-1} \left(\hat{\theta} \left((I - \gamma C^{-1}) \otimes \hat{K} - h(C - \gamma I) \otimes \hat{J} \right) \right. \\ &\quad \left. + \gamma \left(C^{-1} \otimes \hat{K} - hI \otimes \hat{J} \right) \right) = (I \otimes Q^{-1}) Z (I \otimes Q).\end{aligned}\quad (30)$$

□

Consequently, we can prove the following result.

Lemma 2 *The iteration matrix Z corresponding to (26) is similar to a block diagonal matrix,*

$$\begin{pmatrix} Z_d & \\ & Z_a \end{pmatrix}, \quad (31)$$

where, see (19)-(23),

$$Z_d = C^{-1}(C - \gamma I)^2 \otimes hG(I_d - h\gamma G)^{-2}, \quad (32)$$

$$Z_a = C^{-1}(C - \gamma I)^2 \otimes hH(H - h\gamma I_a)^{-2}. \quad (33)$$

Proof Let define the odd-even block permutation matrix,

$$\Pi = \begin{pmatrix} I_d & & & & \\ & I_d & & & \\ & & \ddots & & \\ & & & I_d & \\ I_a & & & & \\ & I_a & & & \\ & & \ddots & & \\ & & & I_a & \\ & & & & I_a \end{pmatrix}_{rm \times rm}.$$

From (30), one then obtains that $Z \sim \Pi \hat{Z} \Pi^T$. The latter is a 2×2 block diagonal matrix in the form (31), with the diagonal blocks given by (see (24)-(25) and (29))

$$\begin{aligned}Z_d &= I \otimes I_d - I \otimes (I_d - h\gamma G)^{-1} [I \otimes (I_d - h\gamma G)^{-1} (I \otimes I_d \\ &\quad - hC \otimes G) - \gamma^2 hI \otimes G(I - h\gamma G)^{-1} (C^{-1} \otimes I_d - hI \otimes G)],\end{aligned}$$

$$\begin{aligned}Z_a &= I \otimes I_a - I \otimes (H - h\gamma I_a)^{-1} [I \otimes H(H - h\gamma I_a)^{-1} (I \otimes H \\ &\quad - hC \otimes I_a) - \gamma^2 hI \otimes (H - h\gamma I_a)^{-1} (C^{-1} \otimes H - hI \otimes I_a)].\end{aligned}$$

Let us prove that Z_a is given by (33). A similar proof holds for (32). One has:

$$\begin{aligned}Z_a &= I \otimes I_a - I \otimes (H - h\gamma I_a)^{-2} \\ &\quad (I \otimes H^2 - hC \otimes H - \gamma^2 hC^{-1} \otimes H + \gamma^2 h^2 I \otimes I_a)\end{aligned}$$

$$\begin{aligned}
&= I \otimes (H - h\gamma I_a)^{-2} ((-2h\gamma I + hC + \gamma^2 hC^{-1}) \otimes H) \\
&= I \otimes (H - h\gamma I_a)^{-2} ((hC^{-1}(C - \gamma I)^2) \otimes H) \\
&= C^{-1}(C - \gamma I)^2 \otimes hH(H - h\gamma I_a)^{-2}.
\end{aligned}$$

□

Remark 1 *From the previous two lemmas, we obtain that the iteration matrix Z of the blended iteration (26) is similar to a matrix made up of two “pieces”:*

- *one piece corresponding to the ODE in (23), i.e. Z_d ,*
- *the other corresponding to the normalized DAE in the same equation, i.e. Z_a .*

It is evident that for Z_d (compare (32) with (16)) the result of Theorem 1 applies. On the other hand, for Z_a the following result easily follows.

Theorem 2 *The matrix Z_a is nilpotent of index ν .*

Proof Obvious from (33), by taking into account (21)-(22).

□

Remark 2 *The previous statements imply that the convergence properties of the blended iteration (26) are essentially unaffected by the algebraic part of the problem. In fact, they depend only on the differential part Z_d , provided that at least ν iterations are carried out, if ν (see (22)) is the index of the DAE.*

4 Numerical tests

The previous analysis has lead to the development of a preliminary release of the code BiM [7], extended to handle the numerical solution of DAE problems. The code, implementing a variable order-variable stepsize blended implicit method, has been modified, according to what seen in Section 3, in order to allow the integration of problems in the form (4) of index up to 3. In the current version, only the methods of order 4-6-8-10 are considered. Indeed, the use of higher order methods needs to address a few side problems: this research is still in progress. We shall refer to this new version of the code as BiMD.

In this section we report a few numerical test comparing the code BiMD with the codes DASSL [4], GAMD [18], MEBDFDAE and MEBDFI [12], RADAU and RADAU5 [15], on a few test problems taken from the release 2.2 of the *Test Set for IVP Solvers* [20]. The release of each code, which has been used for the tests, is the one available in the above mentioned version of the Test Set [20]. All the above codes are written in Fortran 77 with the exception of the code GAMD which is a Fortran 90 code. The numerical experiments have been done on an IBM SP Power 4 platform, by using the Fortran compilers `xlf` and `xlf90` with the same optimization option (`-O5 -qstrict`) for all the codes. The considered problems are:

- Chemical Akzo Nobel problem (index 1 DAE of dimension 6),
- Transistor amplifier (index 1 DAE of dimension 8),
- Fekete problem (index 2 DAE of dimension 160),
- Water tube system (index 2 DAE of dimension 49),
- Car axis problem (index 3 DAE of dimension 13),
- Andrews' squeezing mechanism (index 3 DAE of dimension 27).

The obtained results are summarized by corresponding *work precision diagrams*, plotted in Figures 1 and 2. According to the standards of the Test Set (see [20]), the diagrams with respect to the *mescd* (mixed error significant correct digits) are plotted for each problem:

$$mescd = -\log_{10} (\|(y - y_{ref}) ./ (\mathbf{atol}/\mathbf{rtol} + |y_{ref}|)\|_{\infty}).$$

In the previous formula, y is the vectors with the computed solution at the end of the integration interval and y_{ref} is the vector containing the corresponding reference solution; $|\cdot|$ and $./$ denote the component-wise absolute value and ratio, respectively; finally, \mathbf{atol} and \mathbf{rtol} are the absolute and relative input tolerances, respectively.

All codes are compared on each test problem, with the only exception of DASSL, which is used only on index 1 DAEs. For each problem, the following parameters have been used,

$$\mathbf{atol} = \mathbf{rtol} = \mathbf{h0} = 10^{-(\alpha+m/\beta)}, \quad m = 0, \dots, m^*,$$

where, \mathbf{atol} and \mathbf{rtol} are the tolerances above specified, and $\mathbf{h0}$ is the initial stepsize (not required for DASSL). The specific values of the parameters α, β, m^* are listed in Table 1, for all considered problems. A number of failed runs occurred for obtaining the data for the work precision diagrams: they are summarized in Table 2, where, for each problem, the codes and the corresponding values of m are listed. Finally, in Tables 3-4 some statistics for each problem are listed (*scd* denotes the computed significant correct digits [20]). From the numerical tests, it seems that the new, preliminary, version of the code BiMD has a comparable performance with the existing codes.

5 Conclusions and future developments

In this paper we derived an extension of *blended implicit methods*, formerly defined for the efficient solution of stiff ODE-IVPs, for the numerical solution of DAE problems.

The generalization of the methods turns out to be very straightforward and, more than this, the corresponding *blended iteration* appears to be essentially unaffected by the algebraic part of the equation. This conclusion has been obtained by considering a linear analysis of convergence using linearly implicit DAEs in the form (17), with constant matrices K and J . This linear analysis of convergence naturally generalizes the corresponding analysis carried out in [6] for the ODE case.

Figure 1: work-precision diagrams.

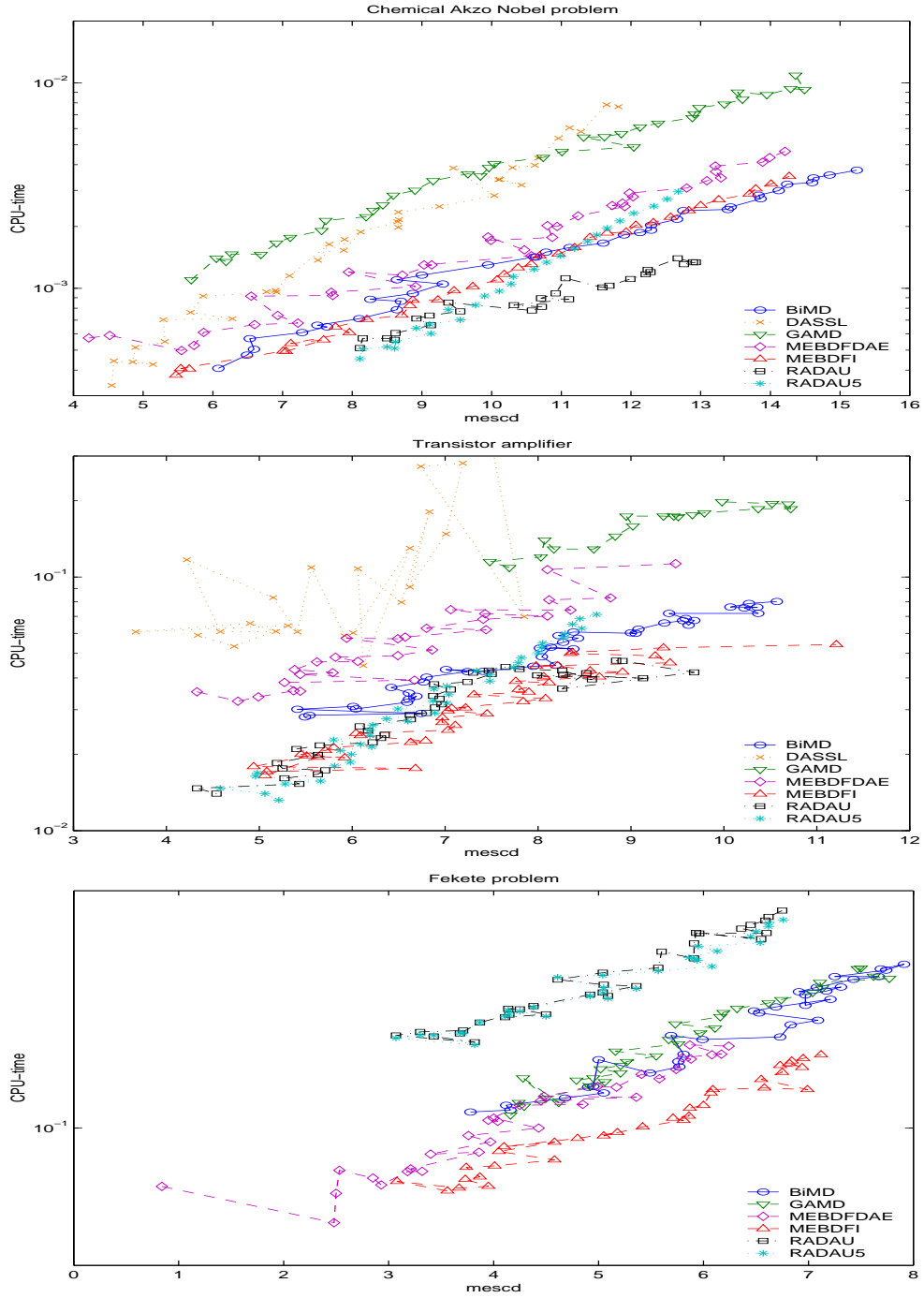


Figure 2: work-precision diagrams.

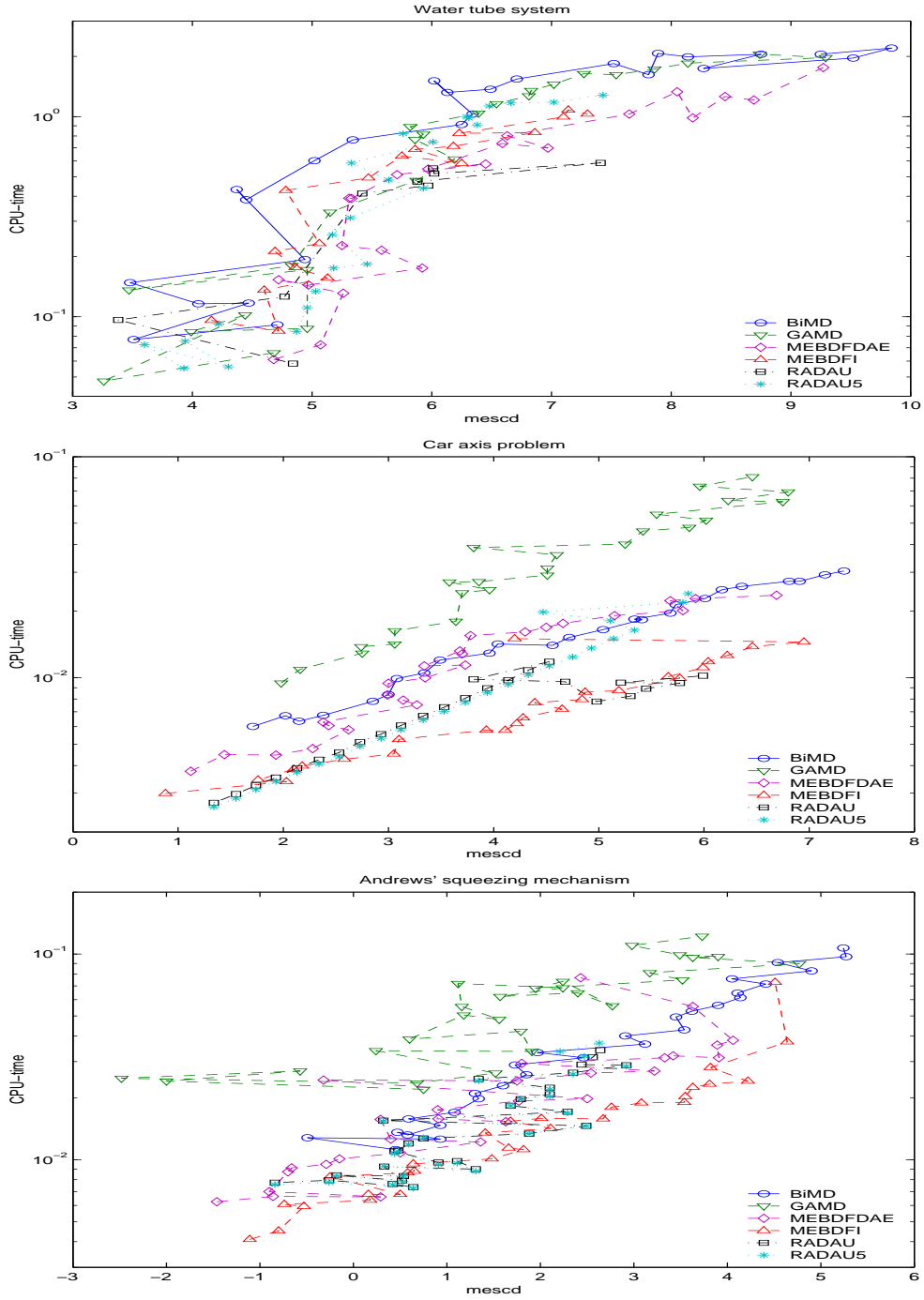


Table 1: parameters used for the numerical tests.

Problem	α	β	m^*
Chemical Akzo Nobel problem	4	4	36
Transistor amplifier	4	8	40
Fekete problem	2	8	32
Water tube system	4	4	24
Car axis problem	4	4	24
Andrews' squeezing mechanism	4	4	28

Table 2: failed runs.

Problem	Code	m
Chemical Akzo Nobel problem	MEBDFI	0
"	RADAU	0-9
"	RADAU5	0-9
Transistor amplifier	GAMD	0-12, 15, 16, 18-20, 23, 25, 27, 29
"	MEBDFDAE	25, 28-33, 35, 37-40
Water tube system	MEBDFDAE	16, 17, 22
"	MEBDFI	18-24
"	RADAU	0, 1, 3, 4, 6, 7, 9-14, 18, 22-24
"	RADAU5	3

The modified blended implicit methods have been used to obtain a preliminary release of the code BiM, called BiMD. Numerical tests carried out on tests problems taken from the *Test Set for IVP Solvers*, show the potentialities of the new code, which well compares with some of the best codes currently available for the numerical solution of DAE problems.

In the near future, we plan to further improve the code BiMD: this will require to address some side problems related to the efficient definition and implementation of the methods. Once this will have been done, we shall made available on the web the new and improved release of the code, at the same URL of the code BiM [7].

References

- [1] P. Amodio, L. Brugnano. A Note on the Efficient Implementation of Implicit Methods for ODEs, *Jour. Comput. Appl. Math.* **87** (1997) 1-9.
- [2] P. Amodio, F. Mazzia. Numerical Solution of Differential Algebraic Equations and Computation of Consistent Initial/Boundary Conditions, *J. Comput. Appl. Math.* **87** (1997) 135-146.
- [3] P. Amodio, F. Mazzia. An Algorithm for the Computation of Consistent Initial Values for Differential-Algebraic Equations, *Numer. Alg.* **19** (1998) 13-23.

Table 3: run statistics.

CHEMICAL AKZO NOBEL PROBLEM									
code	atol rtol h ₀	mescd	scd	steps	accept	#f	#Jac	#LU	CPU-time
BIMD	10 ⁻⁵	7.28	4.72	22	21	331	21	22	6.09 · 10 ⁻⁴
	10 ⁻⁹	11.60	9.57	35	35	936	35	35	1.66 · 10 ⁻³
	10 ⁻¹³	15.24	12.81	60	60	2109	60	60	3.76 · 10 ⁻³
DASSL	10 ⁻⁵	4.89	3.50	92	89	128	14		5.16 · 10 ⁻⁴
	10 ⁻⁹	8.66	7.39	355	351	460	33		1.98 · 10 ⁻³
	10 ⁻¹³	11.82	10.34	1439	1424	1688	74		7.66 · 10 ⁻³
GAMD	10 ⁻⁵	6.69	4.63	16	16	449	16	16	1.46 · 10 ⁻³
	10 ⁻⁹	11.01	9.24	29	29	1403	29	29	4.63 · 10 ⁻³
	10 ⁻¹³	14.36	12.01	57	57	3395	57	57	1.09 · 10 ⁻²
MEBDFDAE	10 ⁻⁵	5.87	4.03	74	74	115	12	12	6.10 · 10 ⁻⁴
	10 ⁻⁹	10.87	8.55	205	205	300	27	27	1.77 · 10 ⁻³
	10 ⁻¹³	14.21	12.67	539	539	764	56	56	4.64 · 10 ⁻³
MEBDFI	10 ⁻⁵	7.01	4.87	74	73	250	12	12	4.95 · 10 ⁻⁴
	10 ⁻⁹	10.57	8.46	201	201	677	24	24	1.31 · 10 ⁻³
	10 ⁻¹³	14.27	12.60	543	542	1792	55	55	3.51 · 10 ⁻³
RADAU	10 ⁻⁵	failed run							
	10 ⁻⁹	10.58	8.11	52	52	539	42	52	7.80 · 10 ⁻⁴
	10 ⁻¹³	12.93	10.61	53	53	1103	41	53	1.34 · 10 ⁻³
RADAU5	10 ⁻⁵	failed run							
	10 ⁻⁹	9.77	7.42	67	67	616	58	67	8.28 · 10 ⁻⁴
	10 ⁻¹³	12.68	10.36	273	273	2520	133	153	2.96 · 10 ⁻³

TRANSISTOR AMPLIFIER									
code	atol rtol h ₀	mescd	scd	steps	accept	#f	#Jac	#LU	CPU-time
BIMD	10 ⁻⁴	5.49	5.25	652	585	9994	585	652	2.81 · 10 ⁻²
	10 ⁻⁷	8.22	7.96	679	613	21325	613	679	5.88 · 10 ⁻²
DASSL	10 ⁻⁴	4.34	2.74	6704	4594	12217	4105		5.90 · 10 ⁻²
	10 ⁻⁷	7.12	6.44	43478	25182	82583	36446		3.96 · 10 ⁻¹
GAMD	10 ⁻⁴	failed run							
	10 ⁻⁷	8.60	7.33	360	318	33117	357	360	1.29 · 10 ⁻¹
MEBDFDAE	10 ⁻⁴	4.33	3.72	1674	1579	3693	294	294	3.53 · 10 ⁻²
	10 ⁻⁷	8.35	8.08	4299	4149	7824	560	560	7.41 · 10 ⁻²
MEBDFI	10 ⁻⁴	5.63	5.55	1598	1501	6060	255	255	1.95 · 10 ⁻²
	10 ⁻⁷	7.31	7.05	3614	3495	13266	430	430	3.44 · 10 ⁻²
RADAU	10 ⁻⁴	4.54	4.26	692	504	6425	500	692	1.40 · 10 ⁻²
	10 ⁻⁷	6.88	6.60	1772	1548	17574	1544	1772	3.78 · 10 ⁻²
RADAU5	10 ⁻⁴	5.21	4.88	690	502	6418	498	690	1.32 · 10 ⁻²
	10 ⁻⁷	6.88	6.60	1776	1550	17642	1545	1776	3.62 · 10 ⁻²

FEKETE PROBLEM									
code	atol rtol h ₀	mescd	scd	steps	accept	#f	#Jac	#LU	CPU-time
BIMD	10 ⁻²	3.78	2.49	29	28	399	28	29	1.15 · 10 ⁻¹
	10 ⁻⁴	7.09	5.84	74	74	1088	74	74	2.57 · 10 ⁻¹
	10 ⁻⁶	7.69	6.23	76	76	2093	76	76	4.03 · 10 ⁻¹
GAMD	10 ⁻²	4.16	2.99	26	24	526	24	26	1.12 · 10 ⁻¹
	10 ⁻⁴	5.76	4.45	38	38	1319	38	38	2.09 · 10 ⁻¹
	10 ⁻⁶	7.50	6.04	59	59	2865	59	59	4.06 · 10 ⁻¹
MEBDFDAE	10 ⁻²	0.84	-0.52	66	63	126	15	15	5.99 · 10 ⁻²
	10 ⁻⁴	3.95	2.64	209	209	334	21	21	1.07 · 10 ⁻¹
	10 ⁻⁶	6.24	4.78	448	448	638	38	38	2.05 · 10 ⁻¹
MEBDFDI	10 ⁻²	3.56	2.10	60	57	192	15	15	5.76 · 10 ⁻²
	10 ⁻⁴	5.81	4.81	218	216	707	23	23	1.07 · 10 ⁻¹
	10 ⁻⁶	7.12	5.66	441	441	1442	30	30	1.90 · 10 ⁻¹
RADAU	10 ⁻²	3.43	1.97	33	30	274	27	32	2.23 · 10 ⁻¹
	10 ⁻⁴	5.36	4.29	61	58	442	54	61	3.47 · 10 ⁻¹
	10 ⁻⁶	6.75	5.34	106	105	963	98	106	6.72 · 10 ⁻¹
RADAU5	10 ⁻²	3.43	1.97	33	30	274	27	32	2.25 · 10 ⁻¹
	10 ⁻⁴	5.36	4.29	61	58	442	54	61	3.39 · 10 ⁻¹
	10 ⁻⁶	6.76	5.31	128	127	919	116	122	6.20 · 10 ⁻¹

Table 4: run statistics.

WATER TUBE SYSTEM									
code	atol rtol h_0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU-time
BIMD	10^{-4}	4.71	1.98	34	30	623	30	34	$9.12 \cdot 10^{-2}$
	10^{-7}	6.02	2.99	407	387	7644	387	407	1.51
GAMD	10^{-4}	4.68	2.41	25	21	498	21	25	$6.60 \cdot 10^{-2}$
	10^{-7}	5.93	3.37	219	197	8042	205	219	$8.17 \cdot 10^{-1}$
MEBDFDAE	10^{-4}	4.68	2.23	123	118	1355	23	23	$6.11 \cdot 10^{-2}$
	10^{-7}	5.97	3.05	1453	1404	11623	190	190	$5.47 \cdot 10^{-1}$
MEBDFDI	10^{-4}	4.16	2.50	139	126	2136	33	33	$9.59 \cdot 10^{-2}$
	10^{-7}	6.18	3.75	1572	1481	15917	213	213	$7.11 \cdot 10^{-1}$
RADAU	10^{-4}	failed run							
	10^{-7}	failed run							
RADAU5	10^{-4}	3.93	1.80	50	45	314	19	40	$5.53 \cdot 10^{-2}$
	10^{-7}	5.32	2.29	283	230	1898	109	223	$3.12 \cdot 10^{-1}$

CAR AXIS PROBLEM									
code	atol rtol h_0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU-time
BIMD	10^{-4}	1.71	0.20	121	121	1658	121	121	$6.01 \cdot 10^{-3}$
	10^{-6}	3.49	2.57	187	183	3507	183	187	$1.20 \cdot 10^{-2}$
	10^{-8}	5.68	3.94	213	210	5909	210	213	$1.96 \cdot 10^{-2}$
GAMD	10^{-4}	1.98	0.39	39	39	2169	39	39	$9.45 \cdot 10^{-3}$
	10^{-6}	3.96	1.50	81	77	5848	77	81	$2.50 \cdot 10^{-2}$
	10^{-8}	5.42	2.76	137	135	10643	135	137	$4.61 \cdot 10^{-2}$
MEBDFDAE	10^{-4}	1.12	-0.51	273	271	754	26	26	$3.77 \cdot 10^{-3}$
	10^{-6}	3.14	1.66	577	576	1578	65	65	$7.91 \cdot 10^{-3}$
	10^{-8}	3.78	1.65	1125	1115	2918	120	120	$1.55 \cdot 10^{-2}$
MEBDFI	10^{-4}	0.88	-0.23	280	278	1246	27	27	$2.99 \cdot 10^{-3}$
	10^{-6}	3.93	1.97	522	520	2277	34	34	$5.79 \cdot 10^{-3}$
	10^{-8}	5.19	3.21	813	812	3282	47	47	$8.73 \cdot 10^{-3}$
RADAU	10^{-4}	1.34	0.19	98	97	850	95	98	$2.71 \cdot 10^{-3}$
	10^{-6}	2.93	1.28	201	200	1742	196	201	$5.55 \cdot 10^{-3}$
	10^{-8}	4.53	2.74	418	417	3783	411	418	$1.18 \cdot 10^{-2}$
RADAU5	10^{-4}	1.34	0.19	98	97	850	95	98	$2.60 \cdot 10^{-3}$
	10^{-6}	2.93	1.28	201	200	1742	196	201	$5.31 \cdot 10^{-3}$
	10^{-8}	4.53	2.74	418	417	3783	411	418	$1.13 \cdot 10^{-2}$

ANDREWS' SQUEEZING MECHANISM									
code	atol rtol h_0	mescd	scd	steps	accept	#f	#Jac	#LU	CPU-time
BIMD	10^{-4}	0.45	4.69	48	42	1042	42	48	$1.12 \cdot 10^{-2}$
	10^{-7}	1.73	4.56	124	122	2701	122	124	$2.89 \cdot 10^{-2}$
	10^{-10}	4.05	7.87	295	294	7126	294	295	$7.57 \cdot 10^{-2}$
GAMD	10^{-4}	-0.03	2.82	84	57	2248	57	84	$2.12 \cdot 10^{-2}$
	10^{-7}	1.56	3.75	131	105	5091	106	131	$4.81 \cdot 10^{-2}$
	10^{-10}	3.63	7.00	219	212	9690	212	219	$9.62 \cdot 10^{-2}$
MEBDFDAE	10^{-4}	-1.46	-0.30	133	117	345	28	28	$6.25 \cdot 10^{-3}$
	10^{-7}	0.91	3.18	347	328	851	49	49	$1.58 \cdot 10^{-2}$
	10^{-10}	3.91	5.92	772	760	1525	96	96	$3.13 \cdot 10^{-2}$
MEBDFI	10^{-4}	-1.11	0.37	118	108	466	23	23	$4.15 \cdot 10^{-3}$
	10^{-7}	1.82	3.87	319	304	1300	42	42	$1.15 \cdot 10^{-2}$
	10^{-10}	3.81	6.13	691	668	2647	77	77	$2.32 \cdot 10^{-2}$
RADAU	10^{-4}	-0.84	1.36	96	56	810	54	96	$7.74 \cdot 10^{-3}$
	10^{-7}	0.48	4.45	114	95	1292	90	114	$1.11 \cdot 10^{-2}$
	10^{-10}	2.35	6.27	261	250	3133	242	261	$2.65 \cdot 10^{-2}$
RADAU5	10^{-4}	-0.84	1.36	96	56	810	54	96	$7.58 \cdot 10^{-3}$
	10^{-7}	0.48	4.45	114	95	1292	90	114	$1.10 \cdot 10^{-2}$
	10^{-10}	2.35	6.27	261	250	3133	242	261	$2.61 \cdot 10^{-2}$

- [4] K. E. Brenan, S. L. Campbell, L. R. Petzold. *Numerical solution of initial-value problems in differential-algebraic equations*. Classics in Applied Mathematics, vol. 14, SIAM, Philadelphia, 1996. Code available at: <http://www.netlib.org/ode/ddassl.f>
- [5] L. Brugnano. Blended Block BVMs (B_3 VMS): A Family of Economical Implicit Methods for ODEs, *Jour. Comput. Appl. Math.* **116** (2000) 41–62.
- [6] L. Brugnano, C. Magherini. Blended Implementation of Block Implicit Methods for ODEs, *Appl. Numer. Math.* **42** (2002) 29–45.
- [7] L. Brugnano, C. Magherini. The BiM Code for the Numerical Solution of ODEs, *Jour. Comput. Appl. Math.* **164-165** (2004) 145–158.
Code available at: <http://math.unifi.it/~brugnano/BiM/index.html>
- [8] L. Brugnano, D. Trigiante. *Solving Differential Problems by Multistep Initial and Boundary Value Methods*, Gordon and Breach, Amsterdam, 1998.
- [9] L. Brugnano, D. Trigiante. Block Implicit Methods for ODEs, in *Recent Trends in Numerical Analysis*, D. Trigiante ed., Nova Science Publ. Inc., New York, 2001, pp. 81–105.
- [10] J. C. Butcher. On the implementation of implicit Runge-Kutta methods, *BIT* **6** (1976) 237–240.
- [11] J. C. Butcher. *Numerical Methods for Ordinary Differential Equations*, John Wiley & Sons, Chichester, 2003.
- [12] J. R. Cash, S. Considine. An MEBDF code for stiff Initial Value Problems, *ACM Trans. Math. Software* **18,2** (1992) 142–158. Codes available at: http://www.ma.ic.ac.uk/~jcash/IVP_software/readme.html
- [13] E. Hairer, C. Lubich, M. Roche. *The Numerical Solution of Differential-Algebraic Systems by Runge-Kutta Methods*, Lecture Notes in Mathematics **1409**, Springer-Verlag, Berlin, 1989.
- [14] E. Hairer, S. P. Nørsett, G. Wanner. *Solving Ordinary Differential Equations I*, 2nd ed., Springer Series in Computational Mathematics, vol. 8, Springer-Verlag, Berlin, 1993.
- [15] E. Hairer, G. Wanner. *Solving Ordinary Differential Equations II, Stiff and Differential-Algebraic Problems*, 2nd ed., Springer Series in Computational Mathematics, vol. 14, Springer-Verlag, Berlin, 1996. Codes available at: <http://www.unige.ch/math/folks/hairer/software.html>
- [16] P. J. van der Houwen, J. J. B. de Swart. Triangularly Implicit Iteration Methods for ODE-IVP Solvers, *SIAM J. Sci. Comput.* **18** (1997) 41–55.
- [17] 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) 157–181.
- [18] F. Iavernaro, F. Mazzia. Solving Ordinary Differential Equations by Generalized Adams Methods: Properties and Implementation Techniques. *Appl. Numer. Math.* **28** (1998) 107–126. Code available at: <http://pitagora.dm.uniba.it/~mazzia/ode/gamd.html>
- [19] H. A. Watts, L. F. Shampine. A-stable Block One-step Methods, *BIT* **12** (1972) 252–266.
- [20] <http://pitagora.dm.uniba.it/~testset/>