A low cost Arnoldi method for large linear initial value problems

Paolo Novati
Dipartimento di Matematica Pura ed Applicata
Universita' degli Studi dell'Aquila
Via Vetoio, Coppito
67010 L'Aquila, Italy
E-mail: novati@univaq.it

Abstract

In this paper we introduce a low cost method for integrating large dimensional linear initial value problems based on the Arnoldi algorithm for matrix functions. We also describe a very efficient step-size control technique and compare a resulting algorithm of order 4 with the standard Matlab solvers on linear stiff problems arising from the discretization of parabolic equations.

1 Introduction

In this paper we consider the linear initial value problem (IVP)

$$\begin{aligned} y' &= -Ay + r(t)v, \quad t > 0, \\ y(0) &= y_0, \end{aligned} \tag{1}$$

where $A \in \mathbb{R}^{N\times N}$ is a large dimensional time-independent matrix, $v \in \mathbb{R}^N$ and $r : [0, +\infty) \to \mathbb{R}$ is a given function. In particular we want to study the practical implementation of a Krylov type method for (1) based on the Arnoldi algorithm, following a resolution scheme based on the decomposition of the forcing term. The resulting explicit one-step method is based on the computation of matrix functions of the type

$$\varphi_p(A) = (-1)^{p+1}A^{-(p+1)}\left[\exp(-A) - (1 - A + \frac{A^2}{2}... + (-1)^p\frac{A^p}{p!}\right], \quad p \geq 0.$$

In recent years, other Krylov type methods have been studied for the solution of large systems of initial value problems (IVPs) especially in the stiff case, where Krylov methods are used by implicit methods in order to accelerate the linear algebra involved in such schemes see e.g. [1], [13] (also with preconditioning [12]). In the linear case, alternative approaches based on the direct

approximation of the matrix exponential (the so-called exponential integrators) by means of Krylov projection methods (using Arnoldi or Lanczos algorithm) have been investigated in [4], [6]. Other exponential integrators, in which the computation of the matrix exponential is performed by means of the polynomial approximation of the exponential function in the complex plane, have been considered in [2], [3], [7], [8], [9]. These exponential integrators are generally quite accurate but can also present some problems, such as the global cost (due as example to the construction of many Krylov subspaces at each step) and the realization of an efficient step-size control scheme.

The method here proposed is essentially an exponential integrator that is able to exploit the charachteristics of (1) and allows to integrate it at the cost of only one Krylov subspace at each step. As we shall see, using the properties of the Arnoldi method for matrix functions, it is also possible to realize a very efficient step-size control technique. Regarding the resolution of stiff IVPs, following the argumentations given in [4], we shall demonstrate that the method behaves like an A-stable method. The aim is to show that for such kind of problems the Arnoldi based method here proposed is an effective alternative to the classical methods.

The paper is structured as follows. In Section 2 we briefly describe the main features of the Arnoldi method for the computation of matrix functions. In Section 3 we introduce our method and in Section 4 we describe an effective step-size control technique. The linear stability of the method is studied in Section 5. Finally, in Section 6 we test a method of order 4 on some large stiff problems comparing the results with those of the Matlab stiff solvers ODE23S and ODE15S.

2 The Arnoldi method for matrix functions

Given a function f and a certain scalar μ, the Arnoldi method for the computation of $f(\mu A)v$ produces approximations of the type

$$f(\mu A)v \approx \beta V_m f(\mu H_m)e_1, \quad m \geq 1, \tag{2}$$

where $\beta = \|v\|_2$ (here and below $\|\cdot\|_2$ denotes the Euclidean norm), $e_1 = (1, 0, ..., 0)^T \in \mathbb{R}^m$, $V_m \in \mathbb{R}^{N\times m}$ is an orthonormal matrix whose columns constitute a basis of the m-th Krylov subspace $K_m(A, v) = span\{v, Av, ..., A^{m-1}v\}$, and H_m is an upper Hessenberg matrix that satisfies

$$H_m = V_m^T A V_m.$$

In the following we shall frequently use the notation

$$\beta V_m f(\mu H_m)e_1 = k_m(f, \mu A)v,$$

where

$$k_m(f, \mu A) := V_m f(\mu H_m)V_m^T. \tag{3}$$

The error arising from the approximation (2) is strictly dependent on the spectral properties of A and on the regularity of f (see.e.g. [5]). However, with respect to μ, it can be bounded as follows

$$\|f(\mu A)v - \beta V_m f(\mu H_m)e_1\| \leq C\mu^m, \tag{4}$$

where $C = C(A, f, m)$ (see e.g. [5], [10], for some evaluations). In the particular case of f analytic in the whole complex plane (as example the exponential function), the numerical method (2) converges superlinearly to $f(\mu A)v$ because $C(A, f, m)^{1/m} \to 0$ as $m \to \infty$.

As well known, since the method generally produces good approximations already for $m << N$, the computational cost is essentially due to the construction of the Krylov subspaces, that is, to the Arnoldi algorithm. This consideration about the computational cost is quite important because the method we are going to introduce exploits an important features of the approximation (2): the computation of $f(\mu A)$ for different values of μ does not influence significantly the total amount of work performed, because the Krylov subspaces remain the same.

3 The numerical metod

For the linear IVP (1), the solution at $t+\delta$, $\delta > 0$, can be expressed in the form

$$y(t+\delta) = \exp(-\delta A)y(t) + \int_0^\delta \exp(-(\delta-\tau)A)r(t+\tau)v. \tag{5}$$

Hence, any quadrature formula for the integral can be used as the basis of a numerical method. Indeed, in [4], two approaches for the numerical evaluation of (5) based on the computation of matrix exponentials and the use of quadrature rules have been proposed. The cost of such methods is strictly dependent on the accuracy of the integration rule, in the sense that high order rules require the computation of many Krylov subspaces at each step. Actually, in the above cited paper an interesting technique based on projection formulas is also presented. Such technique allows rather good accuracy results at the cost of only one or two Krylov subspaces at each step, but, unfortunately, the resulting methods has order 1.

The method we are going to present is also based on the numerical approximation of the integral in (5) but does not require quadrature formulas. Since for $p \geq 0$ we have

$$\int_0^\delta \frac{\exp(-(\delta-\tau)A)\tau^p}{p!} v d\tau = \delta^{p+1}\varphi_p(\delta A)v,$$

where

$$\varphi_p(z) = (-1)^{p+1} z^{-(p+1)} \left[\exp(-z) - (1 - z + \frac{z^2}{2} ... + (-1)^p \frac{z^p}{p!}\right],$$

if the forcing term is such that $r \in C^q$, the solution at $t+\delta$ can be written as

$$y(t+\delta) = \exp(-\delta A)y(t) + \sum_{p=0}^{q} r^{(p)}(t)\delta^{p+1}\varphi_p(\delta A)v + O(\delta^{q+2}).$$

Note that for each $p \geq 0$, the function φ_p is analytic in the whole complex plane. Hence, fixed $s \leq q$ and $m \geq 1$, the idea consists in approximating the solution of (1) with the iteration

$$y_{n+1} = k_m(\exp, -\delta A)y_n + \sum_{p=0}^{s} r^{(p)}(t)\delta^{p+1}k_m(\varphi_p, \delta A)v \tag{6}$$

where k_m is defined by (3). Thus, under the hypotesis that we know the derivatives $r^{(p)}$, $1 \leq p \leq s$, we are able to approximate $y(t+\delta)$ by constructing only two Krylov subspaces, $K_m(A, y_n)$ and $K_m(A, v)$. Since v is fixed during the integration, $K_m(A, v)$ needs to be computed only once during the whole process.

Regarding the accuracy of the method, we can state the following result.

Theorem 1 *Let $r \in C^q$. Fixed $0 \leq s \leq q$ and $m \geq 1$, the method (6) has order $\min(m, s+2) - 1$. Moreover, choosing $m := s+1$ we have a method of order $s = m-1$ whose local error, with respect to an arbitrary vector norm $\|\cdot\|$, is dominated by the quantity $\|[\exp(-\delta A) - k_m(\exp, -\delta A)]\, y_n\|$.*

Proof. By (4), the local error at the point $t+\delta$, that we denote by E_{n+1}, is given by

$$\begin{aligned}
E_{n+1} \quad &: \quad = \left\| \left[\exp(-\delta A)y_n + \sum_{p=0}^{s} r^{(p)}(t)\delta^{p+1}\varphi_p(\delta A)v + O(\delta^{s+2}) \right] - \right. \\
&\qquad \left. \left[k_m(\exp, -\delta A)y_n + \sum_{p=0}^{s} r^{(p)}(t)\delta^{p+1}k_m(\varphi_p, \delta A)v \right] \right\| \\
&\leq \quad \|[\exp(-\delta A) - k_m(\exp, -\delta A)]\, y_n\| + \\
&\qquad \left\| \sum_{p=0}^{s} r^{(p)}(t)\delta^{p+1} \left[\varphi_p(\delta A) - k_m(\varphi_p, \delta A)\right] v \right\| + O(\delta^{s+2}) \\
&\leq \quad C_e\delta^m + \sum_{p=0}^{s} \left| r^{(p)}(t) \right| \delta^{p+1} C_p \delta^m + O(\delta^{s+2}) \\
&= \quad C_e\delta^m + \sum_{p=0}^{s} \left| r^{(p)}(t) \right| C_p \delta^{m+p+1} + O(\delta^{s+2}),
\end{aligned} \tag{7}$$

where C_e and C_p are the constants of (4). Since $p \geq 0$ the method has order $\min(m, s+2) - 1$. Thus, choosing $m := s+1$ the local error is of type $C_e\delta^m + O(\delta^{m+1})$ and is due to $\|[\exp(-\delta A) - k_m(\exp, -\delta A)]\, y_n\|$. ■

If the derivatives of r up to the order s are not available, we are forced to use a suitable numerical approximation. Obviously, it will be important to use an approximation that allows to maintain the order of the method. Let us consider a certain numerical scheme for approximating $r^{(p)}(t)$ by means of $\overline{r}^{(p)}(t)$. The numerical method (6) takes now the form

$$y_{n+1} = k_m(\exp, -\delta A)y_n + \sum_{p=0}^{s} \overline{r}^{(p)}(t)\delta^{p+1}k_m(\varphi_p, \delta A)v, \tag{8}$$

and we can state the following theorem.

Theorem 2 *Let $r \in C^q$. Fixed $0 \leq s \leq q$ and $m \geq 1$, if*

$$\overline{r}^{(p)}(t) = r^{(p)}(t) + O(\delta^{d_p}), \quad d_p \geq 0, \quad p = 1, ..., s, \tag{9}$$

then the method (8) has order $\min(m, d_p + p + 1, s + 2) - 1$, for $p = 1, ..., s$.

Proof. Following the proof of the previous theorem, the local error $\overline{E}_{n+1}$ of (8) satisfies

$$\begin{aligned} \overline{E}_{n+1} &\leq E_{n+1} + \left\| \sum_{p=1}^{s} \left[r^{(p)}(t) - \overline{r}^{(p)}(t) \right] \delta^{p+1} k_m(\varphi_p, \delta A)v \right\| \\ &\leq E_{n+1} + c_1\delta^{d_1+2} + c_2\delta^{d_2+3} + ... + c_s\delta^{d_s+s+1} \end{aligned}$$

where c_p, $p = 1, ..., s$, are suitable constants. Hence, by (7), we easily get the thesis. ■

4 The step-size control

In this section we explain an efficient way to estimate the local error of the method (8) (and also (6)). Assume that the numerical method for approximating the derivatives of r is such that the local error of (8) is given by

$$\overline{E}_{n+1} = \| [\exp(-\delta A) - k_m(\exp, -\delta A)]\, y_n \| + O(\delta^{m+1}) \tag{10}$$

(by Theorem 1, for the method (6) it is sufficient to assume that $m \leq s + 1$). The following result provides a practical estimate for the above quantity.

Proposition 3 *[10]Given an arbitrary vector norm $\|\cdot\|$,*

$$\| [\exp(-\delta A) - k_m(\exp, -\delta A)]\, y_n \| \approx |h_{m+1,m}| \left| e_m^T \varphi_0(\delta H_m) e_1 \right| \|v_{m+1}\| =: e_{n+1}(\delta). \tag{11}$$

Hence, under the assumption (10), the local error can be estimated as follows

$$\overline{E}_{n+1} \approx e_{n+1}(\delta). \tag{12}$$

Now, fixed a certain tolerance $\varepsilon > 0$, assume that δ_n is the step-size accepted for the computation of $y_n \approx y(t)$. Given a certain initial guess δ, since the method has order $m-1$, we know that $\overline{E}_{n+1} \approx c\delta^m$. If δ is such that $\overline{E}_{n+1} \leq \varepsilon$ we define $\delta_{n+1} := \delta$ and compute y_{n+1}. If $\overline{E}_{n+1} > \varepsilon$ we must reduce the step-size. Using the estimate (12) we can easily estimate c, i.e.,

$$c \approx \frac{e_{n+1}(\delta)}{\delta^m}.$$

In this way, fixed a certain constant γ, $0 < \gamma < 1$, we can get the new candidate δ^* by solving

$$\frac{e_{n+1}(\delta)}{\delta^m}\delta^{*m} \leq \gamma\varepsilon,$$

that is,

$$\delta^* = \delta \sqrt[m]{\frac{\gamma\varepsilon}{e_{n+1}(\delta)}}. \tag{13}$$

The procedure is repeated until the required accuracy is reached and we define $\delta_{n+1} := \delta^*$. The guess for the next step-size is then defined as

$$\delta := \delta_{n+1} \sqrt[m]{\frac{\gamma\varepsilon}{e_{n+1}(\delta_{n+1})}}.$$

As in many other step-size control schemes, the quantity γ has been introduced in order to protect the procedure from rough approximations.

Remark 4 *Up to now, the argumentations about the local error have been given with respect to an arbitrary vector norm. Anyway, in the numerical experiments of Section 6 we always use the estimate (11) with respect to the maximum norm* $\|\cdot\|_\infty$.

It is fundamental to observe that even if we are forced to compute many times the quantity $e_{n+1}(\delta)$ before getting an acceptable step, the cost of this computation is negligible because we have only to compute many times the matrix function $\varphi_0(\delta H_m)$ with the same Krylov subspace. In other words, the step-size control does not produce additional work, because we have a good estimation of the local error without computing explicitly any approximation of $y(t+h)$ (as would be necessary employing the Richardson extrapolation or an embedded method).

5 Linear stability

By (3), the method (8) (or (6)) can be written in the form

$$y_{n+1} = V_m \exp(-\delta H_m) V_m^T y_n + g_{n+1},$$

where g_{n+1} is essentially an approximation of the integral in (5). Since the computation of g_{n+1} does not involves y_n in order to investigate the linear stability it is sufficient to analyze the properties of the operator $V_m \exp(-\delta H_m) V_m^T$.

Given a certain matrix B, let $\mu(B)$ be the associated logarithmic norm, defined by (with respect the 2-norm)

$$\mu(B) := \lim_{h \to 0^+} \frac{\|I + hB\|_2 - 1}{h}.$$

We have the following result.

Proposition 5 *[4]If A is positive real, i.e., $x^T Ax > 0$ for each $0 \neq x \in \mathbb{R}^N$, then*

$$\mu(-H_m) \leq \mu(-A) \leq 0.$$

Using the above result, since V_m has orthonormal columns we have

$$\begin{aligned} \left\|V_m \exp(-\delta H_m) V_m^T\right\|_2 &\leq \|\exp(-\delta H_m)\|_2 \\ &\leq \exp(\mu(-\delta H_m)) \\ &\leq \exp(\mu(-\delta A)) \\ &\leq 1 \end{aligned}$$

that proves the stability of the method.

6 Numerical experiments with a method of order 4

In this section we want to show the performance of a method (8) of order 4. In order to build such a method, for the approximation of the derivatives of r we employ the standard central differences formula

$$r'(t) \approx \frac{r(t+h) - r(t-h)}{2h}.$$

Fixed $s > 0$, applying iteratively the above formula with step $h = \delta^d$ for a certain $d \geq 0$, we get approximations $\overline{r}^{(p)}(t)$, $1 \leq p \leq s$, such that

$$\overline{r}^{(p)}(t) = r^{(p)}(t) + O(\delta^{2d}).$$

Hence, by Theorem 2 the local error takes now the form

$$\overline{E}_{n+1} \leq E_{n+1} + C\delta^{2+2d} + O(\delta^{4+2d}),$$

and the order of the method is $\min(m, 2 + 2d, s + 2) - 1$. In order that the local error is dominated by $\|[\exp(-\delta A) - k_m(\exp, -\delta A)]\, y_n\|_\infty$ we must chose $m \leq \min(1 + 2d, s + 1)$. In this way, fixed $s = 4$ and $d = 2$, with $m := 5$ we get a method of order 4.

Using this kind of approximation for the derivatives, we can reduce furtherly the cost of the method (8) substituting $k_m(\varphi_p, \delta A)$ with $k_{m-p}(\varphi_p, \delta A)$. Cleary,

such modification does not change the order of the method that, with the above choices, is now given by

$$y_{n+1} = k_5(\exp, -\delta A)y_n + \sum_{p=0}^{4} \overline{r}^{(p)}(t)\delta^{p+1} k_{5-p}(\varphi_p, \delta A)v. \tag{14}$$

We call ARN4 the method (14).

In the numerical experiments here presented, we compare ARN4 with the stiff solvers implemented in the Matlab codes ODE23S and ODE15S (see [11] for details) on stiff linear IVPs whose matrix arises from the discretization of k-dimensional ($k = 2, 3$) second order partial differential operators of the type

$$-\Delta + \tau_1 \frac{\partial}{\partial x} + \tau_2 \frac{\partial}{\partial y}, \qquad \tau_1, \tau_2 \in \mathbb{R}, \tag{15}$$

where Δ denotes the k-dimensional Laplacian. Dirichlet boundary conditions on the square $(0,1)^k$ are considered. We discretize using central differences with uniform meshsize $h = 1/(n+1)$ along each direction, getting a matrix A of order $N = n^k$.

The comparison is made with respect to the cost of the methods, measured in terms of number of inner products of order N (this choice allows to maintain a closed relation with the dimension of the problem), taking into account of the sparsity pattern of A.

In this sense, since m iterations of the Arnoldi algorithm cost about m matrix-vector multiplication and $m(m+1)/2$ inner products, looking at (14) it is easy to see that s steps of ARN4 cost about $5(s+1)$ matrix-vector multiplication and $15(s+1)$ inner products. Moreover, the computation of $k_5(\exp, -\delta A)y_n$ and $k_{5-p}(\varphi_p, \delta A)v$, $p = 0, ..., 4$, requires additional 20 inner products at each step (see (2)). Hence, we have that the total amount of work is approximatively $5(s+1)$ matrix-vector multiplication and $35s + 15$ inner products. If A arises from the discretization of (15) as explained above, for $k = 2$ an application of A costs about 5 inner products, whereas for $k = 3$, it costs about 7 inner products. Hence, summing the previous quantity, s steps of ARN4 cost about $60s + 40$ inner products for $k = 2$ and $70s + 50$ inner products for $k = 3$.

Regarding the codes ODE23S and ODE15S, their cost is essentially due to a certain number of matrix-vector multiplication (because of the function evaluations), a certain number of LU factorizations of matrices of the type $I - \delta A$, and a related number of triangular systems. We want to express the costs of such computations in terms of inner products of order N. For $k = 2$, the bandwidth of A is $n = \sqrt{N}$, the LU factorization costs about N^2 scalar multiplications, and therefore as much as N inner products. The corresponding triangular systems cost about Nn scalar multiplication and so $\sqrt{N}$ inner products. For $k = 3$, the bandwidth of A is $n^2 = \sqrt[3]{N^2}$, the LU factorization costs about $Nn^4 = N^{7/3}$ scalar multiplications, and therefore as much as $N^{4/3}$ inner products. The corresponding triangular systems cost about Nn^2 scalar multiplications and so $\sqrt[3]{N^2}$ inner products.

The following tables contain the results relative to five numerical experiments. For such tables we used the following abbreviations: AS is the number of accepted steps, FA is the number of failed attempts, MV is the number of matrix-vector multiplications, SP is the number of effective inner products, LU is the number of LU factorizations, TS is the number of triangular systems and, finally, TOT is the total amount of work in term of inner products.

In all experiments the vector v of the forcing term is $v = (1, ..., 1)^T$, whereas the initial condition is given by $y_0 = (1, ..., 1)^T$. The constant γ for the step-size control of ARN4 is fixed equal to 0.5.

Problem 1

In this problem we consider the case $k = 2$, $n = 30$, and $\tau_1 = 20$, $\tau_2 = 0$. The resulting matrix has order $N = 900$. Moreover, we define $r(t) = 50\sin(50t)$. We integrate the corresponding IVP from 0 to 1. The tolerance ε for the step-size control is $\varepsilon = 10^{-2}$. Regarding the parameters *AbsTol* and *RelTol* of the Matlab solvers ODE23S and ODE15S (see [11]) we always define $AbsTol = RelTol = \varepsilon$.

	ODE23S	ODE15S	ARN4
AS	122	128	612
FA	13	13	4
MV	392	282	3065
SP	-	-	21435
LU	135	38	-
TS	810	564	-
TOT	**147760**	**52530**	**36760**

Table 1

Problem 2

As in the previous problem we define $k = 2$, $n = 30$, but now we choose $\tau_1 = \tau_2 = 0$ so that the resulting matrix A is symmetric negative definite. For the forcing term we define $r(t) = -\exp(-t)\cos(t)$. The corresponding IVP is integrated from 0 to 10 with the tolerance $\varepsilon = 10^{-2}$.

	ODE23S	ODE15S	ARN4
AS	32	56	230
FA	0	0	5
MV	96	110	1155
SP	-	-	8065
LU	32	14	-
TS	192	220	-
TOT	**35040**	**19750**	**13840**

Table 2

Problem 3

In this problem we have $k = 3$, $n = 10$, and $\tau_1 = \tau_2 = 0$. The resulting matrix A is symmetric negative definite of order $N = 1000$. We define $r(t) = \exp(-t)\sin(t)$. We integrate the corresponding IVP from 0 to 10 with $\varepsilon = 10^{-3}$.

	ODE23S	ODE15S	ARN4
AS	45	67	135
FA	1	0	0
MV	137	134	680
SP	-	-	4740
LU	46	16	-
TS	276	268	-
TOT	**488559**	**187738**	**9500**

Table 3

Problem 4

In this problem the matrix A is the same of the previous one. The only change regards the forcing term, that is defined by $r(t) = \exp(-0.1t)\cos(50t)$. The interval of integration is now $[0, 5]$ with $\varepsilon = 10^{-3}$.

	ODE23S	ODE15S	ARN4
AS	284	408	313
FA	47	80	69
MV	946	976	1570
SP	-	-	10970
LU	331	160	-
TS	1986	1952	-
TOT	**3515222**	**1802032**	**21960**

Table 4

Problem 5

In this problem we consider a 3-dimensional but non-symmetric case, in which $n = 10$, $\tau_1 = 10$ and $\tau_2 = 5$. We define $r(t) = \exp(-5t)$ and integrate the corresponding IVP from 0 to 10 with $\varepsilon = 10^{-3}$.

	ODE23S	ODE15S	ARN4
AS	47	69	95
FA	0	0	7
MV	141	138	480
SP	-	-	3340
LU	47	16	-
TS	282	276	-
TOT	**499187**	**188566**	**6700**

Table 5

The numerical experiments here presented do not need many comments. Even when ARN4 is clearly disadvantageous with respect to the other two codes in terms of number of steps performed (as in Problems 1 and 2), the very low cost of this method makes it competitive with the others. On the other side, the Problems 3-5 (and many other numerical experiments) relative to 3-dimensional problems show the effectiveness of the method either in the case of highly varying forcing term (see Problem 4) or in the case of smooth forcing term (see Problems

3 and 5). The main reason is that the method is able to exploit the very special features of the Arnoldi algorithm and of the corresponding method for matrix functions so that the total cost is maintained very low. Obviously, this makes the method attractive for large problems.

Looking at the tables above we can also observe that the number of failed attempts of ARN4 is quite low. This confirms the effeciency of the step-size control procedure adopted. Actually, since the cost of a failed attempt is negligible (the estimation of the local error requires only the computation of $\varphi_0(\delta H_m)$, see (11)), in order to reduce the number effective steps we can also choose $0.5 < \gamma < 1$.

References

[1] G.D.Byrne, George D.Pragmatic experiments with Krylov methods in the stiff ODE setting. (English)Computational ordinary differential equations, Proc. Conf., London/UK 1989, Inst. Math. Appl. Conf. Ser., New Ser. **39** (1992), 323-356.

[2] L.Bergamaschi, M.Vianello, Efficient computation of the exponential operator for large, sparse, symmetric matrices, Numer. Linear Algebra Appl. **7** (2000), 27-45.

[3] L. Bergamaschi, M. Caliari, M. Vianello, Efficient approximation of the exponential operator for discrete 2D advection-diffusion problems. Numer. Linear Algebra Appl. **10** (2003), 271-289.

[4] E.Gallopoulos, Y.Saad, Efficient solution of parabolic equations by Krylov approximation methods, SIAM Sci. Stat. Comput. **13** (1992)**,** 1236-1264.

[5] M.Hochbruck, C.Lubich, On Krylov subspace approximation to the matrix exponential operator, SIAM J. Numer. Anal. **34** (1995), 1911-1925.

[6] M.Hochbruck, C.Lubich, H.Selhofer: Exponential integrators for large systems of differential equations, SIAM J. Sci. Comput. **19** (1998), 1552-1574.

[7] I.Moret, P.Novati, An interpolatory approximation of the matrix exponential based on Faber polynomials, Journal C.A.M. **131** (2001), 361-380.

[8] P.Novati, Solving linear initial valuue problems by Faber polynomials, Numer. Linear Algebra Appl., **10** (2003), 247-270.

[9] P. Novati, A method based on Fejèr points for the computation of functions of nonsymmetric matrices. Appl. Numer. Math. **44** (2003), 201-224.

[10] Y.Saad, Analysis of some Krylov subspace approximations to the matrix exponential operator, SIAM J. Numer. Anal. **29** (1992), 209-228.

[11] L.F.Shampine, M.W.Reichelt, The Matlab ODE suite, SIAM J. Sci. Comput., **18** (1997), 1-22.

[12] H.Podhaisky, R.Weiner, B.A.Schmitt, Numerical experiments with Krylov integrators, Appl. Numer. Math. **28** (1998), 413-425.

[13] H.Podhaisky, R.Weiner, B.A.Schmitt, ROWMAP – a ROW-code with Krylov techniques for large stiff ODEs, Appl. Numer. Math. **25** (1997), 303-319.