Ibáñez González, JJ.; Hernández García, V.; Ruíz Martínez, PA.; Arias ., E. (2011). A piecewise-linearized algorithm based on the Krylov subspace for solving stiff ODEs. Journal of Computational and Applied Mathematics. 235(7):1798-1804. doi:10.1016/j.cam.2010.07.012.

A Piecewise-linearized Algorithm based on Krylov Subspace for solving stiff ODEs*

J. Javier Ibáñez[1], Vicente Hernández[1], Pedro A. Ruiz[1] and Enrique Arias[2]

[1] *Instituto de Instrumentación en Imagen Molecular (I3M), Universidad Politécnica Valencia, Camino de Vera s/n, 46022-Valencia (Spain)*

[2] *Albacete Research Institute of Informatics, University of Castilla-La Mancha, Avda. España s/n, 02071-Albacete (Spain)*

emails: jjibanez@dsic.upv.es, vhernand@dsic.upv.es, pruiz@dsic.upv.es, earias@dsi.uclm.es

Abstract

Numerical methods for solving Ordinary Differential Equations (ODEs) have received considerable attention in recent years. In this paper a piecewise-linearized algorithm based on Krylov subspaces for solving Initial Value Problems (IVPs) is proposed. MATLAB versions for autonomous and non-autonomous ODEs of this algorithm have been implemented. These implementations have been compared with other piecewise-linearized algorithms based on Padé approximants, recently developed by the authors of this paper, comparing both precision and computational costs in equality of conditions. Four case studies have been used in the tests that come from biology and chemical kinetics stiff problems. Experimental results show the advantages of the proposed algorithms, especially when the dimension is increased in stiff problems.

Key words: Initial Value Problem (IVP), Ordinary Differential Equation (ODE), Linear Differential Equation (LDE), Piecewise-linearized methods, Padé approximants, Krylov subspace

1 Introduction

Many scientific and engineering problems are described by ODEs where the analytic solution is unknown. In recent years many review articles and books have appeared on numerical methods for integrating stiff ODEs. Stiff problems are very common problems in many fields of the applied sciences: control theory, biology, chemical kinetics, electronic circuit theory, fluids, etc. There exist numerous one step algorithms for solving stiff ODEs based on the implicit Runge-Kutta methods [1, 2, 3]. Another popular

*This work has been supported by the Spanish CICYT project CGL2007-66440-C04-03.

family of algorithms for solving these problems are the multistep algorithms based on the BDF method [4, 5, 6, 7]. In this paper we have developed a one step method based on a piecewise-linearized method [8]. These methods solve an IVP by approximating the right hand side of the corresponding ODE by a Taylor polynomial of degree one. The resulting approximation can be integrated analytically to obtain the solution in each subinterval and yields the exact solution for linear problems. In [8, 9] an exhaustive study of this method is introduced. The proposed method requires a non-singular Jacobian matrix on each subinterval.

In [10] the authors presented a piecewise-linearized method for solving ODEs. This method uses a theorem proved in that article, which enables the approximate solution to be computed at each time step by a block-oriented approach based on diagonal Padé approximations. In this work another approach based on the piecewise-linearized method is introduced. In this case, the matrix-vector product $e^A v$, which appears in these methods, is computed by a Krylov subspace approach. Computational cost and precision of algorithms are compared in equal conditions. The paper is structured as follows. The new approach for solving ODEs based on the Krylov subspace approach is presented in Section 2. The experimental results are shown in Section 3. Finally, conclusions and future expectations are given in Section 4.

2 A piecewise-linearized algorithm for solving ODEs based on the Krylov subspace approach

In [10] the authors presented a piecewise-linearized method for solving ODEs, based on the following theorem which enables the approximate solution to be computed at each time step by a block-oriented approach based on diagonal Padé approximations.

Theorem 1 ([10]) *Let*

$$\dot{x}(t) = f(t, x(t)), t \in [t_0, t_f], \tag{1}$$

be an ODE with initial value

$$x(t_0) = x_0 \in \mathbb{R}^n,$$

so that the first order partial derivatives of $f(t, x)$ are continuous on $[t_0, t_f] \times \mathbb{R}^n$. Given a mesh $t_0 < t_1 < \cdots < t_{l-1} < t_l = t_f$, ODE (1) can be approximated by a set of LDEs obtained as a result of a linear approximation of $f(t, x(t))$ at each subinterval ([9, 11]),

$$\dot{y}(t) = f_i + J_i(y(t) - y_i) + g_i(t - t_i), t \in [t_i, t_{i+1}], \tag{2}$$
$$y(t_i) = y_i,\ i = 0, 1, \ldots, l-1,$$

The solution of (2) is

$$y(t) = y_i + E_{12}^{(i)}(t - t_i) f_i + E_{13}^{(i)}(t - t_i) g_i, \tag{3}$$

where $E_{12}^{(i)}(t - t_i)$ and $E_{13}^{(i)}(t - t_i)$ are blocks $(1, 2)$ and $(1, 3)$ of $E = e^{C_i(t - t_i)}$, where

$$C_i = \begin{bmatrix} J_i & I_n & 0_n \\ 0_n & 0_n & I_n \\ 0_n & 0_n & 0_n \end{bmatrix}. \qquad \square$$

If t is replaced by t_{i+1} in (3), the approximate solution of ODE (1) at t_{i+1}, $i = 0, 1, \ldots, l-1$, is given by

$$y_{i+1} = y_i + E_{12}^{(i)}(\Delta t_i) f_i + E_{13}^{(i)}(\Delta t_i) g_i,\ \Delta t_i = t_{i+1} - t_i. \tag{4}$$

In this work, another approach based on the piecewise-linearized method is introduced as follows.

From [10, p. 716], $e^{C_i \Delta t_i}$ can be expressed as

$$\begin{bmatrix} e^{J_i \Delta t_i} & E_{12}^{(i)}(\Delta t_i) & E_{13}^{(i)}(\Delta t_i) \\ 0_n & I_n & I_n \Delta t_i \\ 0_n & 0_n & I_n \end{bmatrix},$$

whereas the approximate solution y_{i+1} given in (4) can be obtained adding to y_i the first n components of vector

$$e^{C_i \Delta t_i} v_i, \tag{5}$$

where

$$C_i = \begin{bmatrix} J_i & I_n & 0_n \\ 0_n & 0_n & I_n \\ 0_n & 0_n & 0_n \end{bmatrix}, v_i = \begin{bmatrix} 0_{n\times 1} \\ f_i \\ g_i \end{bmatrix}.$$

The matrix-vector product $e^{C_i \Delta t_i} v_i$ can be obtained by a Krylov subspace method [12, 13]. Given $A \in \mathbb{R}^{n\times n}$ and $v \in \mathbb{R}^n$, it is possible to compute an approximation to vector $e^A v$ by using the Arnoldi method. This approximation is given by

$$e^A v \cong v_{opt} = \beta V_p e^{H_p} e_1, \tag{6}$$

where $H_p = (h_{ij}) \in \mathbb{R}^{p\times p}$ is the Hessenberg matrix obtained from the Arnoldi method and $V_p = [v_1, v_2, \ldots, v_p] \in \mathbb{R}^{n\times p}$, with $\{v_i\}_{i=1,2,\ldots,p}$ an orthonormal basis of the Krylov subspace $K_p = \text{span}\{v, Av, \ldots, A^{p-1}v\}$, $\beta = \|v\|_2$ and $e_1 = [1, 0, \cdots, 0]^T$.

In order to reduce computational and storage costs when we want to compute vector y_{i+1}, it is necessary to modify the classical Arnoldi's algorithm without explicitly forming matrix $C_i \Delta t_i$. Algorithm 1 solves IVPs for non-autonomous ODEs by the above piecewise-linearized method based on a Krylov subspace approach. This algorithm uses Algorithm 2 , which computes the approximate solution at t_{i+1} of IVP (1) for non-autonomous ODEs, obtained after the piecewise-linearized process, by a block-oriented implementation of the Krylov subspace approach. Its computational cost is $2n^2p + 6np(p+1) + 2(q + j_{H_p} + 1/3)p^3$ flops, where $j_{H_p} = \max(0, 1 + int(\log_2(||H_p||)))$. It is possible to reduce the computational and storage costs of Algorithm 1 when IVP (1) is autonomous.

3 Experimental results

The main objective of this section is to compare the MATLAB implementations of algorithm developed in Section 2 with the implementations developed by the authors of this paper in [10].

Algorithm 1 solves IVP (1) by a piecewise-linearized method based on a Krylov subspace approach

Function $y = \texttt{inolkr}(t, \texttt{data}, x_0, p, tol, q)$
Inputs: Time vector $t \in \mathbb{R}^{l+1}$; function `data` computes $f(\tau, y) \in \mathbb{R}^n$, $J(\tau, y) \in \mathbb{R}^{n\times n}$ and $g(\tau, y) \in \mathbb{R}^n$ ($\tau \in \mathbb{R}$, $y \in \mathbb{R}^n$); vector $x_0 \in \mathbb{R}^n$; dimension $p \in \mathbb{N}$ of the Krylov subspace; tolerance $tol \in \mathbb{R}^+$; order $q \in \mathbb{N}$ of the diagonal Padé approximation of the exponential function
Output: Matrix $Y = [y_1, \ldots, y_l] \in \mathbb{R}^{n\times l}$, $y_i \in \mathbb{R}^n$, $i = 1, 2, \ldots, l$
1: Compute the vectors c_1 and c_2 that contain the coefficients of terms of degree greater than 0 in the diagonal Padé approximation of the exponential function
2: $y_0 = x_0$
3: **for** $i = 0 : l - 1$ **do**
4: $\quad [J_i, f_i, g_i] = \texttt{data}(t_i, y_i)$
5: $\quad \Delta t_i = t_{i+1} - t_i$
6: $\quad y_{i+1} = \texttt{inlbkr}(J_i, f_i, g_i, y_i, \Delta t_i, p, tol, c_1, c_2)$ (Algorithm 2)
7: **end for**

As test battery four case studies of stiff ODEs were considered, which come from biology and chemical kinetics stiff problems. Numerous tests were made on they. For each case study and algorithm, the characteristic parameters were varied, although only the parameters which offered the same accuracy for the two implementations with the lower computational cost are presented.

What follows is a short description of the implemented algorithms and its characteristic parameters:

- `iaolwp` and `inolwp` solve IVPs for ODEs by a piecewise-linearized approach and a block-oriented version without scaling-squaring implementation of the diagonal Padé approximation method:
  - order $q = 2$ of the diagonal Padé approximation of the exponential function.
- `iaolkr` and `inolkr` solve IVPs for ODEs by a piecewise-linearized method based on Krylov subspaces:
  - dimension $p = 4$ of the Krylov subspace. In reference [12] there is an exhaustive study of the computation of the product between the exponential of a matrix and a vector by using Krylov subspaces. We have experimentally probed that by considering low or medium dimension matrices, it is only needed to consider a very reduce subspace dimension. In this work $p = 4$.
  - tolerance $tol = 10^{-6} \in \mathbb{R}^+$.
  - order $q = 2$ of the diagonal Padé approximation of the exponential function.

For each test, the following results are shown:

Algorithm 2 computes the approximate solution at t_{i+1} of IVP (1) for non-autonomous ODEs, obtained after the piecewise-linearized process, by a block-oriented implementation of the Krylov subspace approach

Function $y_{i+1} = \texttt{inlbkr}(J_i, f_i, g_i, y_i, \Delta t_i, p, tol, c_1, c_2)$

Inputs: Matrix $J_i \in \mathbb{R}^{n\times n}$; vector $f_i \in \mathbb{R}^n$; vector $g_i \in \mathbb{R}^n$; vector $y_i \in \mathbb{R}^n$; step size $\Delta t_i \in \mathbb{R}$; dimension $p \in \mathbb{N}$ of the Krylov subspace; tolerance $tol \in \mathbb{R}^+$; vectors $c_1, c_2 \in \mathbb{R}^q$ with the coefficients of terms of degree greater than 0 in the diagonal Padé approximation of the exponential function

Output: Vector $y_{i+1} \in \mathbb{R}^n$ given by expression (5)

$V(1:n,1) = 0_n$
$V(n+1:2n,1) = f_i$
$V(2n+1:3n,1) = g_i$
$\beta = ||V(n+1:3n,1)||_2$
if $\beta == 0$ **then**
    $y_{i+1} = y_i$
    Return
end if
$V(n+1:3n,1) = V(n+1:3n,1)/\beta$
for $j = 1:p$ **do**
    $w(1:n) = J_i V(1:n,j) + V(n+1:2n,j)$
    $w(n+1:2n) = V(2n+1:3n,j)$
    $w(1:2n) = \Delta t_i w(1:2n)$
    $w(2n+1:3n) = 0_n$
    for $i = 1:j$ **do**
        $H(i,j) = w^T V(1:3n,i)$
        $w = w - H(i,j)V(1:3n,i)$
    end for
    $s = \|w\|_2$
    if $s < tol$ **then**
        $p = j$
        Leave for loop
    end if
    $H(j+1,j) = s$
    $V(1:3n,j+1) = w/s$
end for
computes $E = e^{H_p}$
$y_{i+1} = y_i + \beta V(1:n,1:p)E(1:p,1)$

E_r	Δt=0.1	Δt=0.05	Δt=0.01	Δt=0.005	Δt=0.001
`iaolwp`	2.809e-04	7.523e-05	2.390e-06	5.840e-07	2.366e-08
`iaolkr`	2.348e-04	6.928e-05	2.759e-06	6.423e-07	2.399e-08

Table 1: Relative error (E_r) with $t = 10$ and Δt variable (case study 1).

T_e	Δt=0.1	Δt=0.05	Δt=0.01	Δt=0.005	Δt=0.001
`iaolwp`	0.014	0.021	0.114	0.257	6.231
`iaolkr`	0.025	0.048	0.201	0.428	6.802

Table 2: Execution time (T_e) in seconds with $t = 10$ and Δt variable (case study 1).

- Tables which contain the relative error

$$E_r = \frac{\|x - x^*\|_\infty}{\|x\|_\infty},$$

  where x^* is the computed solution and x is the analytic solution (case study 2) or the solution computed by the MATLAB function ode15s with a vector of relative error tolerances $rtol = 10^{-13}$ and a vector of absolute error tolerances $atol = 10^{-13}$ [14].

- Tables/Figures with the execution time.

The algorithms were implemented in Matlab 7.9 and tested on an Intel Core 2 Duo processor at 2.66 GHz with 2 GB main memory. Several tests have been developed in order to determine the accuracy and efficiency of the algorithms. The implemented algorithms are available online at http://www.grycap.upv.es/odelin.

3.1 Case of study 1 (Pollution problem [15])

This case study corresponds to a stiff IVP of dimension twenty. The problem describes a chemical process consisting of 25 reactions and 20 species. The following tests were done:

- First test (Tables 1 and 2): t=10 and Δt variable.

- Second test (Table 3 and Figure 1): Δt=0.01 and t variable.

E_r	t=20	t=30	t=40	t=50	t=60
`iaolwp`	2.015e-06	1.744e-06	1.537e-06	1.374e-06	1.240e-06
`iaolkr`	2.327e-06	2.013e-06	1.775e-06	1.585e-06	1.431e-06

Table 3: Relative error (E_r) $\Delta t = 0.01$ and t variable (case study 1).

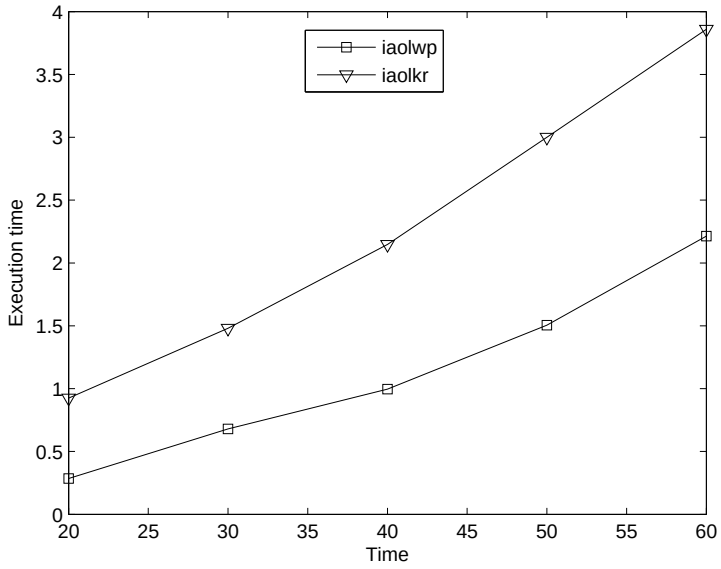

Figure 1: Execution time in seconds of the MATLAB implementations considering $\Delta t = 0.01$ and varying t (case study 1).

E_r	$t = 15400$	$t = 16400$	$t = 17400$	$t = 18400$	$t = 19400$
`inolwp`	4.410e-14	8.833e-14	1.431e-13	1.980e-13	2.528e-13
`inolkr`	4.410e-14	8.833e-14	1.431e-13	1.980e-13	2.528e-13

Table 4: Relative error (E_r) with $\Delta t = 0.1$ and t variable (case study 2).

3.2 Case of study 2 (Emep problem [15])

In this case study a stiff IVP for ODEs of dimension sixty-six is solved. The problem describes a problem which consists of 66 chemical species and about 140 reactions. The following tests were done:

- In the first test t=14450 was considered. With $\Delta t = 0.1$ the relative errors of the three implementation were equal to $2.219 \cdot 10^{-15}$, with executions times equal to 1.290 (`inolwp`) and 0.266 (`inolkr`) seconds.

- Second test (Tables 4 and 5, and Figure 2): Δt=0.1 and t variable.

T_e	$t = 15400$	$t = 16400$	$t = 17400$	$t = 18400$	$t = 19400$
`inolwp`	52.830	157.465	316.257	529.469	790.217
`inolkr`	26.547	102.401	227.630	400.672	623.248

Table 5: Execution time (T_e) in seconds with $\Delta t = 0.1$ and t variable (case study 2).

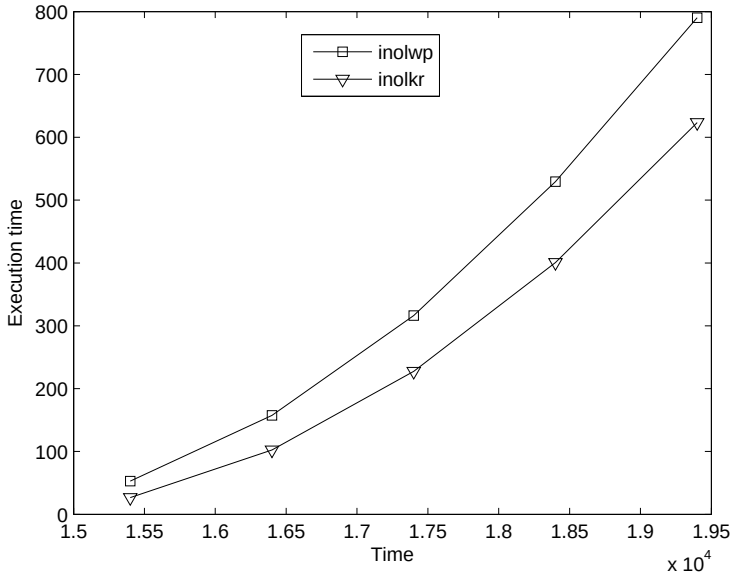

Figure 2: Execution time in seconds of the MATLAB implementations considering $\Delta t = 0.1$ and varying t between 15400 and 19400 (case study 2).

E_r	Δt=0.01	Δt=0.001	Δt=0.0001	Δt=0.00001
`inolwp`	1.572e-02	1.726e-03	1.741e-04	1.742e-05
`inolkr`	1.663e-02	1.728e-03	1.741e-04	1.742e-05

Table 6: Relative error (E_r) considering $n = 100$, $t = 1$ and Δt variable (case study 3).

3.3 Case study 3 (Medical Akzo Nobel problem [15])

This case study corresponds to a stiff non-autonomous ODE [15] of variable dimension $2N$. This problem studies the penetration of radio-labeled antibodies into a tissue infected by a tumor.

The following tests were made:

- First test (Tables 6 and 7): $n = 100$ ($N = 50$), t=1 and Δt variable.
- Second test (Tables 8 and 9): Δt=0.001, $t = 1$ and varying n from 50 to 250 ($N = 25$ to 125).

3.4 Case of study 4 (Brusselator problem) [1, pp. 6]

This case study corresponds to a stiff non-autonomous ODE of variable dimension N. This problem comes from chemical kinetics where the model of Lefever and Nicolis [16] is used and the method of lines is applied on a grid of N points:

- First test (Tables 10 and 11): $n = 100$ ($N = 50$), t=1 and Δt variable.

T_e	Δt=0.01	Δt=0.001	Δt=0.0001	Δt=0.00001
`inolwp`	0.301	4.926	144.484	6362.490
`inolkr`	0.036	0.538	50.044	5263.304

Table 7: Execution time (T_e) in seconds considering $n = 100$, $t = 1$ and Δt variable (case study 3).

E_r	n=50	n=100	n=150	n=200	n=250
`inolwp`	1.636e-03	1.726e-03	1.746e-03	1.743e-03	1.736e-03
`inolkr`	1.637e-03	1.728e-03	1.752e-03	1.763e-03	1.781e-03

Table 8: Relative error (E_r) considering $\Delta t = 0.001$, $t = 1$ and n variable (case study 3).

- Second test (Tables 12 and 13): $t = 1$, Δt=0.001 and n variable.

4 Conclusions and future work

In this work a new piecewise-linearized approach for solving ODEs based on Krylov subspaces has been presented. Two algorithms based on this approach (`inolkr` and `iaolbk`) have been also proposed and compared to the piecewise-linearized algorithms `iaolwp` and `inolwp` based on Padé approximants developed by the authors of this paper in[10].

Numerous test have been made on four case studies that come from biology and chemical kinetics stiff problems. Experimental results show the advantages of the proposed algorithms, especially when they are integrating stiff problems. According to the experimental results, the new algorithms offer in general similar precision and smaller computational cost when the problem size is increased. For example, Algorithm 1 (`inolkr`) was up to 111 times faster than `inolwp` for $n = 250$ and $t = 1$ in case study 3. This is because in the new approach the vector $e^A v$, $A \in \mathbb{R}^{n\times n}, v \in \mathbb{R}^n$, is approximated by the expression $\beta V_p e^{H_p} e_1$, where $p << n$. Nevertheless, when the problems are of small dimension, computational costs of piecewise-linearized algorithms based on diagonal Padé approximants are smaller than the computational costs of piecewise-linearized algorithms based on Padé approximants. In general, all algorithms offer accuracy and good behaviour with stiff problems.

As future work new improvements will be developed such as:

1. To implement algorithms based on the piecewise-linearized approach with error

T_e	n=50	n=100	n=150	n=200	n=250
`inolwp`	0.720	3.531	20.863	63.944	143.920
`inolkr`	0.288	0.482	0.740	1.159	1.367

Table 9: Execution time (T_e) in seconds considering $\Delta t = 0.001$, $t = 1$ and n variable (case study 3).

E_r	Δt=0.01	Δt=0.001	Δt=0.0001	Δt=0.00001
`inolwp`	2.162e-02	3.673e-04	3.715e-05	3.719e-06
`inolkr`	2.263e-02	3.672e-04	3.715e-05	3.719e-06

Table 10: Relative error (E_r) considering $n = 100$, $t = 1$ and Δt variable (case study 4).

T_e	Δt=0.01	Δt=0.001	Δt=0.0001	Δt=0.00001
`inolwp`	0.140	1.688	55.297	4106.738
`inolkr`	0.031	0.465	38.122	3710.951

Table 11: Execution time (T_e) in seconds considering $n = 100$, $t = 1$ and Δt variable (case study 4).

E_r	n=50	n=100	n=150	n=200	n=250
`inolwp`	5.033e-04	3.673e-04	3.308e-04	3.170e-04	3.108e-04
`inolkr`	5.033e-04	3.672e-04	3.307e-04	3.169e-04	3.107e-04

Table 12: Relative error (E_r) considering $\Delta t = 0.001$, $t = 1$ and n variable (case study 4).

T_e	n=50	n=100	n=150	n=200	n=250
`inolwp`	0.521	1.894	2.988	7.616	16.391
`inolkr`	0.279	0.506	0.701	0.963	1.256

Table 13: Execution time (T_e) in seconds considering $\Delta t = 0.001$, $t = 1$ and n variable (case study 4).

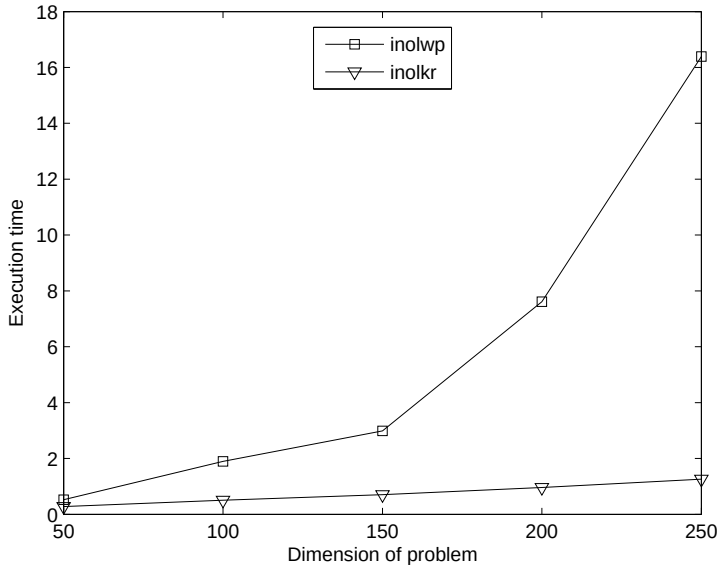

Figure 3: Execution time in seconds of the MATLAB implementations considering $\Delta t = 0.001$ and varying t between 50 and 250 (case study 4).

control in order to vary step size dynamically. The tests reported here considered constant step size. It is possible to improve the developed algorithms, using a variable step size which can be used to estimate the error committed in each iteration [9].

2. Carrying out parallel implementation of the algorithms presented in this work in a distributed memory platform, using the message passing paradigm MPI [17] and BLACS [18] for communications, and PBLAS [19] and ScaLAPACK [20] for computations.

References

[1] E. Hairer, G. Wanner, Solving ordinary differential equations II. Stiff and differential-algebraic problems., in: Springer Series in Computational Mathematics, Vol. 14, Springer-Verlag, 1996.

[2] J. C. Butcher, Numerical Methods for Ordinary Differential, Second Edition, 2008.

[3] J. Vigo-Aguiar, H. Ramos, A new eight-order a-stable method for solving differential systems arising in chemical reactions, Journal of Mathematical Chemistry 40 (2006) 71–83.

[4] C. F. Curtiss, J. O. Hirschfelder, Integration of stiff equations, in: Proc. Nat. Acad. Sci., Vol. 38, 1952, pp. 235–243.

[5] A. C. Hindmarsh, Lsode and lsodi, two new initial value ordinary differential equation solvers, ACM-Signum Newslett 15 (1980) 10–11.

[6] C. W. Gear, Algorithm 407-difsub for solution of ordinary differential equations, Comm. ACM 14 (1971) 185–190.

[7] P. N. Brown, G. D. Byrne, A. C. Hindmarsh, Vode: A variable-coefficient ode solver, SIAM J. Sci. Statist. Comput. 10 (1989) 1038–1051.

[8] J. I. Ramos, C. M. García, Piecewise-linearized methods for initial-value problems, Applied Mathematics and Computation 82 (1997) 273–302.

[9] C. M. García, Métodos de linealización para la resolución numérica de ecuaciones diferenciales, Ph.D. thesis, Departamento de Lenguajes y Ciencias de la Computación, Universidad de Málaga (1998).

[10] J. Ibáñez, V. Hernández, E. Arias, P. Ruiz, Solving initial value problems for ordinary differential equations by two approaches: BDF and piecewise-linearized methods, Computer Physics Communications 180 (5) (2009) 712–723.

[11] C. M. García, Piecewise-linearized and linearized θ-methods for ordinary and partial differential equation problems, Computer & Mathematics with Applications 45 (2003) 351–381.

[12] Y. Saad, Analysis of some Krylov subspace approximations to the matrix exponential operator, SIAM Journal on Numerical Analysis 29 (92) 209–228.

[13] R. B. Sidje, Expokit: A software package for computing matrix exponentials, ACM Trans. Math. Softw. 24 (1998) 130–156.

[14] L. S. Shampine, I. Gladwell, S. Thomson, Solving ODEs with MATLAB, Cambridge University Press, 2003.

[15] W. M. Lioen, J. J. B. de Swart, Test set for initial value problems solvers, release 2.0 (December 1998).

[16] R. Lefever, G. Nicolis, Chemical instabilities and sustained oscillations, J. Theor. Biol 30 (1971) 267–284.

[17] W. Gropp, E. Lusk, A. Skjellum, Using MPI: Portable Parallel Programming with the Message-Passing Interface, MIT Press, 1994.

[18] J. J. Dongarra, R. A. V. D. Geijn, Two dimensional basic linear algebra communications subprograms, Tech. rep., Department of Computer Science, University of Tennessee (1991).

[19] J. Choi, J. Dongarra, S. Ostrouchov, A. Petitet, D. Walker, A proposal for a set of parallel basic linear algebra subprogram, Tech. Rep. UT-CS-95-292, Department of Computer Science, University of Tennessee (1995).

[20] L. S. Blackford, J. Choi, A. Cleary, E. D'Azevedo, J. Demmel, I. Dhillon, ScaLAPACK Users' Guide, SIAM, 1997.