Taylor Series Based Integration in Electric Circuits Simulations

Vaclav SATEK [1,2], *Petr VEIGEND* [1], *Gabriela NECASOVA* [1]

[1]Department of Intelligent Systems, Faculty of Information Technology, Brno University of Technology, Bozetechova 2, 612 66 Brno, Czech Republic
[2]IT4Innovations National Supercomputing Center, VSB–Technical University of Ostrava, Studentska 6231/1B, 708 00 Ostrava, Czech Republic

satek@fit.vut.cz, satek@vsb.cz, iveigend@fit.vut.cz, inecasova@fit.vut.cz

DOI: 10.15598/aeee.v17i3.3369

Abstract. *This paper deals with the extremely precise, stable and fast solution of the ordinary differential equations. The solution of these is performed using a method based on the Taylor series - The Modern Taylor Series Method. The paper investigates two problems to demonstrate the positive properties of the method: linear problem - the behavior of signal transmission on the telegraph line and a non-linear problem - the Van der Pol oscillator. Both problems were analyzed and solved using newly implemented MATLAB Modern Taylor Series Method solvers. The results were then compared to the state-of-the-art MATLAB solvers.*

Keywords

Initial value problems, MATLAB, ordinary differential equations, Taylor series method, telegraph line, Van der Pol oscillator.

1. Introduction

The paper deals with the solution of technical Initial Value Problems (IVPs) representing the problems which arise from common technical practice (especially from electrical and mechanical engineering). Initial value problems are represented by the system of Ordinary Differential Equations (ODEs).

The best-known and the most accurate method of calculating a new value of the numerical solution of ODE [1]:

$$y' = f(t, y), \quad y(t_0) = y_0, \tag{1}$$

is to construct the Taylor series in the form:

$$y_{i+1} = y_i + h \cdot f(t_i, y_i) + \frac{h^2}{2!} \cdot f'(t_i, y_i) + \ldots + \frac{h^n}{n!} \cdot f^{[n-1]}(t_i, y_i), \tag{2}$$

where h is the size of integration step, $y_i = y(t_i)$ is the previous value and $y_i + 1 = y(t_i + h)$ is the next value of the function $y(t)$.

The Taylor series can be very effectively implemented as the variable-order, variable-step-size numerical method [2] - Modern Taylor Series Method (MTSM). The method is based on a recurrent calculation of the terms of the Taylor series for each integration step. Therefore, the complicated calculation of higher order derivatives does not need to be performed, the value of higher derivative is calculated numerically from the previous one [3]. Equation (2) can then be rewritten in the form:

$$y_{i+1} = DY_0 + DY_1 + DY_2 + \cdots + DY_n, \tag{3}$$

where DY_i denotes the terms of the Taylor series. Theoretically, it is possible to compute the solution of homogeneous linear differential equations with constant coefficients with arbitrary order and with arbitrary accuracy. Let us denote as ORD the function which changes during the computation and defines the number of terms of the Taylor series used in the current integration step ($ORD_{i+1} = n$).

The important property of the method is automatic order control, i.e. using as many terms of the Taylor series as the defined accuracy requires. Therefore it is common that the number of terms of the Taylor series varies for different constant integration step sizes.

The MTSM has been implemented in MATLAB [4], in C/C++ languages (FOS and TKSL/C software [2]).

Additionally, the method can be effectively implemented directly in hardware [5].

Several other implementations of the Taylor series method in a variable order and variable step context were presented by different authors. TIDES software [6] and TAYLOR [7], which includes detailed description of a variable step size version. Other implementations based on Taylor series include ATOMF [8], COSY INFINITY [9], and DAETS [10]. The variable stepsize variable-order scheme is also described in [11], [12] and [13], where simulations on a parallel computer are shown. The approach based on an approximate formulation of the Taylor methods can be found in [14].

This paper amends and extends the paper presented at the 2018 Modern Mathematical Methods in Engineering conference [15]. The solution of linear ODEs now uses fixed step and fixed order instead of variable step variable order approach shown in the paper. The MTSM algorithm for nonlinear quadratic ODEs is improved with recurrent calculation of terms of the Taylor series (higher derivatives are not needed for calculation). Due to these changes, the computation time of MTSM was improved in comparison to the paper [15] presented at the conference.

The paper is divided into several sections. In Sec. 2. the effective numerical solution of a system of linear ODEs using higher order MTSM is shown and the Telegraph equation is analyzed. The Sec. 3. presents the solution of quadratic nonlinear ODEs and the nonlinear Van der Pol oscillator is discussed. All algorithms of MTSM are efficiently implemented in MATLAB software [4] using vectorization. Finally, the MTSM algorithms are compared with MATLAB solvers [16].

2. Solution of Linear ODEs

Equation (2) for linear systems of ODEs in the form $\vec{y}' = \mathbf{A}\vec{y} + \vec{b}$ can be rewritten as:

$$\vec{y}_{i+1} = \vec{y}_i + h\left(\mathbf{A}\vec{y}_i + \vec{b}\right) + \frac{h^2}{2!}\mathbf{A}\left(\mathbf{A}\vec{y}_i + \vec{b}\right) + \ldots + \frac{h^n}{n!}\mathbf{A}^{(n-1)}\left(\mathbf{A}\vec{y}_i + \vec{b}\right), \tag{4}$$

where $\mathbf{A}$ is the constant Jacobian matrix and $\vec{b}$ is the constant right-hand side of the system.

Moreover, Eq. (4) can be rewritten in the form Eq. (3) where terms of the Taylor series can be computed recurrently:

$$D\vec{Y}_0 = \vec{y}_i, \quad D\vec{Y}_1 = h\left(\mathbf{A}\vec{y}_i + \vec{b}\right), \quad D\vec{Y}_l = \frac{h}{l}\mathbf{A}DY_{l-1}, \quad l = 2, \ldots, n. \tag{5}$$

The recurrent calculation of Taylor series terms is useful in error control. It is stopped when:

$$\sum_{j=n-stop}^{n} ||D\vec{Y}_j|| \leq eps, \tag{6}$$

where *eps* means error per step and *stop* denotes the number of successive terms of the Taylor series, which have met the stopping criterion Eq. (6). In this paper, the *stop* is set to 3.

For the solution of linear ODEs with constant integration step size h, the constant number of Taylor series terms ($ORD = n$) which satisfies Eq. (6) is obtained.

The Eq. (6) can be also rewritten in the form:

$$\vec{y}_{i+1} = \mathbf{A}_y\vec{y}_i + \mathbf{A}_b\vec{b}, \tag{7}$$

where the matrices $\mathbf{A}_y$ and $\mathbf{A}_b$ are in the form:

$$\mathbf{A}_y = \sum_{j=0}^{n}\frac{h^j}{j!}\mathbf{A}^j, \quad \mathbf{A}_b = \sum_{j=1}^{n}\frac{h^j}{j!}\mathbf{A}^{j-1}. \tag{8}$$

The fixed integration step size can be approximated using:

$$h < \sqrt[n]{\frac{eps \cdot n}{||\mathbf{A}^n||}}, \tag{9}$$

therefore, constant matrices $\mathbf{A}_y$ and $\mathbf{A}_b$ are precalculated only once at the beginning of the solution.

2.1. Telegraph Equation

The telegraph line is represented by the electric circuit depicted in Fig. 1 [17]. The behavior of the circuit is described by the system of ODEs:

$$u'_{C_j} = \frac{1}{C_j}(i_j - i_{j+1}), \quad i'_j = \frac{u_{L_j}}{L_j}, \quad j = 1, \ldots, S, \tag{10}$$

where S denotes the number of RLC segments of the telegraph line. The solution of the system of ODEs Eq. (10) leads to the linear IVP in the form:

$$\vec{y}' = \mathbf{A}\vec{y} + \vec{b}, \quad \vec{y}(0) = \vec{y}_0, \tag{11}$$

where $\mathbf{A}$ is a matrix of constants (R, L, and C parameters of the circuit), $\vec{y}$ is a vector of variables (voltages and currents), $\vec{b}$ is a vector of constants and $\vec{y}_0$ is a vector of initial conditions. The block structure of matrix $\mathbf{A}$ and vectors $\vec{y}$ and $\vec{b}$ can be found in [17].

For the simulation experiments in this paper, the capacitances and inductances are the same, $C_1 = C_2 = \cdots = C_S = 1$ pF and $L_1 = L_2 = \cdots = L_S = 10$ nH (homogeneous lossless telegraph line).

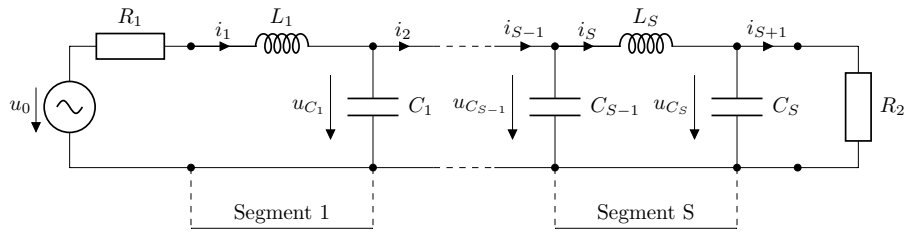


Fig. 1: Model of the telegraph line - series of S segments.

Moreover, the transmission line is adjusted if $R_1 = R_2 = \sqrt{L \cdot C^{-1}} = 100\ \Omega$. The angular velocity is set $\omega = 3 \cdot 10^9$ rad·s^{-1}. The input voltage u_0 should be generally constant (DC circuit) or harmonic (AC circuit) signal. In the case of DC circuit, the input voltage u_0 is hidden in constant right-hand side $\vec{b}$. In the case of AC circuit, the input voltage $u_0 = U_0 \sin(\omega t)$ can be computed using auxiliary system of coupled linear ODEs:

$$\begin{aligned} u_0' &= \omega x, \qquad u_0(0) = 0, \\ x' &= -\omega u_0, \quad x(0) = U_0. \end{aligned} \tag{12}$$

The AC circuit with input voltage $u_0 = \sin(\omega t)$ is used. The propagation constant per unit length of one segment for simple model of transmission line in Fig. 1 can be calculated as $t_{LC} = \sqrt{LC}$. The total delay of the input signal can be computed as $t_{delay} = St_{LC}$. The delay of the output voltage $u_{C_{100}}$ for 100 segments is shown in Fig. 2. The time of simulation was set at $t_{\max} = 2t_{delay}$ for all experiments.

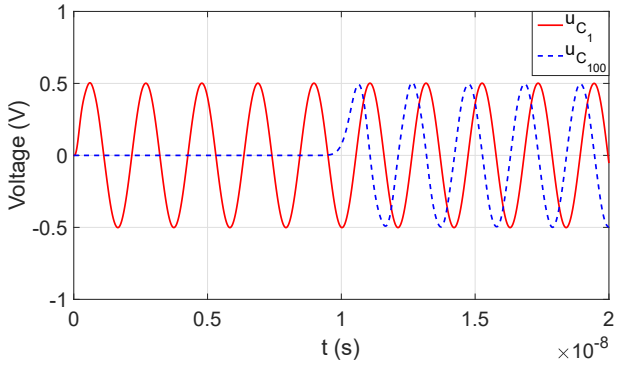


Fig. 2: Delay of the signal on the transmission line with $S = 100$ segments.

The MATLAB solver, using the explicit MTSM with a constant order and constant step size scheme Eq. (7) for linear systems of ODEs Eq. (11) has been implemented. This algorithm was tested on a set of examples of a telegraph line with different number of segments S. The maximum simulation time $t_{\max}$ is dependent on the number of segments. The number of integration steps rises with the maximum simulation time, see Tab. 3. Moreover, it is essential to verify the stability of methods.

The MTSM was compared with vectorized MATLAB explicit **ode** solvers. Both relative and absolute tolerances for all solvers were set to $eps = 10^{10}$. All codes are implemented in MATLAB 2015a and computations were partially performed on SALOMON supercomputer at IT4Innovations National Supercomputing Center, VSB–Technical University of Ostrava [18].

Comparisons of MTSM and MATLAB **ode** solvers [16] are in Tab. 1 and Tab. 2 for MTSM with order 30 and 60, respectively. Each run time is taken as a median value of 100 computations. Ratios of computation times $ratio = \mathbf{ode/expTay} \gg 1$ indicate significantly faster computation using MTSM in all cases.

Tab. 1: Time of solutions: explicit Taylor **expTay** ($ORD = 30$) and MATLAB explicit **ode** solvers comparison.

S	**ode23** *ratio*	**ode45** *ratio*	**ode113** *ratio*	**expTay** (s)
200	2919.2	439.3	148.1	0.0054
600	1101.4	215.7	61.3	0.052
1000	934.7	163.7	47.9	0.14
1400	1015.9	180.5	46.2	0.26
1800	989.9	161.8	41.7	0.42

Tab. 2: Time of solutions: explicit Taylor **expTay** ($ORD = 60$) and MATLAB explicit **ode** solvers comparison.

S	**ode23** *ratio*	**ode45** *ratio*	**ode113** *ratio*	**expTay** (s)
200	4573.5	693.3	234.9	0.0035
600	1639.3	311.8	91.1	0.035
1000	1267.2	221.2	65.9	0.1
1400	1505.6	273.5	68.3	0.17
1800	1536.4	248.3	64.4	0.28

The number of integration steps is depicted in Tab. 3, where abbreviations $\mathbf{Tay}_{30}$ and $\mathbf{Tay}_{60}$ mean **expTay** of $ORD = 30$ and $ORD = 60$, respectively.

Tab. 3: Numerical solution: number of integration steps.

S	**ode23**	**ode45**	**ode113**	$\mathbf{Tay}_{30}$	$\mathbf{Tay}_{60}$
200	95627	26548	4297	147	55
600	267984	71908	12866	440	165
1000	439362	114316	21436	733	275
1400	610765	155172	30005	1026	385
1800	782287	194984	38574	1319	495

More comparisons of MTSM numerical solutions of linear systems of ODEs can be found in [19] and [20].

3. Solution of Nonlinear (Quadratic) ODEs

In this section, the effective solution of a special case of nonlinear quadratic systems of ODEs is described. The nonlinear quadratic system of ODEs is any firstorder ODE that is quadratic in the unknown function. For such system, Taylor series based numerical method can be implemented in a very effective way.

Equation (1) for nonlinear-quadratic systems of ODEs can be rewritten as:

$$\vec{y}' = \mathbf{A}\vec{y} + \mathbf{B}\vec{y}_{jk} + \mathbf{C}\vec{y}^2 + \vec{b}, \quad \vec{y}(0) = \vec{y}_0, \tag{13}$$

where $\mathbf{A} \in \Re^{ne\times ne}$ is the matrix for linear part of the system, $\mathbf{B} \in \Re^{ne\times ne(ne-1)/2}$ is the matrix for mixed quadratic term, $\mathbf{C} \in \Re^{ne\times ne}$ is the matrix for pure quadratic term, $\vec{b} \in \Re^{ne}$ is the right-hand side for the forces incoming to the system, $\vec{y}_0$ is a vector of initial conditions, and the symbol ne stands for the number of equations of the system of ODEs. The unknown function $\vec{y}^2$ represents the vector of multiplications $(y_1y_1,\ y_2y_2,\ldots,\ y_{ne}y_{ne})^T$ and similarly $\vec{y}_{jk}$ represents the vector of mixed terms multiplications $(y_{j_1}y_{k_1},\ y_{j_2}y_{k_2},\ldots,\ y_{j_{ne(ne-1)/2}}y_{k_{ne(ne-1)/2}})T$. The indexes j and k come from combinatorics $C(ne, 2)$. For simplification, the matrices $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and the vector $\vec{b}$ are constant.

Higher derivatives of Eq. (3) can be effectively computed in MATLAB software [4] using matrix-vector multiplication, e.g. higher derivative $\vec{y}^{[p]}$ for pure quadratic term (with matrix $\mathbf{C}$) can be expressed as:

$$\vec{y}^{[p]} = \mathbf{C}\left(\sum_{i=0}^{p-2} \vec{y}^{[p-1-i]}.*\vec{y}^{[i]}\binom{p-1}{i} + \vec{y}.*\vec{y}^{[p-1]}\right), \tag{14}$$

where the operation '.*' stands for the *elementbyelement* multiplication, i.e. $\vec{y}^{[p_1]}.*\vec{y}^{[p_2]}$ is a vector $(y_1^{[p_1]}y_1^{[p_2]},\ldots,y_{ne}^{[p_1]}y_{ne}^{[p_2]})^T$. The binomial coefficients $\binom{p-1}{i}$ can be effectively precalculated using Pascal triangle, for more information, *see pascal* function in MATLAB software.

Moreover, the higher derivatives of the terms $\mathbf{B}\vec{y}_{jk}$, $\mathbf{C}\vec{y}^2$, in Eq. (13) can be included in a recurrent calculation of Taylor series terms $D\vec{Y}_B$ and $D\vec{Y}_C$:

$$\begin{aligned}
&D\vec{Y}_{B_0} = D\vec{Y}_{C_0} = \vec{0},\\
&D\vec{Y}_{B_1} = h(\mathbf{B}\vec{y}_{jk}), \quad D\vec{Y}_{C_1} = h(\mathbf{C}\vec{y}_i^2),\\
&D\vec{Y}_{B_l} = \frac{h}{l}\left(\mathbf{B}\sum_{m=1}^{l} D\vec{Y}_{j,l-m}.*D\vec{Y}_{k,m-1}\right),\\
&D\vec{Y}_{C_l} = \frac{h}{l}\left(\mathbf{C}\sum_{m=1}^{l} D\vec{Y}_{l-m}.*D\vec{Y}_{m-1}\right),\\
&D\vec{Y}_l = D\vec{Y}_{A_{l-1}} + D\vec{Y}_{B_{l-1}} + D\vec{Y}_{C_{l-1}},\\
&l = 2,\ldots,n,
\end{aligned} \tag{15}$$

where the linear term $DY_{A_{l-1}}$ is computed using Eq. (5).

3.1. Van der Pol Oscillator

The Van der Pol oscillator is a nonconservative oscillator with nonlinear damping. Energy is dissipated at high amplitudes and generated at low amplitudes. As a result, there are oscillations around a state at which energy generation and dissipation balance.

Due to the unique nature of the Van der Pol oscillator, it has become the cornerstone for studying systems with limit cycle oscillations. In fact, the Van der Pol equation has become a staple model for oscillatory processes not only in physics, but also in biology, sociology, and even economics.

From mathematical point of view, the Van der Pol equation is an ordinary differential equation:

$$y'' + \mu(y^2 - 1)y' + y = b, \tag{16}$$

where the parameter μ represents the nonlinearity and the strength of damping [21]. Right-hand side function b represents the forcing function. When $\mu = 0$, the equation becomes $y'' + y = 0$, which is a simple harmonic oscillator. For $\mu > 0$ the system enters a limit cycle: near the origin $(y = y' = 0)$, the system is unstable; far from the origin, the system is damped, see Fig. 3.

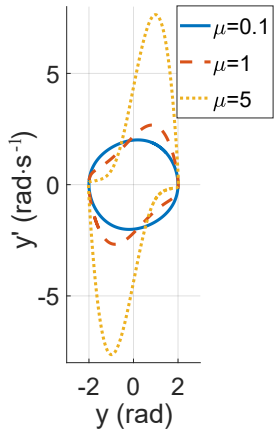


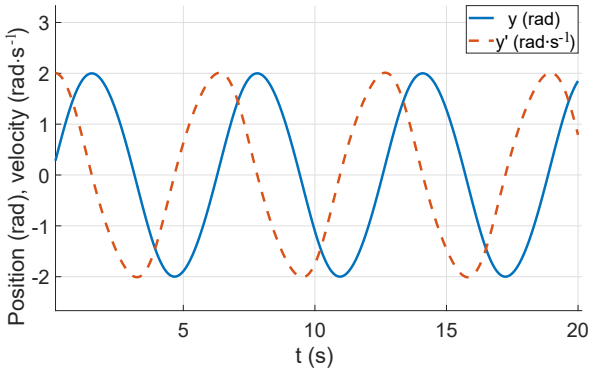
Fig. 3: Limit cycles of unforced Van der Pol oscillator.

The behavior of the system for different parameters μ in the time domain is shown in Fig. 4.

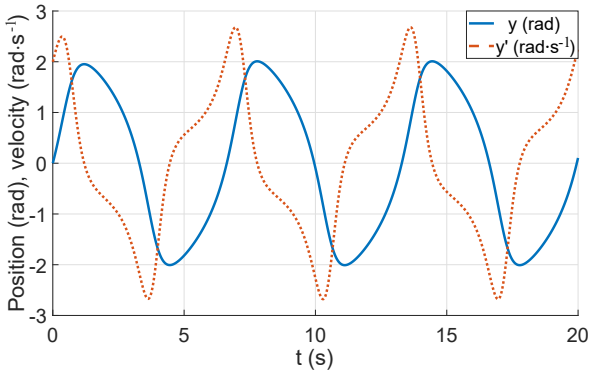
The second order ODE Eq. (16) can be transformed (using substitutions $y_1 = y'$ and $y_2 = y^2$) into the auxiliary system of ODEs:

$$\begin{aligned}
y' &= y_1,\\
y_1' &= \mu(1 - y_2)y_1 - y + b,\\
y_2' &= 2yy_1,
\end{aligned} \tag{17}$$

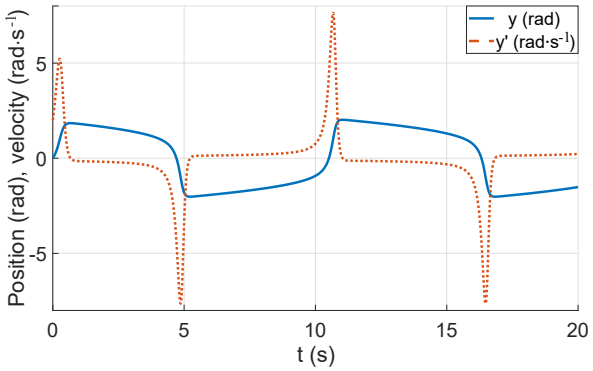
with initial conditions $y(0)$, $y_1(0) = y'(0)$, $y_(0) = y^2(0)$ and zero right-hand side $b = 0$.



(a) $\mu = 0.1$.

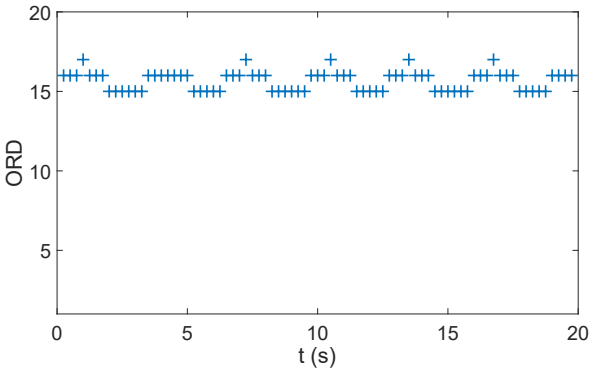


(b) $\mu = 1$.

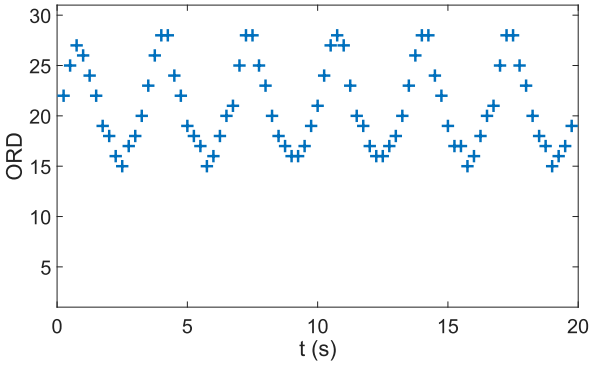


(c) $\mu = 5$.

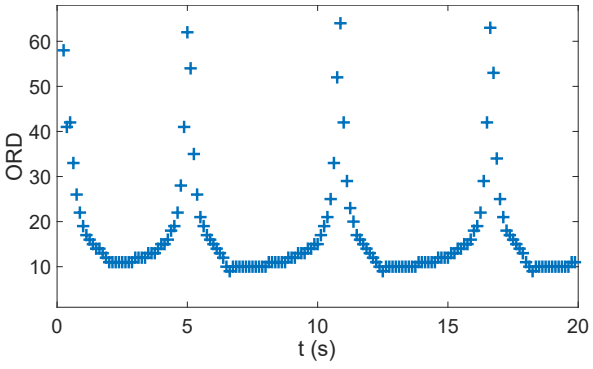
Fig. 4: Behavior of Van der Pol oscillator in the time domain.



(a) $\mu = 0.1$.



(b) $\mu = 1$.



(c) $\mu = 5$.

Fig. 5: The MTSM solution: *ORD* function.

The system Eq. (17) is equivalent with Eq. (13), where:

$$\vec{y} = \begin{pmatrix} y \\ y_1 \\ y_2 \end{pmatrix}, \quad \vec{b} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \quad \mathbf{A} = \mathbf{0},$$
$$\mathbf{B} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -\mu \\ 2 & 0 & 0 \end{pmatrix}, \tag{18}$$
$$\mathbf{C} = \begin{pmatrix} 0 & 1 & 0 \\ -1 & \mu & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

The MATLAB solver of explicit variable step size and variable order MTSM for nonlinar quadratic systems of ODEs Eq. (11) has been implemented. This algorithm was tested on a set of examples of Van der Pol systems Eq. (17) with different values of the parameter μ. The MTSM was again compared with vectorized MATLAB explicit **ode** solvers. Both relative and absolute tolerances for all solvers were set to $eps = 10^{-10}$. Results of the comparisons of the MTSM with MATLAB ode solvers are shown in Tab. 4. Ratios of computation times $ratio$ = **ode** = **expTay** > 1 indicate faster computation of the MTSM in all cases. The number of integration steps is shown in Tab. 5. The MTSM Order (ORD) is shown in Fig. 5.

As can be seen, the *ORD* is changing rapidly especially for problems involving higher non-linearities (larger parameter μ). Behavior of the function *ORD* demonstrates that the method changes used integration order dynamically during calculation with respect to the value of parameter *eps*.

Tab. 4: Time of solutions: explicit Taylor **expTay** and MATLAB explicit **ode** solvers comparison.

μ	**ode23** *ratio*	**ode45** *ratio*	**ode113** *ratio*	**expTay** (s)
0.1	156.4	8.43	1.9	0.063
1	193.9	8.9	4.12	0.088
5	108.3	8.2	3.1	0.15
10	58.1	5.9	2.1	0.2

Tab. 5: Numerical solution: number of integration steps.

μ	**ode23**	**ode45**	**ode113**	**expTay**
0.1	283858	22280	1969	100
1	354201	36660	3986	250
5	337565	57344	5002	1000
10	241783	54700	4581	2000

3.2. Forced Van der Pol Oscillator

The forced Van der Pol oscillator equation adds to the Eq. (16) nonzero right-hand side (e.g. some harmonic signal $b(t) = F\sin(\omega t)$):

$$y'' - \mu(1 - y^2)y' + y = F\sin(\omega t), \qquad (19)$$

where F is the amplitude and ω is angular velocity of the wave signal (rad·s^{-1}). The limit cycle for $\mu = 5$, $\omega = 1$ and $F = 2$ is depicted in Fig. 7.

The behavior of such system in the time domain is shown in Fig. 8. In electric circuits, the wave function can be represented as some AC power supply $b(t) = u_0 = U_0\sin(\omega t)$, see Fig. 6.

Equation (19) is equivalent to the system of five ODEs Eq. (17) with generating system for right-hand side sine function Eq. (12). This autonomous system of ODEs can be again represented in matrix-vector form Eq. (13).

Results of numerical solutions of forced Van der Pol Eq. (19) with different parameter μ and set parameters $\omega = 1$, $F = 2$ are in Tab. 6. Ratio of computation times *ratio* = **ode** = **expTay** > 1 indicates again faster computation of the MTSM in all cases. The number of integration steps for all methods can be found in Tab. 7.

Take a note, that the parameter ω in Eq. (19) has also significant impact on numerical computations, see Tab. 8 and Tab. 9. The last line of Tab. 9 (for $\omega > 10^4$) indicates that the computation time of ode23 solver is more than 50 hours.

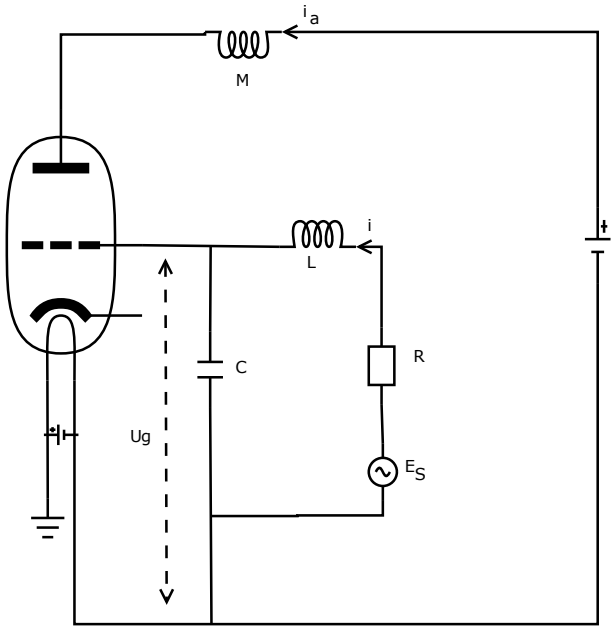


Fig. 6: Forced Van der Pol Oscillator.

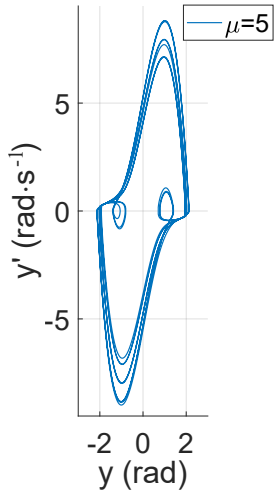


Fig. 7: Limit cycle of forced Van der Pol oscillator.

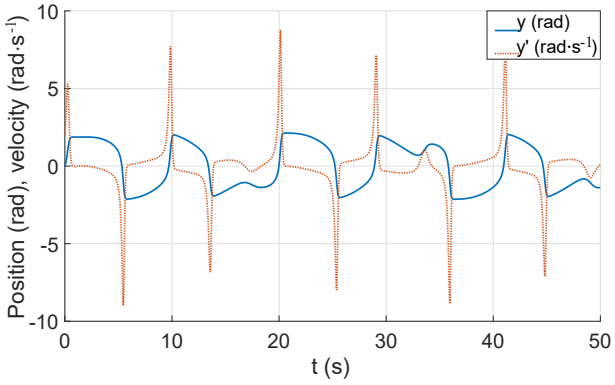


Fig. 8: Behavior of the forced Van der Pol oscillator in the time domain.

More comparisons of MTSM numerical solutions of nonlinear systems of ODEs can be found in [22].

Tab. 6: Forced Van der Pol oscillator: time of numerical solution.

μ	ode23 *ratio*	ode45 *ratio*	ode113 *ratio*	expTay (s)
0.1	250.6	9.4	3.9	0.076
1	190	10	3.8	0.12
5	130.4	8.5	3.1	0.17
10	18.7	6.7	2.3	0.22

Tab. 7: Forced Van der Pol oscillator: number of integration steps.

μ	ode23	ode45	ode113	expTay
0.1	382029	32184	3347	100
1	449441	52732	4779	400
5	464230	67424	5727	1000
10	385322	68740	5665	2000

Tab. 8: Forced Van der Pol oscillator ($\mu = 5$, $F = 2$): time of numerical solution.

ω (rad·s^{-1})	ode23 *ratio*	ode45 *ratio*	ode113 *ratio*	expTay (s)
$\mathbf{10^1}$	434.9	21.8	4.2	0.21
$\mathbf{10^2}$	1025.8	51.3	8.5	0.9
$\mathbf{10^3}$	1526.3	77.9	10.7	10.5
$\mathbf{10^4}$	> 1500	88	12.7	121.4

Tab. 9: Forced Van der Pol oscillator ($\mu = 5$, $F = 2$): number of integration steps.

ω (rad·s^{-1})	ode23	ode45	ode113	expTay
$\mathbf{10^1}$	$19 \cdot 10^5$	$21 \cdot 10^4$	$11 \cdot 10^4$	$10 \cdot 10^2$
$\mathbf{10^2}$	$19 \cdot 10^6$	$21 \cdot 10^5$	$94 \cdot 10^3$	$20 \cdot 10^2$
$\mathbf{10^3}$	$19 \cdot 10^7$	$21 \cdot 10^6$	$94 \cdot 10^4$	$20 \cdot 10^3$
$\mathbf{10^4}$	-	$21 \cdot 10^7$	$94 \cdot 10^5$	$20 \cdot 10^4$

4. Conclusion

This article dealt with the numerical solution of linear and nonlinear systems of ODEs coming from electrical circuits simulations. The model of the telegraph line was chosen as the example of the linear problem, the Van der Pol oscillator as the example of nonlinear one. All calculations were performed using MATLAB software. The MTSM solver for nonlinear quadratic systems of ODEs was successfully implemented. The MTSM is able to solve these problems faster and more efficiently than the state-of-the-art **ode** solvers in MATLAB.

Future work will be focused on the solution of general nonlinear problems, parallelization of the MTSM algorithm and its hardware representation.

Acknowledgment

This research was financially supported by the Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPU II) project "IT4Innovations excellence in science - LQ1602". The paper also includes the results of the internal BUT FIT project FIT-S-17-4014.

References

[1] HAIRER, E., S. P. NORSETT and G. WANNER. *Solving Ordinary Differential Equations I: Non-stiff Problems*. 1st. ed. Berlin: Springer-Verlag, 1987. ISBN 978-3-662-12607-3.

[2] KUNOVSKY, J. High Performance Computing. In: *Faculty of Information Technology* [online]. 2019. Available at: `https://www.fit.vutbr.cz/~satek/MTSM/index.html`.

[3] KUNOVSKY, J. Modern Taylor series method. In: *2015 IEEE 13th International Scientific Conference on Informatics*. Poprad: IEEE, 2015, pp. 1–8. ISBN 978-1-4673-9868-8. DOI: 10.1109/Informatics.2015.7377798.

[4] MATLAB and Simulink software. In: *MathWorks* [online]. 2017. Available at: `http://www.mathworks.com`.

[5] KOCINA, F., G. NECASOVA, P. VEIGEND, V. SATEK and J. KUNOVSKY. Parallel solution of higher order differential equations. In: *2016 International Conference on High Performance Computing & Simulation (HPCS)*. Innsbruck: IEEE, 2016, pp. 302–309. ISBN 978-1-5090-2088-1. DOI: 10.1109/HPCSim.2016.7568350.

[6] RODRIGUEZ, M., A. ABAD, R. BARRIO and F. BLESA. TIDES: A free software based on the Taylor series method. *Monografias de la Real Academia de Ciencias de Zaragoza*. 2011, vol. 35, iss. 1, pp. 83–95. ISSN 1132-6360.

[7] JORBA, A. and M. ZOU. A software package for the numerical integration of ODEs by means of high-order Taylor methods. *Experimental Mathematics*. 2005, vol. 14, iss. 1, pp. 99–117. ISSN 1058-6458. DOI: 10.1080/10586458.2005.10128904.

[8] CHANG, Y. F. and G. CORLISS. ATOMFT: solving ODEs and DAEs using Taylor series. *Computers & Mathematics with Applications*. 1994, vol. 28, iss. 10, pp. 209–233. ISSN 0898-1221. DOI: 10.1016/0898-1221(94)00193-6.

[9] BENZ, M. and K. MAKINO. *COSY INFINITY version 8 reference manual, NSCL Technical Report MSUCL-1088*, 1st ed. East Lansing: National Superconducting Cyclotron Laboratory, Michigan State University, 1997.

[10] NEDIALKOV, N. S. and J. D. PRYCE. Solving differential-algebraic equations by Taylor series (III): The DAETS code. *Journal of Numerical Analysis, Industrial and Applied Mathematics*. 2008, vol. 3, iss. 1, pp. 61–80. ISSN 1790–8140.

[11] BARRIO, R., F. BLESA and M. LARA. VSVO formulation of the Taylor method for the numerical solution of ODEs. *Computers & mathematics with Applications*. 2005, vol. 50, iss. 1, pp. 93–111. ISSN 0898-1221. DOI: 10.1016/j.camwa.2005.02.010.

[12] BARRIO, R. Performance of the Taylor series method for ODEs/DAEs. *Applied Mathematics and Computation*. 2005, vol. 163, iss. 2, pp. 525–545. ISSN 0096-3003. DOI: 10.1016/j.amc.2004.02.015.

[13] MOHAZZABI, P. and J. L. BECKER. Numerical Solution of Differential Equations by Direct Taylor Expansion. *Journal of Applied Mathematics and Physics*. 2017, vol. 5, no. 3, pp. 623–630. ISSN 2327-4379. DOI: 10.4236/jamp.2017.53053.

[14] BAEZA, A., S. BOSCARINO, P. MULET, G. RUSSO and D. ZORIO. Approximate Taylor methods for ODEs. *Computers & Fluids*. 2017, vol. 159, iss. 1, pp. 156–166. ISSN 0045-7930. DOI: 10.1016/j.compfluid.2017.10.001.

[15] NECASOVA, G., P. VEIGEND and V. SATEK. Modern Taylor series method in numerical integration: PART 2. In: *17th Czech-Polish Conference Modern Mathematical Methods in Engineering (3mi)*. Horni Lomna: VSB–TU Ostrava, 2018, pp. 211–220. ISBN 978-80-248-4135-9.

[16] MATLAB: Choose an ODE Solver. In: *MathWorks* [online]. 2017. Available at: `https://www.mathworks.com/help/matlab/math/choose-an-ode-solver.html`.

[17] VEIGEND, P., G. NECASOVA and V. SATEK. Model of the telegraph line and its numerical solution. *Open Computer Science*. 2018, vol. 8, iss. 1, pp. 10–17. ISSN 2299-1093. DOI: 10.1515/comp-2018-0002.

[18] IT4Innovations Documentation: Salomon Cluster Documentation - Matlab Overview. In: *IT4Innovations Documentation* [online]. 2015. Available at: `https://docs.it4i.cz/software/numerical-languages/matlab/`.

[19] NECASOVA, G., P. VEIGEND, V. SATEK and J. KUNOVSKY. Model of the telegraph line. In: *2017 IEEE 14th International Scientific Conference on Informatics*. Poprad: IEEE, 2017, pp. 271–275. ISBN 978-1-5386-0889-0. DOI: 10.1109/INFORMATICS.2017.8327259.

[20] SATEK, V., F. KOCINA, J. KUNOVSKY and A. SCHIRRER. Taylor series based solution of linear ode systems and matlab solvers comparison. *IFAC-PapersOnLine*. 2015, vol. 48, iss. 1, pp. 693–694. ISSN 2405-8963. DOI: 10.1016/j.ifacol.2015.05.210.

[21] JORDAN, D. W., P. SMITH. *Nonlinear ordinary differential equations: an introduction for scientists and engineers*. 4th. ed. New York: Oxford University Press, 2007. ISBN: 978-01-992-0825-8.

[22] SATEK, V., P. VEIGEND and G. NECASOVA. Taylor series based solution of nonlinear-quadratic ODE systems. In: *MATHMOD 2018 Extended Abstract Volume, 9th Vienna Conference on Mathematical Modelling*. Vienna: 2017, pp. 99–100. ISBN 978-3-901608-91-9. DOI: 10.11128/arep.55.a55267.

About Authors

Vaclav SATEK was born in Ostrava, Czech Republic. He received his M.Sc. from Brno University of Technology, Faculty of Information Technology in 2006 and Ph.D. in 2012 at the same institution. His research interests include numerical solution of ordinary and partial differential equations - stiff systems, computational fluid dynamics, contact problems etc.

Petr VEIGEND was born in Brno, Czech Republic. He received his M.Sc. from Brno University of Technology, Faculty of Information Technology in 2014. His research interests include numerical calculations and system modeling and simulations.

Gabriela NECASOVA was born in Brno, Czech Republic. She received his M.Sc. from Brno University of Technology, Faculty of Information Technology in 2014. Her research interests include numerical solution of differential equations and parallel methods.