Int. J. Nonlinear Anal. Appl. 15 (2024) 4, 339–347
ISSN: 2008-6822 (electronic)
http://dx.doi.org/10.22075/ijnaa.2023.30796.4496

The numerical solution of the nonlinear system of stiff differential equations by the modified matrix-exponential method

Mohammad Fattahi*, Mashallah Matinfar

Department of Mathematics, Science of Mathematics Faculty, University of Mazandaran, Babolsar, Iran

(Communicated by Ehsan Kozegar)

Abstract

In this paper, the modified matrix exponential (MME) method under zero-order hold (ZOH) assumption, is applied to solve systems of stiff ordinary differential equations. Some examples are given to illustrate the accuracy and effectiveness of the method. We compare our results with results obtained by matrix exponential (ME) method and by the Matlab ode23 solver.

Keywords: Modified matrix exponential, Matrix exponential, Nonlinear differential equations, Jacobian matrix
2020 MSC: Primary 34A34; Secondary 65F60

1 Introduction

Stiff initial and boundary value problems for ordinary differential equations arise in fluid mechanics, elasticity, electrical networks, chemical reactions, and many other areas of physical importance. A stiff equation is a differential equation for which certain numerical methods for solving the equation are numerically unstable unless the step size is taken to be extremely small. Because of analytical methods which can solve the stiff system are restricted, the ability to solve these equations numerically is important. In the last several decades, some of the numerical methods for stiff differential equations have been studied, such as Haar Wavelets [7], Adomian decomposition [4], Runge-Kutta [3] and so on.

In this paper, we apply the modified matrix exponential method to solve the stiff equations. The paper is organized as follows: a brief description of the MME method is given in section 2. The accuracy of the method with several examples is demonstrated in section 4 and the conclusion is described in section 5.

2 A brief description of the MME

Our objective is to present a time discretization of non-linear systems using modified matrix exponential methods as follows [5, 9]:

$$\frac{dY(t)}{dt} = f(Y(t)) + g(Y(t)) \bigodot v(t), \tag{2.1}$$

*Corresponding author
Email addresses: mf.math99@gmail.com (Mohammad Fattahi), m.matinfar@umz.ac.ir (Mashallah Matinfar)

Received: February 2023 *Accepted:* June 2023

where the vector $Y(t) = [y_1(t), y_2(t), y_3(t), ..., y_n(t)]^t \in X \subset R^{n\times 1}$ represents a set of open and connected states, and $v(t) = [v_1(t), v_2(t), ..., v_n(t)] \in R^{n\times 1}$ are the input variables and $\bigodot$ is a scalar product. Suppose that $f(x)$ and $g(x)$ are real analytic vector fields on X. In general, the mesh $T = t_{k+1} - t_k > 0$ represents an equidistant grid of points on the time axis, $[t_k, t_{k+1}) = [kT, (k+1)T)$ and T are the sampling interval and the sampling period, respectively.

It is also assumed that (2.1) is driven by an input, $v_i(t)$, that is piecewise constant over the sampling interval, i.e. the ZOH is true.

For the ZOH assumption,

$$v_i(t) = v_i(t_k) = \text{constant}, \tag{2.2}$$

for $a \le t_k \le b$. For $i, j = 1, 2, 3, ..., n$, we consider a time interval $t \in [t_k, t_{k+1})$ with the ZOH assumption, we have

$$\zeta_j(t) = Y_j(t) - Y_j(t_k) \tag{2.3}$$

and the following second-order approximation can be obtained:

$$f_i(Y(t)) \approx f_i(Y(t_k)) + \frac{\partial f_i(Y(t_k))}{\partial y_j(t)}\zeta_j(t) + \frac{\partial^2 f_i(Y(t_k))}{\partial y_j(t)^2}\frac{\zeta_j^2(t)}{2}, \tag{2.4}$$

$$g_i(Y(t)) \approx g_i(Y(t_k)) + \frac{\partial g_i(Y(t_k))}{\partial y_j(t)}\zeta_j(t) + \frac{\partial^2 g_i(Y(t_k))}{\partial y_j(t)^2}\frac{\zeta_j^2(t)}{2}. \tag{2.5}$$

From (2.3), we have

$$\dot{\zeta}_j(t) = \dot{Y}_j(t). \tag{2.6}$$

Thus, (2.2) can be approximated as follows:

$$\begin{aligned}
\dot{\zeta}_j(t) &\approx f_i(Y(t_k)) + \frac{\partial f_i(Y(t_k))}{\partial y_j(t)}\zeta_j(t) + \frac{\partial^2 f_i(Y(t_k))}{\partial y_j(t)^2}\frac{\zeta_j^2(t)}{2} + \left(g_i(Y(t_k)) + \frac{\partial g_i(Y(t_k))}{\partial y_j(t)}\zeta_j(t) + \frac{\partial^2 g_i(Y(t_k))}{\partial y_j(t)^2}\frac{\zeta_j^2(t)}{2}\right)v_i \\
&= (f_i(Y(t_k)) + g_i(Y(t_k))v_i) + \left(\frac{\partial f_i(Y(t_k))}{\partial y_j(t)} + \frac{\partial g_i(Y(t_k))}{\partial y_j(t)}v_i\right)\zeta_j(t) + \left(\frac{\partial^2 f_i(Y(t_k))}{\partial y_j(t)^2} + \frac{\partial^2 g_i(Y(t_k))}{\partial y_j(t)^2}v_i\right)\frac{\zeta_j^2(t)}{2} \\
&= \tilde{f}_{ik} + J_{ik}\zeta_j(t) + J'_{ik}\frac{\zeta_j^2(t)}{2},
\end{aligned} \tag{2.7}$$

where

$$\tilde{f}_{ik} = \tilde{f}_i(Y(t_k), v_i) = f_i(Y(t_k)) + g_i(Y(t_k))v_i \tag{2.8}$$

$$J_{ik} = J_i(Y(t_k), v_i) = \frac{\partial f_i(Y(t_k))}{\partial y_j(t)} + \frac{\partial g_i(Y(t_k))}{\partial y_j(t)}v_i \tag{2.9}$$

$$J'_{ik} = J'_i(Y(t_k), v_i) = \frac{\partial^2 f_i(Y(t_k))}{\partial y_j(t)^2} + \frac{\partial^2 g_i(Y(t_k))}{\partial y_j(t)^2}v_i \tag{2.10}$$

Rewriting (2.7), we get:

$$\dot{\zeta}_j(t) = \tilde{f}_{ik} + J_{ik}\zeta_j(t) + J'_{ik}\frac{\zeta_j^2(t)}{2}, \qquad \zeta_j(t_k) = 0. \tag{2.11}$$

Let $N > 0$ be an integer number, the step length is as follows:

$$h_k = \frac{t_{k+1} - t_k}{N} \tag{2.12}$$

An expand vector is considered:

$$\eta_j(t) = \begin{pmatrix} \zeta_j(t) \\ \frac{\zeta_j^2(t)}{2} \\ 1 \end{pmatrix} \tag{2.13}$$

(2.11) can be written as follows:

$$\begin{pmatrix} \dot{\zeta}_j(t) \\ \dot{\zeta}_j(t)\zeta_j(t) \\ 0 \end{pmatrix}_{(i+1)\times 1} = \begin{pmatrix} J_{ik} & J'_{ik} & \tilde{f}_{ik} \\ \dot{\zeta}_j(t) & 0 & 0 \\ \overline{0}^T & 0 & 0 \end{pmatrix}_{(i+1)\times(i+1)} \begin{pmatrix} \zeta_j(t) \\ \frac{\zeta_j^2(t)}{2} \\ 1 \end{pmatrix}_{(i+1)\times(1)} \tag{2.14}$$

Rewriting (2.14), we get:

$$\dot{\eta}_j(t) = C_{ik}\eta_j(t) \tag{2.15}$$

where

$$\dot{\zeta}_j(t_k) = \tilde{f}_{ik}(t_k), \eta_j(t_k) = \begin{pmatrix} \overline{0} \\ \overline{0} \\ 1 \end{pmatrix} = \eta_{j0}, \tag{2.16}$$

$$C_{ik}(t_k) = \begin{pmatrix} J_{ik}(t_k) & J'_{ik}(t_k) & \tilde{f}_{ik}(t_k) \\ \tilde{f}_{ik}(t_k) & 0 & 0 \\ \overline{0}^T & 0 & 0 \end{pmatrix} \in R^{(n+1)\times(n+1)} \tag{2.17}$$

and $\overline{0}$ is an n-dimensional zero column vector and J_{ik} is the first-order derivative and J'_{ik} is the second-order derivative of the Jacobian matrix and $\tilde{f}_{ik}$ is the values of equations in $y_i(t_k)$.

The solution of (2.15) within the time interval $[t_k, t_{k+1})$ is as follows:

$$\eta_j(t_{k+1}) = e^{C_{ik}(t_k)(t_{k+1}-t_k)}\eta_{j0}. \tag{2.18}$$

An exponential matrix is calculated by taking Z as the square matrix and I as the identity matrix. Its exact formula would be as follows:

$$e^Z = \lim_{N\to\infty}\left(I + \frac{Z}{N}\right)^N. \tag{2.19}$$

The following truncated approximation is applicable for an appropriate value of N:

$$e^Z \approx \left(I + \frac{Z}{N}\right)^N, \tag{2.20}$$

Using (2.18) and (2.20) we get:

$$e^{C_{ik}(t_{k+1}-t_k)} \approx (I_{(i+1)\times(i+1)} + C_{ik}(t_k)h_k)^N. \tag{2.21}$$

From (2.18) and (2.21), we can obtain:

$$\eta_j(t_{k+1}) = (I_{(2i+1)\times(2i+1)} + C_{ik}(t_k)h_k)^N\eta_{j0} \tag{2.22}$$

By multiplying the vector $\begin{pmatrix} I_{i\times i} & \overline{0} & \overline{0}\end{pmatrix}$ on the sides of (2.22)

$$\zeta_j(t_{k+1}) = \begin{pmatrix} I_{i\times i} & \overline{0} & \overline{0}\end{pmatrix}(I_{(2i+1)\times(2i+1)} + C_{ik}(t_k)h_{ik})^N \begin{pmatrix} \overline{0} \\ \overline{0} \\ 1 \end{pmatrix}, \tag{2.23}$$

where $\begin{pmatrix} I & \overline{0} & \overline{0}\end{pmatrix} \in R^{(n)\times(n+1)}$. So, The final equation can be obtained as follows:

$$y(t_{k+1}) = y(t_k) + \begin{pmatrix} I_{i\times i} & \overline{0} & \overline{0}\end{pmatrix}(I_{(2i+1)\times(2i+1)} + C_{ik}(t_k)h_{ik})^N \begin{pmatrix} \overline{0} \\ \overline{0} \\ 1 \end{pmatrix}, \tag{2.24}$$

we used the extended vector to apply the modified matrix exponential method. The (2.24) can be written in extended form as follows:

$$\begin{pmatrix} y_1(t_{k+1}) \\ y_2(t_{k+1}) \\ \vdots \\ y_i(t_{k+1}) \end{pmatrix}_{i\times 1} = \begin{pmatrix} y_1(t_k) \\ y_2(t_k) \\ \vdots \\ y_i(t_k) \end{pmatrix}_{i\times 1} + H(t_k, y_i(t_k)), \tag{2.25}$$

where

$$H(t_k, y_i(t_k)) = \begin{pmatrix} I_{i\times i} & \overline{0} & \overline{0}\end{pmatrix}(I_{(2i+1)\times(2i+1)} + C_{ik}(t_k)h_{ik})^N \begin{pmatrix} \overline{0} \\ \overline{0} \\ 1 \end{pmatrix},$$

for $i = 1, 2, 3, \cdots, n$. If $J'_{ik}(t_k) = 0$, then MME and ME methods are equivalent to each other.

3 Finding the appropriate value of N in MME

A proper value for N is essential [2]. An improved form for (2.20) is

$$e^Z \cong \left(I + \frac{Z}{2^b}\right)^{2^b},$$

for an appropriate value of b. We can show that analytical relative matrix error E_t defined by

$$\left(1 + \frac{Z}{2^b}\right)^{2^b} \equiv e^Z(1 + E_t),$$

is given approximately by

$$E_t \approx -\frac{1}{2}.\frac{Z^2}{2^b}$$

and therefore, for any matrix form,

$$\|E_t\| \approx \frac{1}{2}.\frac{\|Z\|^2}{2^b} \leq \frac{1}{2}.\frac{\|Z\|^2}{2^b}$$

Estimating the value of b required to have $E_t < \epsilon$, as follows

$$b^* \equiv int_+ \left(\log_2 \left(\frac{\|Z\|^2}{2\epsilon}\right)\right),$$

where $int(x)$ is the lowest integer greater than or equal to x, $E_t < \epsilon$ is a preassigned tolerance (maximum tolerable value) for E_t and

$$Z = (t_{k+1} - t_k)C_{ik}.$$

For the sake of safety, it's recommended to choose

$$b = b^* + 3.$$

4 Numerical illustration and discussion

Example 4.1. Consider the nonlinear stiff ODE problem as follows [8]:

$$\begin{cases} x'(t) = -x_1^2(t)x_2(t) - x_2^3(t), \\ x'(t) = -x_1(t) - 10^3(2cos(t)x_2(t) - sin(2t)). \end{cases}$$

where $t \in [0, \frac{\pi}{2}]$. The initial values are $x_1(0) = 1$ and $x_2(0) = 0$ and the exact solutions are $x_1(t) = cos(t)$ and $x_2(t) = sin(t)$. Now we construct as follows:

$$J_{ik} = \begin{pmatrix} -2x_1(t_k)x_2(t_k) & -3x_2^2(t_k) \\ 1 & -10^3(2cos(t_k)) \end{pmatrix}_{2\times 2},$$

$$J'_{ik} = \begin{pmatrix} -2x_2(t_k)x_2(t_k) & -6x_2(t_k) \\ 0 & 0 \end{pmatrix}_{2\times 2},$$

$$\tilde{f}_{ik} = \begin{pmatrix} -x_1^2(t_k)x_2(t_k) - x_2^3(t_k) \\ -x_1(t_k) - 10^3(2cos(t_k)x_2(t_k) - sin(2t_k)) \end{pmatrix}_{2\times 1},$$

where $i = 1, 2$. The solutions of example 4.1 are plotted in figure 1 for iteration=2×10^5 and $N = 2^5$ and the error values of the methods which is compared to exact solution are presented in figure 2. The average of absolute error of example 4.1 listed in table 1.

Table 1: The average of absolute error of example 4.1.

methods	**iter**=2×10^4 $N = 2^5$	**iter**=2×10^4 $N = 2^{20}$	**iter**=2×10^4 $N = 2^5$	**iter**=2×10^4 $N = 2^5$
MME	$X_1 : 6.8318 \times 10^{-6}$	6.9150×10^{-6}	6.7304×10^{-7}	6.7205×10^{-8}
	$X_2 : 9.3791 \times 10^{-6}$	8.6239×10^{-6}	9.8446×10^{-7}	9.8911×10^{-8}
ME	$X_1 : 1.8980 \times 10^{-5}$	1.9633×10^{-5}	1.8710×10^{-6}	1.8683×10^{-7}
	$X_2 : 1.1864 \times 10^{-5}$	1.2187×10^{-5}	1.1579×10^{-6}	1.1551×10^{-7}
ode23	$X_1 : 1.9670 \times 10^{-4}$	1.9670×10^{-4}	2.0007×10^{-6}	2.0007×10^{-6}
	$X_2 : 4.4654 \times 10^{-4}$	4.4654×10^{-4}	4.4502×10^{-4}	4.4502×10^{-4}

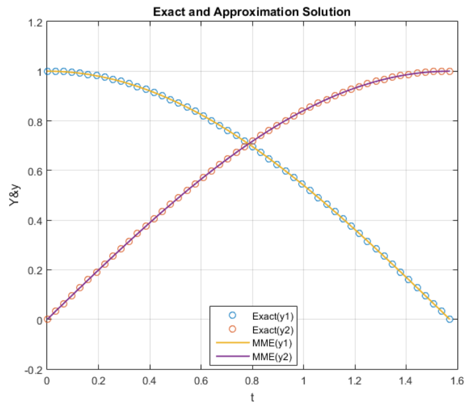

Figure 1: The exact and MME solution of example 4.1.

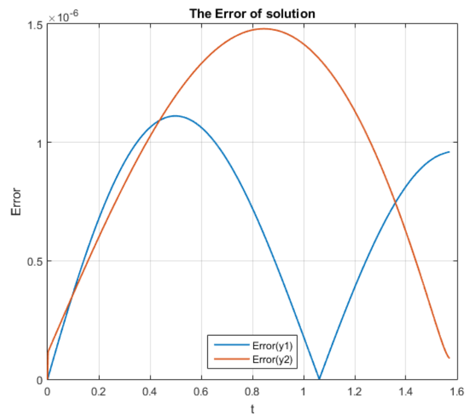

Figure 2: The errors of the MME method of example 4.1.

Example 4.2. Consider the stiff nonlinear initial value problem as follows [1]:

$$\begin{cases} x_1'(t) = -0.04x_1(t) + 10^4 x_2(t)x_3(t) - 0.96e^{-t}, \\ x_2'(t) = 0.04x_1(t) - 10^4 x_2(t)x_3(t) - 10^7 x_2^2(t) - 0.04e^{-t}, \\ x_3'(t) = 3 \times 10^7 x_2^2(t) + e^{-t}. \end{cases}$$

where $t \in [0, 15]$. The initial values are $x_1(0) = 1$, $x_2(0) = 0$ and $x_3(0) = 0$ and the exact solutions are $x_1(t) = e^{-t}$, $x_2(t) = 0$ and $x_3(t) = 1 - e^{-t}$.

Now we construct as follows:

$$J_{ik} = \begin{pmatrix} -0.04 & 10^4 x_3(t_k) & 10^4 x_2(t_k) \\ 0.04 & -10^4 x_3(t_k) - 2 \times 10^7 x_2(t_k) & -10^4 x_2(t_k) \\ 0 & 6 \times 10^7 x_2(t_k) & 0 \end{pmatrix}_{3\times 3}$$

$$J'_{ik} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & -2\times 10^7 & 0 \\ 0 & 6\times 10^7 & 0 \end{pmatrix}_{3\times 3},$$

$$\tilde{f}_{ik} = \begin{pmatrix} -0.04x_1(t_k) + 10^4 x_2(t_k)x_3(t_k) - 0.96e^{-t_k} \\ 0.04x_1(t_k) - 10^4 x_2(t_k)x_3(t_k) - 10^7 x_2^2(t_k) - 0.04e^{-t_k} \\ 3\times 10^3 x_2^2(t_k) + e^{-t_k} \end{pmatrix}_{3\times 1},$$

where $i = 1, 2, 3$. The values of error of example 4.2 rather than exact solution for $iteration = 6 \times 10^5$ and $N = 2^5$ are exhibited in table 2 and the solutions of example 4.2 are plotted in figure 3. The average of absolute error of example 4.2 listed in table 3 and the error values of the methods which is compared to exact solution are presented in figure 4.

Table 2: The values of error of example 4.2 rather than exact solution.

t	**x**	**MME**	**ME**	**Ode23**
0.8	X_1	3.18696×10^{-7}	3.03377×10^{-6}	3.03380×10^{-6}
	X_2	3.18720×10^{-7}	3.03380×10^{-6}	9.3791×10^{-5}
1.6	X_1	7.91787×10^{-7}	3.73712×10^{-6}	1.97248×10^{-5}
	X_2	7.91831×10^{-7}	3.73714×10^{-6}	5.69838×10^{-5}
2.4	X_1	1.15260×10^{-6}	3.75668×10^{-6}	3.01947×10^{-5}
	X_2	1.15265×10^{-6}	3.75670×10^{-6}	8.77373×10^{-5}
3.2	X_1	1.38133×10^{-6}	3.63226×10^{-6}	3.82825×10^{-5}
	X_2	1.38138×10^{-6}	3.63227×10^{-6}	1.18487×10^{-4}
4	X_1	1.51404×10^{-6}	3.51649×10^{-6}	4.99561×10^{-5}
	X_2	1.51409×10^{-6}	3.51651×10^{-6}	1.49231×10^{-4}

Table 3: The average of absolute error of example 4.2

methods	**iter=**6×10^4 $N = 2^5$	**iter=**10^5 $N = 2^5$	**iter=**6×10^5 $N = 2^5$	**iter=**6×10^5 $N = 2^{10}$
MME	$X_1 : 7.6581 \times 10^{-6}$	4.4304×10^{-6}	7.2261×10^{-7}	7.2268×10^{-7}
	$X_2 : 8.5399 \times 10^{-6}$	4.6280×10^{-6}	7.2352×10^{-7}	7.2370×10^{-7}
ME	$X_1 : 1.0223 \times 10^{-4}$	6.1335×10^{-5}	1.0222×10^{-5}	1.0222×10^{-5}
	$X_2 : 1.0223 \times 10^{-4}$	6.1335×10^{-5}	1.0223×10^{-5}	1.0223×10^{-5}
ode23	$X_1 : 9.4592 \times 10^{-5}$	9.4592×10^{-5}	failed	failed
	$X_2 : 2.8378 \times 10^{-4}$	2.8378×10^{-4}	failed	failed

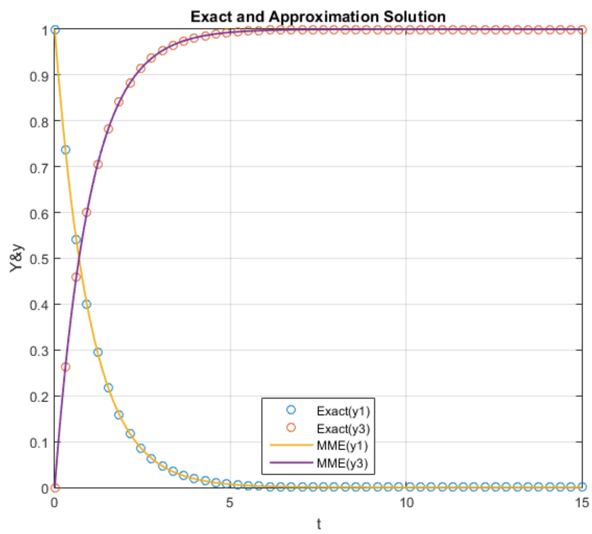

Figure 3: The exact and MME solution of example 4.2.

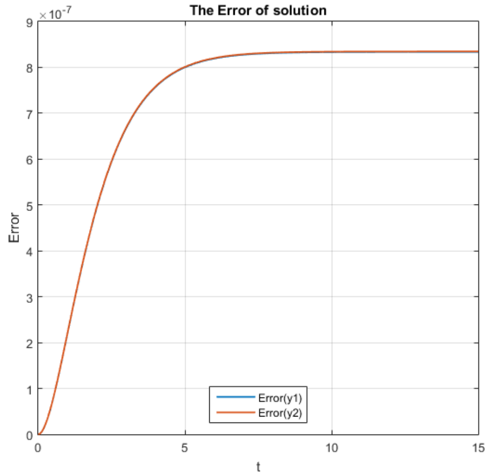

Figure 4: The errors of MME method of example 4.2.

Example 4.3. Consider the very stiff nonlinear initial value problem as follows [6]:

$$\begin{cases} x_1'(t) = 2tx_2^{\frac{1}{5}}(t)x_4(t), \\ x_2'(t) = 10te^{5(x_3(t)-1)}x_4(t), \\ x_3'(t) = 2tx_4(t), \\ x_4'(t) = -2tLn(x_1(t)), \end{cases},$$

where $t \in [0, 2.5]$. The initial values are $x(0) = (1, 1, 1, 1)$ and the exact solutions are $x(t) = (e^{sin(t^2)}, e^{5sin(t^2)}, sin(t^2)+1, cos(t^2))$. Now we construct as follows:

$$J_{ik} = \begin{pmatrix} 0 & \frac{2}{5}tx_4(t_k)x_2(t_k)^{-\frac{4}{5}} & 0 & 2tx_2(t_k)^{\frac{1}{5}} \\ 0 & 0 & 50tx_4(t_k)e^{5(x_3(t_k)-1)} & 10te^{5(x_3(t_k)-1)} \\ 0 & 0 & 0 & 2t \\ -\frac{2t}{x_1(t_k)} & 0 & 0 & 0 \end{pmatrix}_{4\times 4},$$

$$J'_{ik} = \begin{pmatrix} 0 & -\frac{8}{25}tx_4(t_k)x_2(t_k)^{-\frac{9}{5}} & 0 & 0 \\ 0 & 0 & 250tx_4(t_k)e^{5(x_3(t_k)-1)} & 0 \\ 0 & 0 & 0 & 0 \\ \frac{2t}{x_1^2(t_k)} & 0 & 0 & 0 \end{pmatrix}_{4\times 4},$$

$$\tilde{f}_{ik} = \begin{pmatrix} 2t_kx_2^{\frac{1}{5}}(t_k)x_4(t_k) \\ 10t_ke^{5(x_3(t_k)-1)}x_4(t_k) \\ 2t_kx_4(t_k) \\ -2t_kLn(x_1(t_k)) \end{pmatrix}_{4\times 1},$$

where $i = 1, 2, 3, 4$. The solutions of example 4.3 are plotted in figure 5 for $iteration = 7 \times 10^5$ and $N = 2^{20}$ and the error values of the methods which is compared to exact solution are presented in figure 6. The average of absolute error of example 4.3 listed in table 4.

Table 4: The average of absolute error of example 4.3.

methods	**iter**=6.5×10^4 $N = 2^2$	**iter**=6.5×10^4 $N = 2^{20}$	**iter**=7×10^5 $N = 2^{20}$
MME	$X_1 : 6.5617 \times 10^{-3}$	9.7823×10^{-6}	9.4725×10^{-7}
	$X_2 : 9.8648 \times 10^{-3}$	5.6039×10^{-4}	5.2057×10^{-5}
	$X_3 : 8.7241 \times 10^{-3}$	9.2018×10^{-6}	9.2412×10^{-7}
	$X_4 : 8.8071 \times 10^{-3}$	9.4651×10^{-6}	9.5627×10^{-7}

ME	$X_1 : 6.6945 \times 10^{-3}$	1.5367×10^{-5}	1.4271×10^{-6}
	$X_2 : 1.0348 \times 10^{-2}$	1.1610×10^{-3}	1.0778×10^{-4}
	$X_3 : 8.8022 \times 10^{-3}$	1.029×10^{-5}	9.6492×10^{-7}
	$X_4 : 8.9189 \times 10^{-3}$	9.5505×10^{-6}	9.2195×10^{-7}
ode23	$X_1 : 1.6524 \times 10^{-2}$	1.6524×10^{-2}	1.8311×10^{-2}
	$X_2 : 5.6368 \times 10^{-2}$	5.6368×10^{-2}	6.5253×10^{-2}
	$X_3 : 2.2230 \times 10^{-2}$	2.2230×10^{-2}	2.4547×10^{-2}
	$X_4 : 1.9485 \times 10^{-2}$	1.9485×10^{-2}	2.1365×10^{-2}

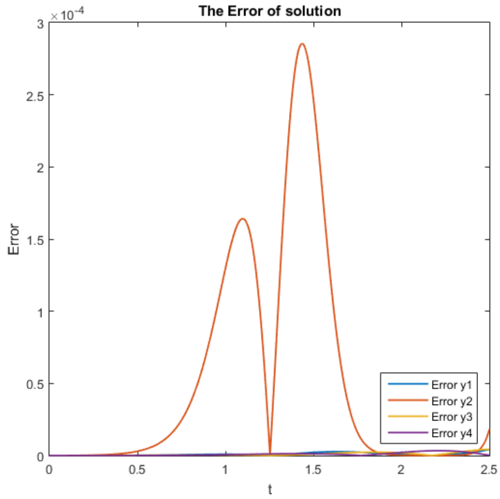

Figure 5: The exact and MME solution of example 4.3.

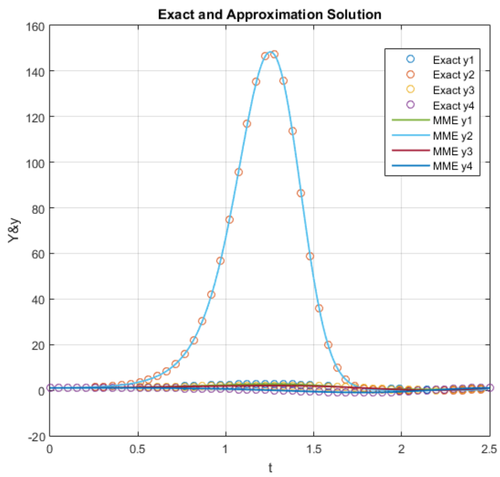

Figure 6: The errors of MME of example 4.3.

5 Conclusion

In this paper, we studied the modified matrix exponential method for solving the stiff differential equations. In order to apply the modified matrix exponential method, we used the extended vector as (2.25). The results were compared with the matrix exponential method and Matlab ode23 solver. The various examples and their errors display that MME method is a very good method for solving the type of stiff differential equation. It is important to appropriately set the value of N in this method. By increasing the iteration of the method or the value of N, we can get a better approximation with fewer errors. Because the MME method uses a small step size of h_k, it is appropriate for solving very stiff systems of ODEs and it is largely independent of ill-conditioning.

References

[1] A. Abdi, S.A. Hosseini and H. Podhaisky, *Adaptive linear barycentric rational finite differences method for stiff ODEs*, J. Comput. Appl. Math. **357** (2019), 204–214.

[2] F. Allffi-Pentini, V.D. Fonzo and V. Paris, *A novel algorithm for the numerical integration of systems of ordinary differential equations arising in chemical problem*, J. Math. Chem. **33** (2003), 1–15.

[3] M. Calvo, S. Gonzalez-Pinto and J.I. Montijano, *Runge–Kutta methods for the numerical solution of stiff semi-linear system*, BIT Numer. Math. **40** (2000), 611–639.

[4] M.S.H. Chowdhury, I. Hashim and Md. Alal Hosen, *Solving linear and non-linear stiff system of ordinary differential equations by multistage Adomian decomposition method*, Proc. The Third Intl. Conf. Adv. Appl. Sci. Environ. Technol. **125** (2015), 79–82.

[5] M. Fattahi and M. Matinfar, *The numerical solution of the second kind of Abel equations by the modified matrix-exponential method*, Int. J. Nonlinear Anal. Appl. In Press, (2023), 1–7.

[6] G.Y. Kulikov, R. Weiner and G.M. Amiraliyev, *Doubly quasi-consistent fixed-stepsize numerical integration of stiff ordinary differential equations with implicit two-step peer methods*, J. Comput. Appl. Math. **340** (2018), 256–275.

[7] U. Lepik, *Haar wavelet method for solving stiff differential equations*, Math. Model. Anal. **14** (2009), 79–82.

[8] S. Li, *Canonical Euler splitting method for nonlinear composite stiff evolution equations*, Appl. Math. Comput. **289** (2016), 220–236.

[9] Y. Park and K.T. Chong, *The numerical solution of the point kinetics equation using the matrix exponential method*, Ann. Nuclear Energy **55** (2013), 42–48.