ISSN: 1017-060X (Print)

ISSN: 1735-8515 (Online)

Bulletin of the Iranian Mathematical Society

Vol. 42 (2016), No. 2, pp. 297–314

Title:

A new approach for solving the first-order linear matrix differential equations

Author(s):

A. Golbabai, S. P. A. Beik and D. k. Salkuyeh

Published by Iranian Mathematical Society
http://bims.ims.ir

Bull. Iranian Math. Soc.
Vol. 42 (2016), No. 2, pp. 297–314
Online ISSN: 1735-8515

A NEW APPROACH FOR SOLVING THE FIRST-ORDER LINEAR MATRIX DIFFERENTIAL EQUATIONS

A. GOLBABAI*, S. P. A. BEIK AND D. K. SALKUYEH

(Communicated by Nezam Mahdavi-Amiri)

Abstract. The main contribution of the current paper is to propose a new effective numerical method for solving the first-order linear matrix differential equations. Properties of the Legendre basis operational matrix of integration together with a collocation method are applied to reduce the problem to a coupled linear matrix equations. Afterwards, an iterative algorithm is examined for solving the obtained coupled linear matrix equations. Numerical experiments are presented to demonstrate the applicably and efficiency of our method.

Keywords: Linear matrix differential equation, Legendre polynomials, Coupled linear matrix equations, Iterative algorithm.
MSC(2010): Primary: 65M06; Secondary: 15A24, 65M70.

1. Introduction

Matrix differential equations occur frequently as a model in chemistry, physics and engineering, see [1–3,5,12,19–21,24]. In the present work, we focus on the following first-order matrix differential equation

$$
\begin{cases} Y'(x) = A(x)Y(x) + B(x), \qquad a \le x \le b, \\ Y(a) = Y_a, \end{cases} \tag{1.1}
$$

where $Y \in \mathbb{R}^{p\times q}$ is an unknown matrix, the matrices $Y_a \in \mathbb{R}^{p\times q}$, $A(x) : [a, b] \to \mathbb{R}^{p\times p}$, and $B(x) : [a, b] \to \mathbb{R}^{p\times q}$ are given. Consider the first-order matrix differential equation (1.1) such that $A(x), B(x) \in C^s([a, b])$, $s \ge 1$, which guarantees the existence of a unique and continuously differentiable solution $Y(x)$ of (1.1); for more details see [13].

The applications of the first-order matrix differential equation (1.1) in the real world are admirably discussed in [8–10]. In [10], the matrix cubic splines have been utilized to compute an approximate solution of (1.1).

Article electronically published on April 30, 2016.
Received: 4 August 2013, Accepted: 31 December 2014.
*Corresponding author.

Assume that $\triangle_{[a,b]}$ denotes an arbitrary partition of $[a,b]$ defined by

$$\triangle_{[a,b]}= \{a = x_0 < x_1 < \cdots < x_n = b\}, \quad x_k = a + kh, \quad k = 0,1,2,\dots,n,$$

where n is a positive integer and $h = (b-a)/n$ is the step size. Recently, Defez et al. [8] have proposed a numerical method by means of higher-order matrix splines for solving (1.1). To do so, at each subinterval $[a+kh, a+(k+1)h]$, the exact solution $Y(x)$ is approximated by a matrix spline $S(x)$ of order $m \in \mathbb{N}$ with $1 \leq m \leq s$, where s stands for the order of differentiability. In [8], it is pointed out that the approximate solution of (1.1) can be calculated by using splines of order m in the interval $[a,b]$ with a local (global) error of the order $\mathcal{O}(h^m)$ $(\mathcal{O}(h^{m-1}))$ under some certain conditions.

In this paper, we present a novel framework to obtain the numerical solution of (1.1). In fact by using orthogonal polynomials [7] and collocation points, a new computational approach is given for solving (1.1). To this end, we first expand each entry of the approximate solution $Y'(x)$ with unknown coefficients in terms of Legendre polynomials up to degree m. Then, with the aid of a collocation method, we obtain a coupled linear matrix equations where its solution gives the unknown coefficients. In [23] the authors have developed the well-known CGNE iterative algorithm to solve general coupled matrix equations. We use this algorithm to solve the the obtained matrix equations.

The rest of this paper is organized as follows. In Section 2, we first review some definitions and concepts which are used throughout this paper. Then, we recall the basic properties of the Legendre functions which are required for our subsequent developments. In Section 3, we first discuss how the Legendre basis can be applied to reduce the problem (1.1) to the solution of a coupled matrix equations. Then, an iterative algorithm for solving the obtained coupled matrix equations is exploited. Section 4 is devoted to report some numerical experiments which demonstrate the accuracy of the proposed numerical scheme for solving (1.1) in compare with those given in [8]. Finally, the paper is ended with a brief conclusion in Section 5.

2. **Definitions and Properties**

In this section, we review some principles and results which are utilized for obtaining our main results. Throughout this paper, we use $\text{tr}(A)$ and A^T to represent the trace and the transpose of the matrix A, respectively. Furthermore, $\mathbb{R}^{m\times n}$ denotes the set of all $m \times n$ real matrices. The Kronecker product of the matrices $A = [a_{ij}]_{m\times s}$ and $B = [b_{ij}]_{n\times q}$ is defined as the $mn \times sq$ matrix $A \otimes B = [a_{ij}B]$. The "*vec*" operator transforms a matrix A of size $m \times s$ to a vector $a = \text{vec}(A)$ of size $ms \times 1$ by stacking the columns of A. For two matrices Y and Z in $\mathbb{R}^{m\times n}$, the inner product $\langle Y, Z\rangle_F$ is defined by $\langle Y, Z\rangle_F = \text{tr}(Y^T Z)$, where the associated matrix norm is the Frobenius norm denoted by $\|.\|_F$.

For two given functions $f, g \in L^2[a,b]$, the L^2 inner product on the interval $[a,b]$ is defined by

$$\langle f, g\rangle_2 = \int_a^b f(x)g(x)dx.$$

For a given function $f \in L^2[a,b]$, we call $\|f\|_2$ as the square norm of f and define as follows:

$$\|f\|_2 = \left(\int_a^b f(x)^2 dx\right)^{\frac{1}{2}}.$$

The following inner product and its induced norm will be utilized for establishing our theoretical results.

Definition 2.1. $\Phi = (\Phi_1, \Phi_2, \dots, \Phi_k)$ and $\widetilde{\Phi} = (\widetilde{\Phi}_1, \widetilde{\Phi}_2, \dots, \widetilde{\Phi}_k)$ be two matrix groups where $\Phi_i, \widetilde{\Phi}_i \in \mathbb{R}^{p\times q}$ for $i = 1, 2, \dots, k$. We define the inner product $\langle ., .\rangle$ as follows:

$$\langle \Phi, \widetilde{\Phi}\rangle := \sum_{j=1}^{k} \operatorname{tr}(\Phi_j^T \widetilde{\Phi}_j). \tag{2.1}$$

Remark 2.2. For any matrix group $\Phi = (\Phi_1, \Phi_2, \dots, \Phi_k)$, where $\Phi_i \in \mathbb{R}^{p\times q}$ for $i \in 1, \dots, k$, the norm of Φ is defined by $\|\Phi\|^2 := \sum\limits_{j=1}^{k} \operatorname{tr}(\Phi_j^T \Phi_j)$.

2.1. **Properties of the Legendre basis.** In this subsection, we recall some properties of the Legendre polynomials for our future use. The Legendre polynomials are defined on the interval [-1,1] and can be determined by the following recurrence formulae:

$$L_0(t) = 1, \qquad L_1(t) = t,$$

$$L_{m+1}(t) = \left(\frac{2m+1}{m+1}\right)tL_m(t) - \left(\frac{m}{m+1}\right)L_{m-1}(t), \qquad m = 1, 2, 3, \dots.$$

The Legendre polynomials are orthogonal polynomials with respect to L^2 inner product; for more details see [14, 18] and the references therein.

In order to use these polynomials on an arbitrary interval $[a,b]$, we define the so-called shifted Legendre polynomials by introducing the change of variable

$$t = \frac{2(x-a)-h}{h}, \qquad a \le x \le b,$$

where $h = b - a$. The shifted Legendre polynomials in x are then obtained by

$$p_0(x) = 1, \quad p_1(x) = \frac{2(x-a)-h}{h},$$

and for $m = 1, 2, 3, \dots$,

$$p_{m+1}(x) = \frac{2m+1}{h(m+1)}(2(x-a)-h)p_m(x) - \frac{m}{m+1}p_{m-1}(x).$$

Let us assume that $\phi_i \equiv \sqrt{\frac{2i+1}{h}}\, p_i$. Straightforward computations show that

$$\int_a^b \phi_i(x)\phi_j(x)dx = \begin{cases} 1, & \text{for} \quad i = j, \\ 0, & \text{for} \quad i \neq j. \end{cases}$$

Another way of presenting the shifted Legendre polynomials in $[a, b]$ in its analytical form is (see [18])

$$\phi_i(x) = \sqrt{\frac{2i+1}{h}} \sum_{k=0}^{i} \frac{(-1)^{i+k}(i+k)!(x-a)^k}{(i-k)!(k!)^2 h^k}. \tag{2.2}$$

2.2. **The best approximation of a function.** Let $H = L^2[a, b]$ and

$$Y = \operatorname{span}\{\phi_0, \phi_1, \ldots, \phi_m\},$$

where $m \in \mathrm{N} \cup \{0\}$ and ϕ_i's are the Legendre polynomials. Since $Y \subset H$ is a finite dimensional vector space, for every $f \in H$ there exists a unique $y_0 \in Y$ such that

$$\|f - y_0\|_2 \leq \|f - y\|_2 \qquad \forall y \in Y,$$

where $\|f\|_2 = \sqrt{\langle f, f\rangle}$. Here, the function y_0 is called the best approximation to f out of Y. Since $y_0 \in Y$,

$$f(x) \approx y_0(x) = \sum_{j=0}^{m} c_j \phi_j(x) = C^T \Psi(x),$$

where $C^T = (c_0, c_1, ..., c_m)$ such that c_j's are uniquely calculated by $c_j = \langle f, \phi_j\rangle$ and

$$\Psi^T(x) = (\phi_0(x), \phi_1(x), \ldots, \phi_m(x)). \tag{2.3}$$

For more details see [16].

Theorem 2.3. *([16]) Suppose that H is a Hilbert space and Y is a finite dimensional closed subspace of H and $\{y_1, y_2, \ldots, y_n\}$ is any basis for Y. Let x be an arbitrary element in H and y_0 be the unique best approximation to x from Y. Then*

$$\|x - y_0\|_2^2 = \frac{G(x, y_1, y_2, ..., y_n)}{G(y_1, y_2, ..., y_n)},$$

where

$$G(x, y_1, y_2, ..., y_n) = \begin{vmatrix} \langle x, x\rangle & \langle x, y_1\rangle & \ldots & \langle x, y_n\rangle \\ \langle y_1, x\rangle & \langle y_1, y_1\rangle & \ldots & \langle y_1, y_n\rangle \\ \vdots & \vdots & & \vdots \\ \langle y_n, x\rangle & \langle y_n, y_1\rangle & \ldots & \langle y_n, y_n\rangle \end{vmatrix}.$$

2.3. **The operational matrix of integration.** The integration of the vector $\Psi(t)$ in (2.3) can be approximated by

$$\int_a^t \Psi(s)ds \approx P\Psi(t),$$

where the $(m+1)\times(m+1)$ matrix P is called the operational matrix of integration. There are several ways to compute P. We will use the analytical form of Legendre basis to show how this matrix is constructed. By (2.2), we see that

$$\int_a^t \phi_i(s)ds = \sqrt{\frac{2i+1}{h}} \sum_{k=0}^{i} \frac{(-1)^{i+k}(i+k)!(t-a)^{k+1}}{(i-k)!(k!)^2(k+1)h^k}, \quad i=0,\dots,m.$$

Now we approximate $(t-a)^{k+1}$ in terms of Legendre basis up to degree m as

$$(t-a)^{k+1} \approx \sum_{j=0}^{m} \beta_j^k \phi_j(t),$$

in which

$$(2.4)\ \beta_j^k = \int_a^b (t-a)^{k+1}\phi_j(t)dt = \sqrt{\frac{2j+1}{h}} \sum_{l=0}^{j} \frac{(-1)^{j+l}(j+l)!h^{k+2}}{(j-l)!(l!)^2(k+l+2)}, \quad j=0,\dots,m.$$

Hence, it is not difficult to see that each entry p_{ij} of $P=[p_{ij}]_{(m+1)\times(m+1)}$ has the following form:

$$p_{ij} = h\sqrt{2i+1}\sqrt{2j+1}\sum_{k=0}^{i}\sum_{l=0}^{j} \frac{(-1)^{i+k+j+l}(i+k)!(j+l)!}{(i-k)!(k!)^2(j-l)!(l!)^2(k+l+2)(k+1)}.$$

Therefore, the operational matrix of integration for Legendre polynomials in $[a,b]$ can be written as

$$(2.5)\qquad P = \frac{h}{2}\begin{pmatrix} 1 & \gamma_1 & & & \\ -\gamma_1 & 0 & \gamma_2 & & \\ & -\gamma_2 & 0 & \gamma_3 & \\ & & \ddots & \ddots & \gamma_m \\ & & & -\gamma_m & 0 \end{pmatrix},$$

with $\gamma_k = \frac{1}{\sqrt{(2k+1)(2k-1)}}$ for $k=1,\dots,m$ (see [6]). The following lemma is stated for operational matrix of integration.

Lemma 2.4. *Let $\Psi(x)$ be the vector and P be the matrix which are defined by (2.3) and (2.5), respectively. in respectively. Then, for the Legendre operational* *matrix of integration we have*

$$\left\|\int_a^x \Psi(t)dt - P\Psi(x)\right\|_\infty = \max_{0\le i\le m}\left\|\int_a^x \phi_i(t)dt - \sum_{j=0}^m p_{ij}\phi_j(x)\right\|_2$$
$$= \sqrt{\frac{2m+1}{h}}\frac{(2m)!}{(m!)^2(m+1)h^m}\left(\frac{G((x-a)^{m+1},\phi_0,\phi_1,\dots,\phi_m)}{G(\phi_0,\phi_1,\dots,\phi_m)}\right)^{\frac{1}{2}}.$$

Proof. It is not difficult to verify that

$$\left\|\int_a^x \phi_i(t)dt - \sum_{j=0}^m p_{ij}\phi_j(x)\right\|_2$$
$$= \sqrt{\frac{2i+1}{h}}\left\|\sum_{k=0}^{i}\frac{(i+k)!}{(i-k)!(k!)^2(k+1)h^k}((x-a)^{k+1} - \sum_{j=0}^m \beta_j^k\phi_j(x))\right\|_2,$$

for $i = 0,\dots,m$, where β_j^k's are defined in (2.4). On the other hand, in view of Theorem 2.3, we may conclude that

$$\left(\frac{G((x-a)^{k+1},\phi_0,\phi_1,\dots,\phi_m)}{G(\phi_0,\phi_1,\dots,\phi_m)}\right)^{\frac{1}{2}} = \left\|(x-a)^{k+1} - \sum_{j=0}^m \beta_j^k\phi_j\right\|_2.$$

We point out here that for $k = 0,1,...,m-1$,

$$\left\|(x-a)^{k+1} - \sum_{j=0}^m \beta_j^k\phi_j(x)\right\|_2 = 0.$$

Therefore, we have

$$\left\|\int_a^x \Psi(t)dt - P\Psi(x)\right\|_\infty = \sqrt{\frac{2m+1}{h}}\frac{(2m)!}{(m!)^2(m+1)h^m}\left(\frac{G((x-a)^{m+1},\phi_0,\phi_1,\dots,\phi_m)}{G(\phi_0,\phi_1,\dots,\phi_m)}\right)^{\frac{1}{2}}.$$

□

3. **Proposed strategy for solving the main problem**

In this section we discuss how Legendre polynomials can be exploited to reduce our main problem to a system of matrix equations. Then CGNE algorithm [23] is handled for solving the obtained matrix equations. Let us approximate each entry of $Y'(x) = [Y'_{ij}(x)]_{p\times q}$ in (1.1) by the Legendre polynomials. Consequently, we have

$$Y'_{ij}(x) \approx \mathcal{A}_{ij}\Psi(x), \qquad i = 1,\dots,p, \quad j = 1,\dots,q, \tag{3.1}$$

where $\Psi(x)$ is defined as in (2.3), $\mathcal{A}_{ij} \in \mathbb{R}^{1\times(m+1)}$ are the unknown row vectors to be determined and m is the order of the Legendre polynomial. It is not difficult to see that

$$Y^{'}(x) \approx \begin{pmatrix} \mathcal{A}_{11}\Psi(x) & \dots & \dots & \mathcal{A}_{1q}\Psi(x) \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \mathcal{A}_{p1}\Psi(x) & \dots & \dots & \mathcal{A}_{pq}\Psi(x) \end{pmatrix}$$

$$= \begin{pmatrix} \mathcal{A}_{11} & \dots & \dots & \mathcal{A}_{1q} \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \mathcal{A}_{p1} & \dots & \dots & \mathcal{A}_{pq} \end{pmatrix} \begin{pmatrix} \begin{pmatrix} \phi_0(x) \\ \vdots \\ \phi_m(x) \end{pmatrix} & & & 0_{(m+1)\times 1} \\ & \ddots & & \\ & & \ddots & \\ 0_{(m+1)\times 1} & & & \begin{pmatrix} \phi_0(x) \\ \vdots \\ \phi_m(x) \end{pmatrix} \end{pmatrix}.$$

For simplicity, we set

$$\overline{\mathcal{A}} = \begin{pmatrix} \mathcal{A}_{11} & \dots & \dots & \mathcal{A}_{1q} \\ \vdots & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \mathcal{A}_{p1} & \dots & \dots & \mathcal{A}_{pq} \end{pmatrix}.$$

Therefore, we have

$$Y^{'}(x) \approx \overline{\mathcal{A}}(I_q \otimes \Psi(x)). \tag{3.2}$$

Moreover, the definition of the operational matrix of integration P implies that

$$Y(x) \approx \overline{\mathcal{A}}(I_q \otimes P\Psi(x)) + Y(a). \tag{3.3}$$

By Substituting Eqs. (3.2) and (3.3) in (1.1), we derive

$$\overline{\mathcal{A}}(I_q \otimes \Psi(x)) = A(x)\overline{\mathcal{A}}(I_q \otimes P\Psi(x)) + A(x)Y(a) + B(x). \tag{3.4}$$

Now, for evaluating the unknown coefficients in (3.4), we collocate this equation at $(m+1)$ points. The suitable collocation points, the roots of

$$(1 - (\frac{2(x-a)-h}{h})^2)T_m^{\prime}(\frac{2(x-a)-h}{h}),$$

(the Chebyshev-Gauss-Lobatto nodes in $[a,b]$) are utilized where T_m stands for the well-known Chebyshev polynomial of order m. These $m+1$ roots that we use them as collocation knots are defined by

$$t_{i+1} = \frac{h}{2}(\cos(\frac{i\pi}{m}) + 1) + a, \quad i = 0, \dots, m,$$

which are all in $[a,b]$. By replacing the above knots in (3.4), we reach the following coupled linear matrix equations

$$\overline{\mathcal{A}}C_i = D_i\overline{\mathcal{A}}E_i + F_i + G_i, \qquad i = 1,2,\dots,m+1,$$

where $C_i = I_q \otimes \Psi(t_i)$, $D_i = A(t_i)$, $E_i = I_q \otimes P\Psi(t_i)$, $F_i = A(t_i)Y(a)$ and $G_i = B(t_i)$.

Therefore, in order to numerically solve the problem (1.1), we may solve the following coupled linear matrix equation

$$XC_i - D_iXE_i = H_i, \quad i = 1,2,\dots,m+1, \tag{3.5}$$

where $H_i = F_i + G_i$, C_i, D_i and E_i are constant matrices and the unknown matrix $X := \overline{\mathcal{A}}$ is to be determined.

Using the following relation (see [4]),

$$\operatorname{vec}(AXB) = (B^T \otimes A)\operatorname{vec}(X),$$

it can be found that the coupled matrix equations (3.5) are equivalent to the following linear system

$$\begin{pmatrix} C_1^T \otimes I_p - E_1^T \otimes D_1 \\ \vdots \\ C_{m+1}^T \otimes I_p - E_{m+1}^T \otimes D_{m+1} \end{pmatrix} \operatorname{vec}(X) = \begin{pmatrix} \operatorname{vec}(H_1) \\ \vdots \\ \operatorname{vec}(H_{m+1}) \end{pmatrix}, \tag{3.6}$$

where I_p is the identity matrix of order p. The next theorem shows that under a mild condition the coefficient matrix of the system (3.6) is nonsingular.

Theorem 3.1. *For sufficiently small h, the coefficient matrix of the system (3.6) is nonsingular.*

Proof. Let us define the matrices S and $T(h)$ as follows:

$$\begin{aligned} S &:= \begin{pmatrix} C_1^T \otimes I_p \\ \vdots \\ C_{m+1}^T \otimes I_p \end{pmatrix} = \begin{pmatrix} (I_q \otimes \Psi^T(t_1)) \otimes I_p \\ \vdots \\ (I_q \otimes \Psi^T(t_{m+1})) \otimes I_p \end{pmatrix} \\ &= \begin{pmatrix} I_q \otimes \Psi^T(t_1) \\ \vdots \\ I_q \otimes \Psi^T(t_{m+1}) \end{pmatrix} \otimes I_p, \end{aligned}$$

$$T(h) := \begin{pmatrix} E_1^T \otimes D_1 \\ \vdots \\ E_{m+1}^T \otimes D_{m+1} \end{pmatrix}.$$

The coefficient matrix of (3.6) can be written as $W = S - T(h)$. We first show that the matrix S is nonsingular. According to Corollary 13.13 in [17], S is nonsingular if and only the matrix

$$U = \begin{pmatrix} I_q \otimes \Psi^T(t_1) \\ \vdots \\ I_q \otimes \Psi^T(t_{m+1}) \end{pmatrix},$$

is nonsingular. It is easy to verify that if U is singular, then there exists a nonzero vector $v = [v_0, v_1, \ldots, v_m]^T$ such that $\Psi^T(t_i)v = 0$ ($i = 1, 2, \ldots, m+1$). This shows that the polynomial $q(x) = \Psi^T(x)v$, which is of degree m, has $m+1$ distinct zeros which is a contradiction. From (2.5), it is apparent that $T(h) \to 0$ as h tends to zero. Therefore, for sufficiently small h we have $\|S^{-1}T(h)\| < 1$, for every induced matrix norm. By Lemma 2.3.3 in [15], $W = S(I - S^{-1}T(h))$ is nonsingular. $\square$

Here, we would like to point out that the above theorem reveals that the linear system (3.6) is consistent and it has a unique solution. Note that the linear system (3.6) is equivalent to the coupled matrix equations (3.5). Thence, we may infer that the coupled matrix equations (3.5) have a unique solution.

Throughout the paper we assume that the parameter h is sufficiently small such that the matrix W is nonsingular. The classical methods such as the GMRES method [22] can be applied for solving the above linear system. However, the size of the coefficient matrix of the linear system (3.6) is $pq(m+1)$ and it may become too large even for moderate values of p, q and m. Therefore, we focus on the coupled matrix equations (3.5) and implement Algorithm 3.2 in [23] to solve it instead of solving the linear system (3.6).

3.1. **Solving the coupled linear matrix equations.** In this subsection, we first review the CGNE algorithm proposed by Zhang in [23] to solve the general coupled matrix equations $\mathcal{M}(X) = H$ where

$$\mathcal{M}(X) = (M_1(X), M_2(X), \ldots, M_p(X)),$$

in which

$$M_i(X) = A_{i1}X_1B_{i1} + A_{i2}X_2B_{i2} + \cdots + A_{ip}X_pB_{ip}, \quad i = 1, 2, \ldots, p,$$

$H = (H_1, H_2, \ldots, H_p)$ and $X = (X_1, X_2, \ldots, X_p)$. Here, $A_{ij} \in \mathbb{R}^{m\times m}$, $B_{ij} \in \mathbb{R}^{n\times n}$, $H_i \in \mathbb{R}^{m\times n}$ for $i, j = 1, 2, \ldots, p$, are given matrices and $X_j \in \mathbb{R}^{m\times n}$ are unknown matrices. By defining $\mathcal{M}^T$ as

$$\mathcal{M}^T(X) = (\tilde{M}_1(X), \tilde{M}_2(X), \ldots, \tilde{M}_p(X))$$

with

$$\tilde{M}_i(X) = A_{1i}^TX_1B_{1i}^T + A_{2i}^TX_2B_{2i}^T + \cdots + A_{pi}^TX_pB_{pi}^T, \quad i = 1, 2, \ldots, p,$$

the CGNE algorithm for solving $\mathcal{M}(X) = H$ can be written as Algorithm 1 (for more details see [23]).

CGNE for solving $\mathcal{M}(X) = H$.

```
1. Choose an arbitrary matrix $X^{(0)}$ and a tolerance $\epsilon$.
2. Compute $R^{(0)} = H - \mathcal{M}(X^{(0)})$ and $P^{(0)} = \mathcal{M}^T(R^{(0)})$.
3. For $k = 0, 1, \ldots,$ Do:
4.     $\alpha_k := \|R^{(k)}\|^2/\|P^{(k)}\|^2$
5.     $X^{(k+1)} := X^{(k)} + \alpha_k P^{(k)}$
6.     $R^{(k+1)} := R^{(k)} - \alpha_k \mathcal{M}(P^{(k)})$
7.     If $\|R^{(k+1)}\| < \epsilon$ , then stop.
8.     $\beta_k := \|R^{(k+1)}\|^2/\|R^{(k)}\|^2$
9.     $P^{(k+1)} := \mathcal{M}^T(R^{(k+1)}) + \beta_k P^{(k)}$
10. EndDo
```

In the sequel, we show that one can employ Algorithm 1 to solve (3.5). To do this we define the linear operator $\mathcal{M}$ as

$$\begin{aligned} \mathcal{M} : \mathbb{R}^{p\times q(m+1)} &\to \mathbb{R}^{p\times q} \times \cdots \times \mathbb{R}^{p\times q}, \\ X &\mapsto \mathcal{M}(X) = (M_1(X), \ldots, M_p(X)), \end{aligned}$$

where $M_i(X) = XC_i - D_i X E_i$, $i = 1, 2, \ldots, m+1$. Therefore, using the linear operator $\mathcal{M}(X)$, the matrix equations (3.5) can be reformulated in the following form

$$\mathcal{M}(X) = H,$$

where $H = (H_1, \ldots, H_{m+1})$ and $H_i \in \mathbb{R}^{p\times q}$, $i = 1, 2, \ldots, m+1$. In this case the linear operator $\mathcal{M}^T$ is defined as

$$\begin{aligned} \mathcal{M}^T : \mathbb{R}^{p\times q} \times \cdots \times \mathbb{R}^{p\times q} &\to \mathbb{R}^{p\times q(m+1)}, \\ Y = (Y_1, \ldots, Y_{m+1}) &\mapsto \mathcal{M}^T(Y), \end{aligned}$$

where

$$\mathcal{M}^T(Y) = \sum_{i=1}^{m+1} \left(Y_i C_i^T - D_i^T Y_i E_i^T \right).$$

Now, with the above notations Algorithm 1 can be used to solve (3.5).

3.2. **Implementing the method.** For solving (1.1), we use a step-by-step manner. More precisely, we first choose a step length $h \neq 0$. Then, starting with the given initial values $x_0 := a$, $Y_0 := Y(x_0)$ and considering the equidistant points $x_i = x_0 + ih$, $i = 1, 2, \ldots$, we successively compute the approximate solution $Z(x)$ to $Y(x)$ on $[x_i, x_{i+1})$ for $i = 0, 1, \ldots, [\frac{b-a}{h}] - 1$. That is, we employ the following algorithm.

4. Numerical experiments

In this section, three numerical examples are presented to demonstrate the efficiency of Algorithm 2 for solving (1.1). All the numerical experiments are performed using Mathematica 6 with a machine unit round off precision

```
Algorithm for solving (1.1).
1. Choose a step length $h \neq 0$. Set $n = [\frac{b-a}{h}]$.
2. Set $Z_0 = Y_0$ and $x_0 = a$.
3. For $i = 0, 1, \ldots, n-1$, Do
4.      Compute the approximate solution $Z(x)$ of $Y(x)$ on the
subinterval $x_i \leq x < x_{i+1}$ by solving the following linear matrix differential
        equation with the approach described in Subsection 3.1,
            $\begin{cases} Z'(x) = A(x)Z(x) + B(x), & x_i \leq x < x_{i+1}, \\ Z(x_i) = Z_i. \end{cases}$
5.      Set $Z_{i+1} = Z(x_{i+1})$ and goto 3.
6. EndDo.
```

of around 10^{-16}. We would like to point out that, in Algorithm 2 at each subinterval $[x_l, x_{l+1}]$, $l = 0, 1, \ldots, n-1$, Algorithm 1 is applied with the stopping criterion $\|R^{(k+1)}\| < 10^{-16}$. To compare the approximate solution of the problem computed by the proposed method $Z(x) = [Z_{ij}(x)]_{p\times q}$ and the exact solution $Y(x) = [Y_{ij}(x)]_{p\times q}$ we define the following two notations

$$Err = \max_{i,j} \|Y_{ij} - Z_{ij}\|_\infty = \max_{i,j} \max_{x_l \leq x \leq x_{l+1}} |Y_{ij}(x) - Z_{ij}(x)|, \tag{4.1}$$

$$E(x) = \|Y(x) - Z(x)\|_F. \tag{4.2}$$

In continuation, we report the application of Algorithm 2 for solving (1.1).

Example 4.1. Consider the following first-order linear matrix differential equation (see [8])

$$\begin{cases} Y'(x) = \dfrac{1}{x^3 - x - 1} \begin{pmatrix} 2x^2 - 1 & x^2 - 2x - 1 \\ -x - 1 & x^3 + x^2 - x - 1 \end{pmatrix} Y(x), & 0 \leq x \leq 1, \\ Y(0) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}. \end{cases} \tag{4.3}$$

It can be verified that the exact solution of (4.3) is

$$Y(x) = \begin{pmatrix} e^x \\ xe^x \end{pmatrix}.$$

Example 4.2. Consider the following problem (see [8])

$$\begin{cases} Y'(x) = A(x)Y(x) + B(x), \\ Y(0) = \begin{pmatrix} 3 & 0 \\ 1 & 1 \end{pmatrix}, & x \in [0, 1], \end{cases}$$

where

$$A(x) = \begin{pmatrix} 1 & -1 \\ 1 & e^x \end{pmatrix}, \qquad B(x) = \begin{pmatrix} -3e^{-x} - 1 & 2 - 2e^{-x} \\ -3e^{-x} - 2 & 1 - 2\cosh(x) \end{pmatrix},$$

which has the exact solution

$$Y(x)=\begin{pmatrix} 2e^{-x}+1 & e^{-x}-1 \\ e^{-x} & 1 \end{pmatrix}.$$

Example 4.3. Consider the following problem

$$\begin{cases} Y'(x)=A(x)Y(x)+B(x), & \\ Y(0)=\begin{pmatrix} 1/8 & 0 \\ 1 & 1/8 \end{pmatrix}, & x\in[0,1], \end{cases}$$

where

$$A(x)=\begin{pmatrix} x & 0 \\ 0 & 1 \end{pmatrix}, \qquad B(x)=\begin{pmatrix} b_{11} & 0 \\ -1 & b_{22} \end{pmatrix},$$

and b_{11} and b_{22} defined as

$$b_{11}=\begin{cases} \frac{-(1-2x)^2}{8}(2x^2-6-x), & x\geq 0.5, \\ \frac{(1-2x)^2}{8}(2x^2-6-x), & x<0.5, \end{cases}$$

$$b_{22}=\begin{cases} \frac{-(1-2x)^2}{8}((2x-7)\cos x+(2x-1)\sin x), & x\geq 0.5, \\ \frac{(1-2x)^2}{8}((2x-7)\cos x+(2x-1)\sin x), & x<0.5. \end{cases}$$

The exact solution of the problem is

$$Y(x)=\begin{pmatrix} y_{11} & 0 \\ 1 & y_{22} \end{pmatrix},$$

in which

$$y_{11}(x)=\begin{cases} (x-0.5)^3, & x\geq 0.5, \\ (0.5-x)^3, & x<0.5, \end{cases}$$

and

$$y_{22}=\begin{cases} (x-0.5)^3\cos x, & x\geq 0.5, \\ (0.5-x)^3\cos x, & x<0.5. \end{cases}$$

In Table 1, we report the approximate solution of the first example computed by Algorithm 2 in each subinterval for $m=4$ and $h=0.1$. The comparison results between the Spline method [8] and Algorithm 2 are presented in Tables 2 and 3. In fact, these tables compare Err (defined by Eq. (4.1)) of the computed solutions by the Spline method and Algorithm 2. For more investigation, the function $\log_{10}E(x)$ (defined by (4.2)) in the interval $[0,1]$ of the proposed method for different values of m is displayed in Figure 1. As we see the computed solution by the proposed method is in good agreement with the exact solution. Moreover, we observe that the proposed method is superior to the Spline method.

Similar to Example 4.1, numerical results for Example 4.2 and 4.3 are presented in Table 4, Figure 2 and Table 5, Figure 3, respectively. As we see the proposed method is more effective than the Spline method proposed by Defez et al. in [8].

TABLE 1. Approximate solution computed by the proposed method for Example 4.1 ($m = 4$ and $h = 0.1$).

$[x_i, x_{i+1}]$	Approximation (computed by Algorithm 2)
$[0, 0.1]$	$\begin{pmatrix} 1 + x + 0.500007x^2 + 0.166472x^3 + 0.043819x^4 \\ 0.999999x + 1.000036x^2 + 0.499031x^3 + 0.177439x^4 \end{pmatrix}$
$[0.1, 0.2]$	$\begin{pmatrix} 1 + 0.999978x + 0.500305x^2 + 0.164609x^3 + 0.048429x^4 \\ 0.999887x + 1.001553x^2 + 0.489536x^3 + 0.200942x^4 \end{pmatrix}$
$[0.2, 0.3]$	$\begin{pmatrix} 1.000010 + 0.999805x + 0.501558x^2 + 0.160511x^3 + 0.053523x^4 \\ 0.000050 + 0.998991x + 1.008061x^2 + 0.468244x^3 + 0.227425x^4 \end{pmatrix}$
$[0.3, 0.4]$	$\begin{pmatrix} 1.000058 + 0.999179x + 0.504639x^2 + 0.153731x^3 + 0.059152x^4 \\ 0.000304 + 0.995678x + 1.024375x^2 + 0.432329x^3 + 0.257256x^4 \end{pmatrix}$
$[0.4, 0.5]$	$\begin{pmatrix} 1.000222 + 0.997557x + 0.510665x^2 + 0.143749x^3 + 0.065374x^4 \\ 0.001190 + 0.986926x + 1.056893x^2 + 0.378452x^3 + 0.290848x^4 \end{pmatrix}$
$[0.5, 0.6]$	$\begin{pmatrix} 1.000661 + 0.9940723x + 0.521049x^2 + 0.129966x^3 + 0.072250x^4 \\ 0.003603 + 0.967783x + 1.113956x^2 + 0.302683x^3 + 0.328658x^4 \end{pmatrix}$
$[0.6, 0.7]$	$\begin{pmatrix} 1.001663 + 0.987439x + 0.537552x^2 + 0.111692x^3 + 0.079850x^4 \\ 0.009204 + 0.930674x + 1.206296x^2 + 0.200406x^3 + 0.371204x^4 \end{pmatrix}$
$[0.7, 0.8]$	$\begin{pmatrix} 1.003706 + 0.975819x + 0.562356x^2 + 0.088134x^3 + 0.088249x^4 \\ 0.020837 + 0.864516x + 1.347551x^2 + 0.066215x^3 + 0.419064x^4 \end{pmatrix}$
$[0.8, 0.9]$	$\begin{pmatrix} 1.007551 + 0.956671x + 0.598143x^2 + 0.058380x^3 + 0.097534x^4 \\ 0.043102 + 0.753610x + 1.554893x^2 - 0.106214x^3 + 0.472884x^4 \end{pmatrix}$
$[0.9, 1]$	$\begin{pmatrix} 1.014349 + 0.926561x + 0.648192x^2 + 0.021383x^3 + 0.107797x^4 \\ 0.083129 + 0.576262x + 1.849759x^2 - 0.324255x^3 + 0.533387x^4 \end{pmatrix}$

TABLE 2. Comparison of *Err* error for the Spline method [8] and the proposed method for Example 4.1 ($m = 4$ and $h = 0.1$).

Interval	[0, 0.1]	[0.1,0.2]	[0.2,0.3]	[0.3,0.4]	[0.4,0.5]	[0.5,0.6]	[0.6,0.7]	[0.7,0.8]	[0.8,0.9]	[0.9,1]
Algorithm 2	1.75×10^{-9}	3.97×10^{-9}	6.7×10^{-9}	1.01×10^{-8}	1.4×10^{-8}	1.9×10^{-8}	2.5×10^{-8}	3.3×10^{-8}	4.1×10^{-8}	5.2×10^{-8}
Spline Algorithm [8]	1.14×10^{-7}	2.62×10^{-7}	4.51×10^{-7}	6.89×10^{-7}	9.89×10^{-7}	1.36×10^{-6}	1.82×10^{-6}	2.37×10^{-6}	3.05×10^{-6}	3.86×10^{-6}

TABLE 3. Comparison of Err error for the Spline method [8] and the proposed method for Example 4.1 ($m = 5$ and $h = 0.1$).

Interval	[0, 0.1]	[0.1,0.2]	[0.2,0.3]	[0.3,0.4]	[0.4,0.5]
Algorithm 2	9.56×10^{-12}	2.15×10^{-11}	3.63×10^{-11}	5.45×10^{-11}	7.68×10^{-11}
Spline Algorithm [8]	1.80×10^{-9}	4.09×10^{-9}	7.00×10^{-9}	1.07×10^{-8}	1.53×10^{-8}
Interval	[0.5,0.6]	[0.6,0.7]	[0.7,0.8]	[0.8,0.9]	[0.9,1]
Algorithm 2	1.04×10^{-10}	1.36×10^{-10}	1.75×10^{-10}	2.22×10^{-10}	2.76×10^{-10}
Spline Algorithm [8]	2.10×10^{-8}	2.80×10^{-8}	3.65×10^{-8}	4.67×10^{-8}	5.90×10^{-8}

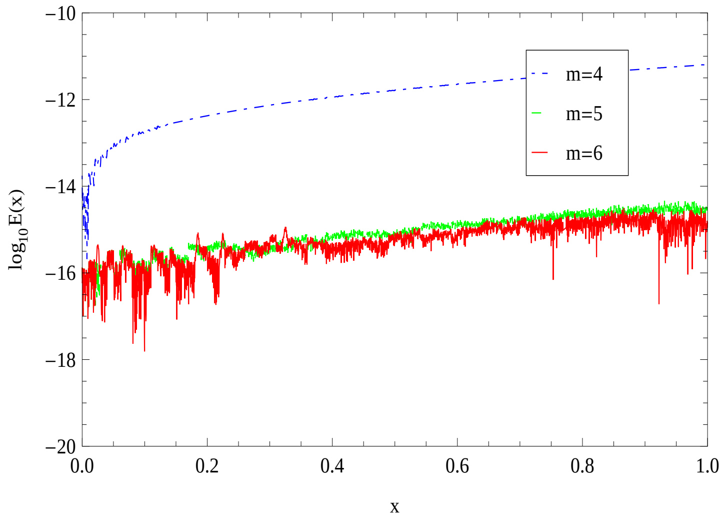

FIGURE 1. $\log_{10} E(x)$ for Example 4.1 ($m = 4, 5, 6$ and $h = 0.01$).

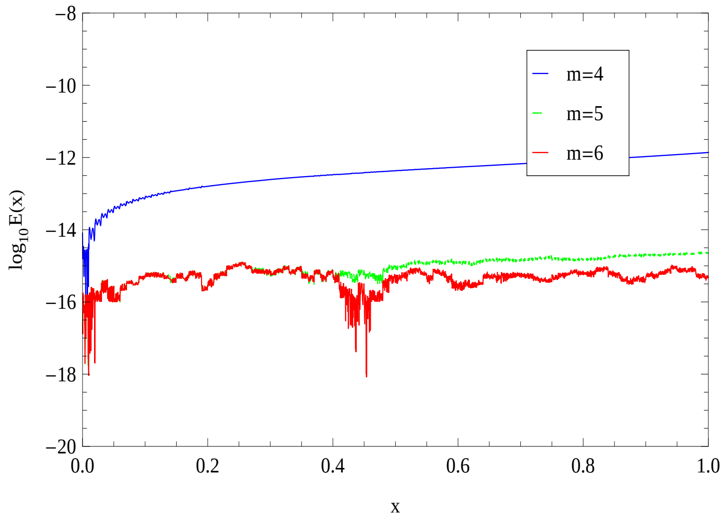

FIGURE 2. $\log_{10} E(x)$ for Example 4.2 ($m = 4, 5, 6$ and $h = 0.01$).

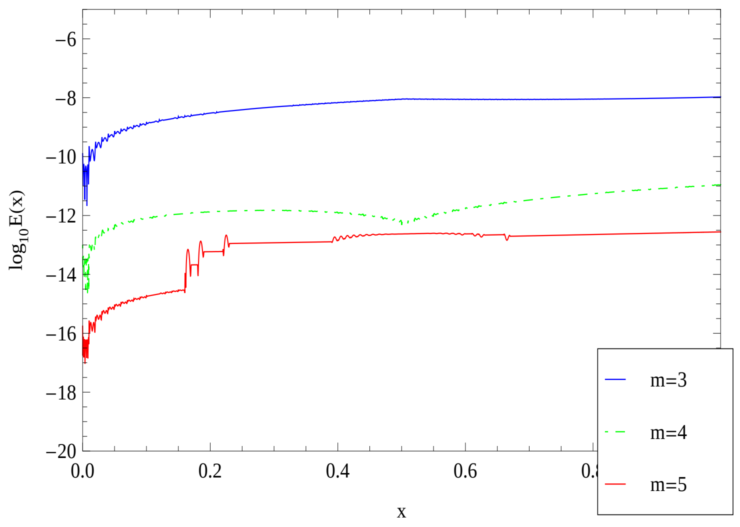

FIGURE 3. $\log_{10} E(x)$ Example 4.3 ($m = 3, 4, 5$ and $h = 0.01$).

TABLE 4. Comparison of Err error for the Spline method [8] and the proposed method for Example 4.2 ($m = 4, 5$ and $h = 0.1$).

Interval	[0, 0.1]	[0.1,0.2]	[0.2,0.3]	[0.3,0.4]	[0.4,0.5]	[0.5,0.6]	[0.6,0.7]	[0.7,0.8]	[0.8,0.9]	[0.9,1]
Algorithm 2 ($m = 4$)	6.28×10^{-10}	1.22×10^{-9}	1.78×10^{-9}	2.28×10^{-9}	2.73×10^{-9}	3.09×10^{-9}	3.31×10^{-9}	3.41×10^{-9}	3.26×10^{-9}	2.80×10^{-9}
Spline Algorithm [8] ($m = 4$)	5.06×10^{-8}	1.02×10^{-7}	1.54×10^{-7}	2.10×10^{-7}	2.70×10^{-7}	3.38×10^{-7}	4.19×10^{-7}	5.21×10^{-7}	6.59×10^{-7}	8.51×10^{-7}
Algorithm 2 ($m = 5$)	2.79×10^{-12}	5.45×10^{-12}	7.94×10^{-12}	1.02×10^{-11}	1.22×10^{-11}	1.39×10^{-11}	1.50×10^{-11}	1.55×10^{-11}	1.50×10^{-11}	1.32×10^{-11}
Spline Algorithm [8] ($m = 5$)	6.75×10^{-10}	1.36×10^{-9}	2.06×10^{-9}	2.80×10^{-9}	3.60×10^{-9}	4.50×10^{-9}	5.57×10^{-9}	6.93×10^{-9}	8.75×10^{-9}	1.13×10^{-8}

TABLE 5. Comparison of Err error for the Spline method [8] and the proposed method for Example 4.3 ($m = 4, 5$ and $h = 0.1$).

Interval	[0, 0.1]	[0.1,0.2]	[0.2,0.3]	[0.3,0.4]	[0.4,0.5]	[0.5,0.6]	[0.6,0.7]	[0.7,0.8]	[0.8,0.9]	[0.9,1]
Algorithm 2 ($m = 4$)	7.98×10^{-9}	1.28×10^{-8}	1.46×10^{-8}	1.76×10^{-8}	1.65×10^{-8}	1.70×10^{-8}	3.31×10^{-8}	5.35×10^{-8}	7.81×10^{-8}	1.06×10^{-7}
Algorithm 2 ($m = 5$)	1.75×10^{-10}	3.73×10^{-10}	5.90×10^{-10}	8.24×10^{-10}	1.07×10^{-9}	1.23×10^{-9}	1.19×10^{-9}	1.16×10^{-9}	1.16×10^{-9}	1.18×10^{-9}

5. Conclusion

The properties of the Legendre basis together with a collocation method have been utilized to numerically solve a class of first-order matrix differential problems by a new framework. The proposed approach reduces the main problem to the coupled linear matrix equations. An iterative algorithm is exploited to solve the obtained coupled linear matrix equations. Numerical examples have illustrated to show the efficiency and applicably of our method.

Acknowledgments

The authors would like to express their heartfelt gratitude to the anonymous referees for their valuable suggestions and constructive comments which have improved the quality of the paper.

References

[1] A. Üsküplü Altnbaşak and M. Demiralp, Solutions to linear matrix ordinary differential equations via minimal, regular, and excessive space extension based universalization, *J. Math. Chem.* **4**8 (2010), no. 2, 253–265.

[2] P. W. Atkins and J. C. de Paula, Physical Chemistry, 7th Edition, Oxford University Press, Oxford, 2002.

[3] S. Barnett, Matrices in Control Theory, Van Nostrand Reinhold Co., London-New York-Toronto, 1971.

[4] D. S. Bernstein, Matrix Mathematics: theory, facts, and formulas, Second edition, Princeton University Press, Princeton, 2009.

[5] A. Canada, P. Drbek, A. Fonda (eds.), Handbook of Differential Equations: Ordinary Differential Equations, Elsevier-North-Holland, Amsterdam, 2006.

[6] R. Y. Chang and M. L. Wang, Shifted Legendre direct method for variational problems, *J. Optim. Theory Appl.* **3**9 (1983), no. 2, 299–307.

[7] T. S. Chihara, An Introduction to Orthogonal Polynomials, Gordon and Breach, New York-London-Paris, 1978.

[8] E. Defez, A. Hervás, J. Ibáñez and M. M. Tung, Numerical solutions of matrix differential models using higher-order matrix splines, *Mediterr. J. Math.* **9** (2012), no. 4, 865–882.

[9] E. Defez, A. Hervás, L. Soler and M. M. Tung, Numerical solutions of matrix differential models cubic spline II, *Math. Comput. Modelling.* **4**6 (2007) 657–669.

[10] E. Defez, L. Soler, A. Hervás and C. Santamaría, Approximating and computing nonlinear matrix differential models, *Math. Comput. Modelling.* **5**5 (2012), no. (7-8), 2012–2022.

[11] E. Defez, M. M. Tung, J. J. Ibáñez and J. Sastre, Approximating and computing nonlinear matrix differential models, *Math. Comput. Modelling.* **5**5 (2012), no. 7-8, 2012–2022.

[12] L. D. Faddeyev, The inverse problem in the quantum theory of scattering, *J. Mathematical Phys.* **4** (1963), no. 1, 72–104.

[13] T. M. Flett, Differential Analysis, Cambridge University Press, Cambridge-New York, 1980.

[14] F. Fakhar-Izadi and M. Dehghan, The spectral methods for parabolic Volterra integro-differential equations, *J. Comput. Appl. Math.* **2**35 (2011), no. 14, 4032–4046.

[15] G. H. Golub and C. F. V. Loan, Matrix Computations, second ed., The Johns Hopkins University Press, Baltimore, 1989.

[16] E. Kreyszig, Introductory Functional Analysis with Applications, John Wiley & Sons, Inc., 1978.

[17] A. J. Laub, Matrix Analysis for Scientists & Engineers, SIAM, Philadelphia, 2005.

[18] A. Lotfi, M. Dehghan and S. A. Yousefi, A numerical technique for solving fractional optimal control problems, *Comput. Math. Appl.* **6**2 (2011), no. 3, 1055–1067.

[19] D. A. McQuarrie, Quantum Chemistry, 2nd edn., University Science Books, California, 2008.

[20] A. C. Norris, Computational Chemistry, An Introduction to Numerical Methods, John Wiley & Sons, Ltd., Chichester, 1981.

[21] W. T. Reid, Riccati differential equations, Mathematics in Science and Engineering, 86, Academic Press, New York-London, 1972.

[22] Y. Saad, Iterative Methods for Sparse linear Systems, PWS Press, New York, 1995.

[23] J. J. Zhang, A note on the iterative solutions of the general coupled matrix equation, *App. Math. Comput.* **2**17 (2011), no. 22, 9380–9386.

[24] H. Zheng and W. Han, On some discretization methods for solving a linear matrix ordinary differential equation, *J. Math. Chem.* **4**9 (2011), no. 5, 1026–1041.

(Ahmad Golbabai) School of Mathematics, Iran University of Science and Technology, P.O. Box 16846-13114, Tehran, Iran.
E-mail address: `golbabai@iust.ac.ir`

(Samaneh Panjeh Ali Beik) School of Mathematics, Iran University of Science and Technology, P.O. Box 16846-13114, Tehran, Iran.
E-mail address: `panjehali@iust.ac.ir`

(Davod Khojasteh Salkuyeh) Faculty of Mathematical Sciences, University of Guilan, Rasht, Iran.
E-mail address: `khojasteh@guilan.ac.ir; salkuyeh@gmail.com`