Shahid Bahonar University of Kerman
Journal of Mahani Mathematical Research
Print ISSN: 2251-7952 Online ISSN: 2645-4505

EXTENDED BLOCK HESSENBERG METHOD FOR LARGE-SCALE SYLVESTER DIFFERENTIAL MATRIX EQUATIONS

A. Tajaddini

Article type: Research Article
(Received: 07 December 2023, Received in revised form 05 April 2024)
(Accepted: 22 May 2024, Published Online: 23 May 2024)

Abstract. In this paper, we consider large-scale low-rank Sylvester differential matrix equations. We present two iterative methods for the approximate solution of such differential matrix equations. The first approach is based on the integral expression of the exact solution which exploits an extended block Krylov subspace method to compute the exponential of a matrix times a block of vectors. In the second method, we first project the initial value problem onto an extended block Krylov subspace and acquire a low-dimensional Sylvester differential matrix equation with a low-rank constant term. Then the reduced Sylvester differential matrix equation is solved by the backward differentiation formula method (BDF) and the derived solution is used to construct the low-rank approximate solutions of the original initial value problem. The iterative approaches are followed until some certain accuracy is obtained. We give some theoretical results and some numerical examples to show the efficiency of the proposed methods.

Keywords: Sylvester differential matrix equations, Extended block Hessenberg, Low-rank.
2020 MSC: Primary 34Axx, 65Fxx, 15Axx.

1. Introduction

This paper presents two iterative methods for the numerical solutions of the low-rank Sylvester differential matrix equation (in-short LR-SDE) of the form

$$\begin{cases} \dot{X}(t) = AX(t) + X(t)B + EF^\top, \quad t \in [t_0, T] \\ X(t_0) = X_0, \end{cases} \tag{1}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{s\times s}$ are nonsingular matrices and $E \in \mathbb{R}^{n\times r}$, $F \in \mathbb{R}^{s\times r}$ are full of rank with $r \ll min(n, s)$. The low-rank Lyapunov differential matrix equation corresponds to the symmetric case when $B = A^\top$ and $F = E$.

Differential Sylvester and Lyapunov equations come into view in different fields of applied mathematics, for instance, control theory, design theory, model reduction problems, and robust control problems [3, 6, 7].

✉ atajadini@uk.ac.ir, ORCID: 0000-0003-2513-194X
https://doi.org/10.22103/jmmr.2024.22641.1548
Publisher: Shahid Bahonar University of Kerman
How to cite: A. Tajaddini, *Extended block Hessenberg method for large-scale Sylvester differential matrix equations*, J. Mahani Math. Res. 2024; 13(2): 383 - 409.

As far as we know, although these differential matrix equations are important especially when their matrix coefficients are large-scale, few numerical methods have been proposed to find their approximate solution. To obtain the approximate solution of such differential matrix equations, methods such as backward differentiation formula (BDF) and/or Rosenbrock have been presented in [11, 12]. These methods usually suffer from a problem of storage. To overcome this problem, the authors in [10, 19] combined Krylov subspace methods with Taylor series expansion or BDF methods. Bouhamidi et al. [15] combined the constant solution approach with Krylov subspace methods to achieve an approximate solution of the Sylvester algebraic equation, and then form an approximate solution of the large-scale Sylvester differential matrix equation. Moreover, there is a large variety of methods to approximate the solution of large-scale matrix differential equations such as differential Lyapunov, Sylvester, and Riccati matrix equations, for further details see [19, 20, 28, 29]. The basic idea used in these methods is to apply an extended Krylov subspace and then utilize the Galerkin-type orthogonality condition.

In [20], M. Hached et al. introduced some numerical methods for computing approximate solutions to some large differential linear matrix equations. They solved differential generalized Sylvester matrix equations with full-rank right-hand sides using global Galerkin and norm-minimization approaches. In addition, they examined large differential Lyapunov matrix equations with low-rank right-hand sides and applied the extended global (or extended block) Arnoldi process to generate low-rank approximate solutions, see [19, 20]. In [28], E. M. Sadek et al. proposed iterative methods based on the global extended Krylov subspace method to solve large-scale differential Sylvester matrix equations with low-rank right-hand sides. In [29], L. Sadek et al. studied the extended block Arnoldi-based algorithms for solving low-rank large-scale differential nonsymmetric Stein matrix.

In this paper, motivated by the merits of the extended block Krylov subspaces, we present some numerical methods based on the block Hessenberg process to solve the large-scale Sylvester differential matrix equation with the low-rank constant term (1). The extended block Krylov subspace contains more information than the classical block Krylov subspace because it is enriched by A^{-1}. This is due to the fact that extended block Krylov subspaces can enlarge the search space, which makes all Krylov subspaces associated with each right-hand side contained. To generate a basis of the subspace, we should apply a process for constructing a basis. Since the extended block Hessenberg process requires less arithmetic operations and storage than the extended block Arnoldi process, we use it.

Herein, we give two approaches based on the extended block Hessenberg process. The first is based on approximating the exponential matrix in the exact solution exploiting the extended block Krylov method named the exponential Hessenberg method (EHess-exp). The other is based on a low-rank

approximation of the solution of the corresponding Sylvester differential matrix equation using the extended block Hessenberg algorithm, which is called the low-rank Sylvester differential extended block Hessenberg method (LRSD-EBHess). These approaches are based on the extended block Hessenberg process have been developed for the first time and have had a very good performance for large-scale problems.

The rest of the paper is structured as follows. First, we give some definitions and notations and recall some properties. Then, we describe and give some properties of the extended block Hessenberg process with maximum strategy in Section 2. In Section 3, we define the EHess-exp method based on the extended block Krylov subspace and a quadrature method to approximate matrix exponential and compute the numerical solution of the Sylvester differential matrix equation (1). In Section 4, we show how to project the initial Sylvester differential matrix equation onto an extended block Krylov subspace to get low-rank approximate solutions in a factored form. Then, we solve the obtained low-dimensional Sylvester differential equation using the backward differentiation formula (BDF). We also establish some theoretical results on the residual and the error norm provided by the two approaches. In the last section, we report the experimental results and conclude this paper.

Throughout this work, the following definitions and notations will be used.

Definition 1.1. Assume that A is an $m \times n$ and B is $p \times q$ matrix. The Kronecker product of two matrices A and B is denoted by $A \otimes B$ and is the $pm \times qn$ block matrix:

$$(A \otimes B) = (a_{i,j}B),$$

where $a_{i,j}$ denotes the (i, j)-th element of the matrix A.

The properties of the Kronecker product incorporated in Proposition 1.2 follow immediately from the definition.

Proposition 1.2. *If $A, C \in \mathbb{R}^{n\times n}$ and $B, D \in \mathbb{R}^{m\times m}$. Then*

1 . $(A \otimes B)^\top = A^\top \otimes B^\top$.
2 . $(A \otimes B)(C \otimes D) = (AC \otimes BD)$.

The following definition introduces the notion of a left inverse for a non-square matrix.

Definition 1.3. ([30]) Suppose $A, B \in \mathbb{R}^{n\times r}$ and let $B^\top A$ be a nonsingular matrix. Then the left inverse of the matrix A is denoted by A^L and defined as

$$A^L = (B^\top A)^{-1}B^\top.$$

Observe that this left inverse satisfies $A^L A = I_r$. Note that the left inverse of a matrix is not unique. For instance, if A has full column rank, a left inverse of a matrix A is $A^L = (A^\top A)^{-1}A^\top$. This is the same as pseudo-inverse, denoted by $A^\dagger$.

The Frobenius norm of a matrix $A \in \mathbb{R}^{m\times n}$ is defined by $\|A\|_F^2 = \texttt{trace}(A^\top A)$, where $\texttt{trace}(Z)$ denotes the sum of the diagonal elements of the matrix Z.

Definition 1.4. Let the matrices $\mathbb{V}_m \in \mathbb{R}^{n\times m}$ and $\mathbb{W}_m \in \mathbb{R}^{s\times m}$ that have left inverses be given. In addition, let $A \in \mathbb{R}^{n\times s}$. Then the semi-norm of the matrix A is denoted by $|A|_{(\mathbb{V}_m,\mathbb{W}_m)}$ and is defined as

$$|A|^2_{(\mathbb{V}_m,\mathbb{W}_m)} = \|\mathbb{V}_m^L A(\mathbb{W}_m^L)^\top\|_F^2,$$

where $\mathbb{V}_m^L$ and $\mathbb{W}_m^L$ are the left inverses of the matrices $\mathbb{V}_m$ and $\mathbb{W}_m$, respectively.

Furthermore, we will refer to the maximum norm of a matrix $A \in \mathbb{R}^{m\times n}$ in the form

$$\|A\|_{\texttt{max}} = \max_{\substack{1\le i\le m\\ 1\le j\le n}} |a_{i,j}|.$$

For a bounded matrix valued function $R(t)$ defined on the interval $[t_0, T]$, the uniform convergence norm

$$\|R\|_\infty = \max_{t\in[t_0,T]} \|R(t)\|_F,$$

is considered.

2. The extended block Hessenberg process

In this section, we represent the extended block Krylov subspace and use the extended block Hessenberg process with the maximum strategy to construct a basis for it.

Before describing the extended block Hessenberg process with the maximum strategy, let us first introduce the projection subspace that is considered herein. Let $A \in \mathbb{R}^{n\times n}$ be a nonsingular matrix and $V \in \mathbb{R}^{n\times r}$ be a block vector. The extended block Krylov subspace associated with pairs (A, V) is denoted by $\mathbb{K}_m^e(A, V)$ and is defined as

$$\mathbb{K}_m^e(A, V) = \texttt{blockSpan}\left\{V, A^{-1}V, AV, A^{-2}V, \ldots, A^{m-1}V, A^{-m}V\right\}.$$

It should be noted that it can be written as a sum of two classical block Krylov subspaces $\mathcal{K}_m(A, V)$ and $\mathcal{K}_m(A^{-1}, A^{-1}V)$. The block extended Krylov subspace $\mathbb{K}_m^e(A, V)$ contains more information than the classical block Krylov subspace because it is enriched by A^{-1}. To construct a basis of $\mathbb{K}_m^e(A, V)$, we can apply the extended block Hessenberg process with the maximum strategy that is summarized in Algorithm 1. This procedure is similar to the block Hessenberg method with maximum strategy which was introduced in [4].

If the upper triangular matrices $H_{j+1,j}, j = 1, 2, \ldots, m$ are not rank-deficient, Algorithm 1 generates an $n \times 2mr$ matrix $\mathcal{V}_m = [V_1, V_2, \ldots, V_m]$ with $V_j \in \mathbb{R}^{n\times 2r}$ and block upper Hessenberg matrix $\mathcal{H}_m \in \mathbb{R}^{2mr\times 2mr}$ with $H_{i,j} \in \mathbb{R}^{2r\times 2r}$ such that the block vectors $V_1, V_2, \ldots, V_m$ form a basis of the extended block Krlov subspace $\mathbb{K}_m^e(A, V)$.

Input: A an $n \times n$ matrix, V an $n \times r$ matrix and m an integer.
Output: The matrix $\mathcal{V}_{m+1} \in \mathbb{R}^{n \times 2(m+1)r}$ with column block vectors $V_1, V_2, \ldots, V_{m+1}$ and the semi-block upper-Hessenberg $\overline{\mathcal{H}}_m \in \mathbb{R}^{2(m+1)r \times 2mr}$.

Compute the partial LU factorization of $[V, A^{-1}V]$:

$$[V_1, \Lambda] = \texttt{lu}([V, A^{-1}V]); \quad [\sim, p_1] = \texttt{max}(V_1);$$

for $j = 1, \ldots, m$ **do**
  Set $V_j^{(1)} = V_j(:, 1:r)$, and $V_j^{(2)} = V_j(:, r+1:2r)$;
  Set $\mathcal{V}_j = [\mathcal{V}_{j-1}, V_j]$, and $\widehat{U}_{j+1}^{(0)} = [AV_j^{(1)}, A^{-1}V_j^{(2)}]$;

  for $i = 1, \ldots, j$ **do**
    $H_{i,j} = (V_i(p_i, :))^{-1}\widehat{U}_{j+1}^{(i-1)}(p_i, :)$;
    $\widehat{U}_{j+1}^{(i)} = \widehat{U}_{j+1}^{(i-1)} - V_j H_{i,j}$;
  end for
end for
Compute the partial LU factorization of $\widehat{U}_{j+1}^{(i)}$ and the $(j+1)$th permutation vector p_{j+1}:

$$[V_{j+1}, H_{j+1,j}] = \texttt{lu}(\widehat{U}_{j+1}^{(i)}); \quad [\sim, p_{j+1}] = \texttt{max}(V_{j+1});$$

$=0$

Algorithm 1: The extended block Hessenberg process with maximum strategy (EBHess).

It is worth mentioning that the matrix $\mathcal{V}_m$ has a left inverse. To this end, let $p_j = [i_1^{(j)}, i_2^{(j)}, \ldots, i_{2r}^{(j)}]$ be the permutation vectors that are generated by Algorithm 1, for $j = 1, 2, \ldots, m$. We define the $n \times 2r$ matrices $P_j = [e_{i_1^{(j)}}, e_{i_2^{(j)}}, \ldots, e_{i_{2r}^{(j)}}]$ such that $P_j^\top V_j = I_{2r}$, $P_j^\top V_i = 0$, for $i = 1, 2, \ldots, j-1$ and e_j is the j-th column of the identity matrix I_n. With regard to these definitions, we get

$$\mathcal{P}_m^\top \mathcal{V}_m = \mathcal{L}_m,$$

where $\mathcal{L}_m \in \mathbb{R}^{2mr \times 2mr}$ is a unit lower-triangular matrix and $\mathcal{P}_m = [P_1, P_2, \ldots, P_m]$. As a result of this discussion, we can see that the left inverse of $\mathcal{V}_m$ (namely $\mathcal{V}_m^L$) is given by

$$\mathcal{V}_m^L = (\mathcal{P}_m^\top \mathcal{V}_m)^{-1}\mathcal{P}_m^\top.$$

For more details please refer to [1].

Let $\mathcal{T}_m \in \mathbb{R}^{2mr \times 2mr}$ be the restriction of the matrix A to the extended block Krylov subspace $\mathbb{K}_m^e(A, V)$, that is $\mathcal{T}_m = \mathcal{V}_m^L A \mathcal{V}_m$, where $\mathcal{V}_m^L$ is the left inverse of the matrix $\mathcal{V}_m$. If the matrix $\mathcal{V}_m$ is an orthonormal matrix, it is shown in [2, 31] that the matrix $\mathcal{T}_m = \mathcal{V}_m^\top A \mathcal{V}_m$ is an upper-Hessenberg matrix. With

a process similar to what was said in [1], it can be proved that the matrix $\mathcal{T}_m = \mathcal{V}_m^L A \mathcal{V}_m$ is also a block upper-Hessenberg matrix with $2r \times 2r$ blocks. Moreover, the following Hessenberg relations blue are satisfied for A:

$$A\mathcal{V}_m = \mathcal{V}_{m+1}\overline{\mathcal{T}}_m, \tag{2}$$

$$= \mathcal{V}_m\mathcal{T}_m + V_{m+1}T_{m+1,m}\mathbb{E}_m^\top, \tag{3}$$

$$\mathcal{V}_m^L A\mathcal{V}_m = \mathcal{T}_m, \tag{4}$$

where $\overline{\mathcal{T}}_m = \begin{bmatrix} \mathcal{T}_m \\ T_{m+1,m}\mathbb{E}_m^\top \end{bmatrix}$ and $\mathbb{E}_m \in \mathbb{R}^{2mr\times 2r}$ denotes the last $2r$ columns of the identity matrix I_{2mr}. Since the proof of these relations is similar to the proof of block Arnoldi relations given in [2,21], therefore, their proof is omitted.

The next proposition shows that the block columns of $\overline{\mathcal{T}}_m$ can be derived recursively from $\overline{\mathcal{H}}_m$ without requiring matrix-vector products with A. Before we discuss how the recursive relations can be obtained, let us partition the matrices $\overline{\mathcal{T}}_m$ and $\overline{\mathcal{H}}_m$ in this way

$$\overline{\mathcal{T}}_m = [T_1, T_2, \ldots, T_{2m-1}, T_{2m}], \overline{\mathcal{H}}_m = [H_1, H_2, \ldots, H_{2m-1}, H_{2m}],$$

where $T_{2j-1} = \overline{\mathcal{T}}_m\mathcal{E}_{2j-1}$, $T_{2j} = \overline{\mathcal{T}}_m\mathcal{E}_{2j}$, $H_{2j-1} = \overline{\mathcal{H}}_m\mathcal{E}_{2j-1}$, $H_{2j} = \overline{\mathcal{H}}_m\mathcal{E}_{2j}$, $(j = 1, 2, \ldots, m)$ with $\mathcal{E}_i = [0_{r\times(i-1)r}, I_r, 0_{r\times(2m-i)r}]^\top$, $(i = 1, 2, \ldots, m)$. In addition, let the matrix Λ in Line 1 and $H_{j+1,j}$ in Line 11 of Algorithm 1 are partitioned as follows

$$\Lambda = \begin{bmatrix} \Lambda_{1,1} & \Lambda_{1,2} \\ 0 & \Lambda_{2,2} \end{bmatrix}, H_{j+1,j} = \begin{bmatrix} H_{j+1,j}^{(1,1)} & H_{j+1,j}^{(1,2)} \\ 0 & H_{j+1,j}^{(2,2)} \end{bmatrix},$$

in which $\Lambda_{i,k}, H_{j+1,j}^{(i,k)} \in \mathbb{R}^{r\times r}$.

Proposition 2.1. *Suppose that $\overline{\mathcal{T}}_m$ and $\overline{\mathcal{H}}_m$ are the block upper Hessenberg matrices as defined above. Then the odd and even block columns of $\overline{\mathcal{T}}_m$ satisfy recursive relations:*

1. $T_{2j-1} = H_{2j-1}$, *for* $j = 1, 2, \ldots, m$.
2. $T_2 = \Big(\mathcal{E}_1\Lambda_{11} - H_1\Lambda_{1,2}\Big)\Lambda_{2,2}^{-1}$.
3. $T_{2j+2} = \Big(\mathcal{E}_{2j} - T_{:,1:2j+1}H_{1:2j+1,2j}\Big)(H_{j+1,j}^{(2,2)})^{-1}$, *for* $j = 1, 2, \ldots, m-1$.

where $\mathcal{E}_i \in \mathbb{R}^{2mr\times r}$ is the i-th block column of the identity matrix I_{2mr} with $r \times r$ blocks.

Proof. To prove the first part (1), we consider the relation

$$[AV_j^{(1)}, A^{-1}V_j^{(2)}] = \sum_{i=1}^{j+1} V_i H_{i,j}, \tag{5}$$

where is derived from Lines 4 - 11 of Algorithm 1. If we block matrices $V_i \in \mathbb{R}^{n\times 2r}$ and $H_{i,j} \in \mathbb{R}^{2r\times 2r}$ as follows:

$$V_i = [V_i^{(1)}, V_i^{(2)}], \quad H_{i,j} = \begin{pmatrix} H_{i,j}^{(1,1)} & H_{i,j}^{(1,2)} \\ H_{i,j}^{(2,1)} & H_{i,j}^{(2,2)} \end{pmatrix},$$

for $i = 1, 2, \ldots, j+1$ with $H_{j+1,j}^{(2,1)} = 0$, then the relation (5) transforms to

$$\begin{cases} AV_j^{(1)} & = \sum_{i=1}^{j} V_i \begin{pmatrix} H_{i,j}^{(1,1)} \\ H_{i,j}^{(2,1)} \end{pmatrix} + V_{j+1}^{(1)} H_{j+1,j}^{(1,1)}, \\ A^{-1}V_j^{(2)} & = \sum_{i=1}^{j+1} V_i \begin{pmatrix} H_{i,j}^{(1,2)} \\ H_{i,j}^{(2,2)} \end{pmatrix}. \end{cases} \tag{6}$$

From the first equation (6), we have

$$AV_j^{(1)} = \mathcal{V}_{m+1} H_{2j-1}. \tag{7}$$

On the other side, by considering the block Hessenberg relation (2), $AV_j^{(1)}$ can be written as

$$AV_j^{(1)} = \mathcal{V}_{m+1} T_{2j-1}. \tag{8}$$

Putting relations (7) and (8) together gives the first part.

According to line 1 of Algorithm 1, and the partitioning of the matrix Λ,

$$V = V_1^{(1)} \Lambda_{1,1}, \tag{9}$$

$$A^{-1}V = V_1^{(1)} \Lambda_{1,2} + V_1^{(2)} \Lambda_{2,2}. \tag{10}$$

Then premultiplying (10) by A and using (9), it obtains

$$AV_1^{(2)} \Lambda_{2,2} = V_1^{(1)} \Lambda_{1,1} - AV_1^{(1)} \Lambda_{1,2}. \tag{11}$$

Since $V_1^{(1)} = \mathcal{V}_{m+1}\mathcal{E}_1$, $\mathcal{V}_{m+1}^L \mathcal{V}_{m+1} = I_{2(m+1)r}$, and $\Lambda_{2,2}$ is invertible, then the second block column of the matrix $\overline{\mathcal{T}}_m$ can be expressed as

$$T_2 = \Big(\mathcal{E}_1 \Lambda_{11} - H_1 \Lambda_{1,2}\Big) \Lambda_{2,2}^{-1}.$$

where $\mathcal{E}_1$ is the $2mr \times r$ corresponding to the first r columns of the identity matrix I_{2mr}.

In the sequel, the main goal is to find a recursive relation for the even block columns of $\overline{\mathcal{T}}_m$ in terms of the block columns of $\overline{\mathcal{H}}_m$. To this end, we consider the second equation of (6) and multiply it from the left by A it can be seen that

$$\begin{aligned} AV_{j+1}^{(2)} H_{j+1,j}^{(2,2)} &= V_j^{(2)} - A\mathcal{V}_j H_{1:2j,2j} - AV_{j+1}^{(1)} H_{j+1,j}^{(1,2)} \\ &= \mathcal{V}_{m+1}\mathcal{E}_{2j} - \mathcal{V}_{m+1} T_{:,1:2j} H_{1:2j,2j} - \mathcal{V}_{m+1} T_{:,2j+1} H_{j+1,j}^{(1,2)} \\ &= \mathcal{V}_{m+1}\Big(\mathcal{E}_{2j} - T_{:,1:2j+1} H_{1:2j+1,2j}\Big). \end{aligned}$$

It should be noted that $\mathcal{V}_{m+1}^{L} A V_{j+1}^{(2)} = T_{2j+2}$ and $H_{j+1,j}^{(2,2)}$ is a full rank matrix. Therefore, this completes the proof of the third part. □

3. An approximation of the matrix exponential

This section's primary goal is briefly reviewing two approaches for computing approximate solutions to large-scale Sylvester differential matrix equations (1). Let us first recall the following proposition which gives the exact solution of (1).

Proposition 3.1. *([3]) Let $A(t), B(t)$ and $C(t)$ be the continuous matrix functions on $\mathbb{R}$. Then the Sylvester differential matrix equation*

$$\dot{X}(t) = A(t)X(t) + X(t)B(t) + C(t),\ X(t_0) = X_0, \tag{12}$$

has a unique solution as

$$X(t) = \Phi_A(t, t_0)X_0\Phi_{B^\top}^{\top}(t, t_0) + \int_{t_0}^{t} \Phi_A(t, \tau)C(\tau)\Phi_{B^\top}^{\top}(t, \tau)d\tau,\ t, t_0 \in \mathbb{R},$$

where the transition (or evolution) matrix $\Phi_A(t, t_0)$ is the unique solution of the following problem

$$\frac{\partial}{\partial t}\Phi_A(t, t_0) = A(t)\Phi_A(t, t_0),\ \Phi_A(t_0, t_0) = I_n.$$

In the special case when the matrix functions $A(t)$ and $B(t)$ are constant, $\Phi_A(t, t_0) = e^{(t-t_0)A}$ for $t, t_0 \in \mathbb{R}$, see Theorem 1.1.1 in [3]. In this case, the exact solution of the Sylvester differential matrix equation (12) can be expressed as

$$X(t) = e^{(t-t_0)A}X_0e^{(t-t_0)B} + \int_{t_0}^{t} e^{(t-\tau)A}C(\tau)e^{(t-\tau)B}d\tau. \tag{13}$$

From now on, return to the case where the matrix functions $A(t), B(t)$, and $C(t)$ are assumed to be constant functions on $\mathbb{R}$. According to what has been said and (13), the exact solution of the Sylvester differential matrix equation (1) is as follows

$$X(t) = e^{(t-t_0)A}X_0e^{(t-t_0)B} + \int_{t_0}^{t} e^{(t-\tau)A}EF^{\top}e^{(t-\tau)B}d\tau. \tag{14}$$

Note that choosing $X_0 = 0$ in the initial condition $X(t_0) = X_0$ does not cause any restriction. Let $X(t)$ be the exact solution of the Sylvester differential matrix equation (12), then the matrix function $Z(t) = X(t) - X_0$ is the unique solution of the following initial value problem

$$\dot{Z}(t) = AZ(t) + Z(t)B + C_0,\ Z(t_0) = 0, \tag{15}$$

where $C_0 = AX_0 + X_0B + C$. To obtain $X(t)$, we first solve the initial value problem (15) and acquire $Z(t)$ and then conclude $X(t) = Z(t) + X_0$.

Assuming that $X_0 = 0$ and $C = EF^\top$, the exact solution (13) can be written as follows

$$X(t) = \int_{t_0}^{t} e^{(t-\tau)A} EF^\top e^{(t-\tau)B} d\tau = \int_{0}^{t-t_0} e^{sA} EF^\top e^{sB} ds. \tag{16}$$

To integrate formula (16), we follow two approaches: In the first approach, we approximate factors $e^{(t-\tau)A}E$ and $e^{(t-\tau)B^\top}F$ and then compute the desired approximation through a quadrature method. In the second approach, we project the initial value problem (1) onto an extended block Krylov subspace and then obtain the low-rank approximate solutions.

In the first, consider the expression of the exact solution

$$X(t) = \int_{t_0}^{t} e^{(t-\tau)A} EF^\top e^{(t-\tau)B} d\tau. \tag{17}$$

To approximate $e^{(t-\tau)A}E$ and $e^{(t-\tau)B^\top}F$, we use an extended block Krylov subspace method. By Algorithm 1, we compute the left invertible matrices $\mathcal{V}_m$ and $\mathcal{W}_m$ whose columns form a basis of the subspaces $\mathbb{K}^e_m(A, E)$ and $\mathbb{K}^e_m(B^\top, F)$, respectively. Following the idea in [26, 27], an approximation to $Z^A(\tau) = e^{(t-\tau)A}E$ and $Z^B(\tau) = e^{(t-\tau)B^\top}F$ can get as

$$Z^A_m(\tau) = \mathcal{V}_m e^{(t-\tau)\mathcal{T}^A_m} \mathcal{V}^L_m E, \quad Z^B_m(\tau) = \mathcal{W}_m e^{(t-\tau)\mathcal{T}^B_m} \mathcal{W}^L_m F,$$

where $\mathcal{T}^A_m = \mathcal{V}^L_m A \mathcal{V}_m$ and $\mathcal{T}^B_m = \mathcal{W}^L_m B^\top \mathcal{W}_m$. Thus, the term in front of the integral in the exact solution (17) can be approximated as follows

$$\begin{aligned} Z^A(\tau)(Z^B(\tau))^\top &= e^{(t-\tau)A} EF^\top e^{(t-\tau)B} \approx Z^A_m(\tau)(Z^B_m(\tau))^\top \\ &= \mathcal{V}_m \big(e^{(t-\tau)\mathcal{T}^A_m} E_m\big)\big(e^{(t-\tau)\mathcal{T}^B_m} F_m\big)\mathcal{W}_m^\top, \end{aligned} \tag{18}$$

in which $E_m = \mathcal{V}^L_m E$ and $F_m = \mathcal{W}^L_m F$. It should be noted that the matrices E_m and F_m can be written in a simpler form. For this reason, consider the LU decomposition with partial pivoting (PLU decomposition) of $[E, A^{-1}E]$ and $[F, (B^\top)^{-1}F]$, i.e.,

$$[E, A^{-1}E] = V_1 \Lambda^A, \quad [F, (B^\top)^{-1}F] = W_1 \Lambda^B,$$

where $\Lambda^A = \begin{pmatrix} \Lambda^A_{1,1} & \Lambda^A_{1,2} \\ 0_r & \Lambda^A_{2,2} \end{pmatrix}$ and $\Lambda^B = \begin{pmatrix} \Lambda^B_{1,1} & \Lambda^B_{1,2} \\ 0_r & \Lambda^B_{2,2} \end{pmatrix}$ with $\Lambda^A_{i,j}, \Lambda^B_{i,j} \in \mathbb{R}^{r\times r}$. These PLU decompositions imply

$$E_m = \mathcal{V}^L_m E = \mathcal{E}_1 \Lambda^A_{1,1}, \quad F_m = \mathcal{W}^L_m F = \mathcal{E}_1 \Lambda^B_{1,1}, \tag{19}$$

where $\mathcal{E}_1$ is the $2mr \times r$ matrix corresponding to the first r columns of the identity matrix I_{2mr}.

Setting $\widetilde{Z}^A_m(\tau) = e^{(t-\tau)\mathcal{T}^A_m}\mathcal{E}_1\Lambda^A_{1,1}, \widetilde{Z}^B_m(\tau) = e^{(t-\tau)\mathcal{T}^B_m}\mathcal{E}_1\Lambda^B_{1,1}$ and using (18), the exact solution (17) can be approximated as

$$X(t) \approx X_m(t) = \mathcal{V}_m Y_m(t) \mathcal{W}_m, \tag{20}$$

where $Y_m(t) = \int_{t_0}^{t} \widetilde{Z}^A_m(\tau)\big(\widetilde{Z}^B_m(\tau)\big)^\top d\tau$.

Since m is generally very small ($m \ll n$), the factors $\widetilde{Z}_m^A(\tau)$ and $\widetilde{Z}_m^B(\tau)$ can be calculated via the `expm` function of Matlab, and the approximation of $Y_m(t)$ can be computed by Gauss quadrature formula.

Next, to present the second approach, we follow a process similar to the one described in [26] in brief. Again, consider the integral (16).

It is worth noting that it should be integrated from a function of the form $Z^A(s)(Z^B(s))^\top$ where $Z^A(s) = e^{sA}E$ and $Z^B(s) = e^{sB^\top}F$. These matrix functions in the interval $[0, t-t_0]$ must be approximated as $Z^A(s) \approx Z_m^A(s) = p_m(sA)E$ and $Z^B(s) \approx Z_m^B(s) = q_m(sB^\top)F$, in which p_m and q_m are polynomials of degree $m-1$. Therefore, the approximation of $X(t)$ is a form of

$$X_m(t) = \int_0^{t-t_0} Z_m^A(s)(Z_m^B(s))^\top ds. \tag{21}$$

To make the approximation $X_m(t)$ of $X(t)$ as precise as desired, we should increase the degree of polynomials p_m and q_m enough.

Let $\mathbb{V}_m = \begin{bmatrix} E & AE & \cdots & A^{m-1}E \end{bmatrix}$, $\mathbb{W}_m = \begin{bmatrix} F & B^\top F & \cdots & (B^\top)^{m-1}F \end{bmatrix}$, $p_m(s) = \alpha_0 + \alpha_1 s + \cdots + \alpha_{m-1} s^{m-1}$, and $q_m(s) = \beta_0 + \beta_1 s + \cdots + \beta_{m-1} s^{m-1}$. Then the matrix function $Z_m^B(s)(Z_m^A(s))^\top$ can be presented as

$$\begin{aligned} Z_m^A(s)(Z_m^B(s))^\top &= \mathbb{V}_m\big(z_m(s) \otimes I_r\big)\big(\tilde{z}_m(s)^\top \otimes I_r\big)\mathbb{W}_m^\top \\ &= \mathbb{V}_m\big(z_m(s)\tilde{z}_m(s)^\top \otimes I_r\big)\mathbb{W}_m^\top, \end{aligned}$$

in which $z_m(s)^\top = (\alpha_0, \alpha_1 s, \ldots, \alpha_{m-1} s^{m-1})$ and $\tilde{z}_m(s)^\top = (\beta_0, \beta_1 s, \ldots, \beta_{m-1} s^{m-1})$. Substituting this in (21) yields,

$$X_m(t) = \mathbb{V}_m\Big(\int_0^{t-t_0} \big(z_m(s)\tilde{z}_m(s)^\top \otimes I_r\big)ds\Big)\mathbb{W}_m^\top \equiv \mathbb{V}_m Y_m(t)\mathbb{W}_m^\top,$$

where $Y_m(t) = \int_0^{t-t_0} \big(z_m(s)\tilde{z}_m(s)^\top \otimes I_r\big)ds$. As a result, the approximate solutions of the Sylvester differential matrix equation (1) are of the form

$$X_m(t) = \mathbb{V}_m Y_m(t)\mathbb{W}_m^\top,$$

where $Y_m(t)$ is a matrix function of size $mr \times mr$.

It is worthy of note that the approximations $p_m(sA)E$ and $q_m(sB^\top)F$ of $e^{sA}E$ and $e^{sB^\top}F$ are elements of the matrix Krylov subspace $K_m(A, E)$ and $K_m(B^\top, F)$, respectively, where

$$\begin{aligned} K_m(A, E) &= \mathtt{Span}\left\{E, AE, \ldots, A^{m-1}E\right\} \\ &= \left\{\sum_{i=1}^{m} \alpha_{i-1}A^{i-1}E | \alpha_i \in \mathbb{R}, i = 0, 1, \ldots, m-1\right\}. \end{aligned}$$

In this approach, to generate the approximate solution $X_m(t)$, we should build the block Krylov matrices $\mathbb{V}_m$ and $\mathbb{W}_m$. These would entail some additional matrix-matrix products. To avoid computing the matrix-matrix products, we should construct a basis of $K_m(A, E)$ and $K_m(B^\top, F)$ via the well-known global

Arnoldi or the global Hessenberg algorithm starting with E and F, respectively. But we consider the extended block Krylov subspace instead of the matrix Krylov subspace because it contains more information than the block and matrix Krylov subspace. In this case, the approximate solution $X_m(t)$ can be expressed as

$$X_m(t) = \mathcal{V}_m Y_m(t) \mathcal{W}_m^\top, \tag{22}$$

in which $Y_m(t) \in \mathbb{R}^{2mr \times 2mr}$ and the matrices $\mathcal{V}_m \in \mathbb{R}^{n \times 2mr}$, $\mathcal{W}_m \in \mathbb{R}^{s \times 2mr}$ are generated by Algorithm 1. These matrices are left invertible.

From (20) and (22), it can be seen that the approximate solutions of the Sylvester differential matrix equation (1) in both viewpoints are the same, while the matrix $Y_m(t)$ is calculated differently.

Since the matrices $\mathcal{V}_m$ and $\mathcal{W}_m$ are left invertible, it can be shown that the matrix function $Y_m(t)$ is the solution of a low-dimensional Sylvester differential matrix equation. This is discussed in the following Proposition.

Proposition 3.2. *Let the matrices $\mathcal{V}_m$ and $\mathcal{W}_m$ be left invertible with the left inverse $\mathcal{V}_m^L$ and $\mathcal{W}_m^L$, respectively. In addition, let $X_m(t)$ be the approximate solution of the initial value problem* (1)*. Then the matrix function $Y_m(t)$ is the approximate solution of the reduced Sylvester differential equation*

$$\dot{Y}(t) = \mathcal{T}_m^A Y(t) + Y(t)(\mathcal{T}_m^B)^\top + E_m F_m^\top, \tag{23}$$

with the initial condition $Y(t_0) = 0$, $\mathcal{T}_m^A = \mathcal{V}_m^L A \mathcal{V}_m$, $\mathcal{T}_m^B = \mathcal{W}_m^L B \mathcal{W}_m$, $E_m = \mathcal{V}_m^L E$ and $F_m = \mathcal{W}_m^L F$.

Proof. By premultipling and postmultipling (1) by $\mathcal{V}_m^L$ and $(\mathcal{W}_m^L)^\top$, respectively, and using the fact that $X_m(t)$ is the approximate solution of it, we have

$$\mathcal{V}_m^L \dot{X}_m(t)(\mathcal{W}_m^L)^\top \approx \mathcal{V}_m^L A X_m(t)(\mathcal{W}_m^L)^\top + \mathcal{V}_m^L X_m(t) B (\mathcal{W}_m^L)^\top + \mathcal{V}_m^L E F^\top (\mathcal{W}_m^L)^\top.$$

Since $X_m(t) = \mathcal{V}_m Y_m(t) \mathcal{W}_m^\top$, $\mathcal{V}_m^L \mathcal{V}_m = I_{2mr}$ and $\mathcal{W}_m^\top (\mathcal{W}_m^L)^\top = I_{2mr}$, then the desired result can be acquired. □

An immediate corollary of the above result is that the residual matrix function associated with the approximate solution $X_m(t)$ satisfies the following condition

$$\begin{aligned}
\mathcal{V}_m^L R_m(t)(\mathcal{W}_m^L)^\top &= \mathcal{V}_m^L \Big(A X_m(t) + X_m(t) B + E F^\top - \dot{X}_m(t) \Big) (\mathcal{W}_m^L)^\top \\
&= \mathcal{V}_m^L A \mathcal{V}_m Y_m(t) + Y_m(t) (\mathcal{W}_m^L B^\top \mathcal{W}_m)^\top \\
&+ \mathcal{V}_m^L E F^\top (\mathcal{W}_m^L)^\top - \mathcal{V}_m^\top \dot{X}_m(t) (\mathcal{W}_m^L)^\top \\
&= 0.
\end{aligned}$$

In the following Proposition, we gain an expression of the residual norm of $R_m(t)$ to avoid computing matrix products with the large matrices A and B.

This result shows how to calculate the norm of $R_m(t)$ without using the approximation $X_m(t)$ that is computed in a factored form only when convergence is gained.

Proposition 3.3. *Let $Y_m(t)$ be the exact solution of (23) and let $X_m(t) = \mathcal{V}_m Y_m(t) \mathcal{W}_m^\top$ be the approximate solution of the Sylvester differential equation (1) achieved after m iterations of the extended block Hessenberg Algorithm. Then, the residual semi-norm $R_m(t)$ corresponding to $X_m(t)$ satisfy*

$$r_m(t) = |R_m(t)|_{(\mathcal{V}_{m+1}, \mathcal{W}_{m+1})} = \sqrt{\alpha_m(t) + \beta_m(t)}, \tag{24}$$

where $\alpha_m(t) = \|T^A_{m+1,m} \mathbb{E}_m^\top Y_m(t)\|_F^2$ and $\beta_m(t) = \|Y_m(t)(T^B_{m+1,m} \mathbb{E}_m^\top)^\top\|_F^2$.

Proof. We start with $R_m(t) = \dot{X}_m(t) - AX_m(t) - X_m(t)B + EF^\top$ that is the residual matrix function corresponding to the approximate solution $X_m(t)$. Since $A\mathcal{V}_m = \mathcal{V}_m \mathcal{T}_m^A + V_{m+1} T^A_{m+1,m} \mathbb{E}_m^\top$ and $B^\top \mathcal{W}_m = \mathcal{W}_m \mathcal{T}_m^B + W_{m+1} T^B_{m+1,m} \mathbb{E}_m^\top$, then $R_m(t)$ can be reformulated as

$$\begin{aligned} R_m(t) &= \dot{X}_m(t) - AX_m(t) - X_m(t)B + EF^\top \\ &= \mathcal{V}_m \dot{Y}_m(t) \mathcal{W}_m^\top - A\mathcal{V}_m Y_m(t) \mathcal{W}_m^\top - \mathcal{V}_m Y_m(t) \mathcal{W}_m^\top B + EF^\top \\ &= \mathcal{V}_m \Big(\dot{Y}_m(t) - \mathcal{T}_m^A Y_m(t) - Y_m(t)(\mathcal{T}_m^B)^\top + (\mathcal{E}_1 \Lambda^A_{1,1})(\mathcal{E}_1 \Lambda^B_{1,1})^\top \Big) \mathcal{W}_m^\top \\ &\quad - V_{m+1} T^A_{m+1,m} \mathbb{E}_m^\top Y_m(t) \mathcal{W}_m^\top - \mathcal{V}_m Y_m(t) (W_{m+1} T^B_{m+1,m} \mathbb{E}_m^\top)^\top. \end{aligned}$$

According to $Y_m(t)$ that is the exact solution of (23), we have

$$R_m(t) = \mathcal{V}_{m+1} \begin{pmatrix} 0_{2mr \times 2mr} & -Y_m(t)(T^B_{m+1,m} \mathbb{E}_m^\top)^\top \\ -T^A_{m+1,m} \mathbb{E}_m^\top Y_m(t) & 0_{2r \times 2r} \end{pmatrix} \mathcal{W}_{m+1}^\top, \tag{25}$$

where $\mathcal{V}_{m+1} = [\mathcal{V}_m, V_{m+1}]$ and $\mathcal{W}_{m+1} = [\mathcal{W}_m, W_{m+1}]$. By Definition 1.4, the semi-norm of the residual matrix function $R_m(t)$ is written as follows

$$\begin{aligned} |R_m(t)|^2_{(\mathcal{V}_{m+1}, \mathcal{W}_{m+1})} &= \|\mathcal{V}^L_{m+1} R_m(t) (\mathcal{W}^L_{m+1})^\top\|_F^2 \\ &= \|T^A_{m+1,m} \mathbb{E}_m^\top Y_m(t)\|_F^2 + \|Y_m(t)(T^B_{m+1,m} \mathbb{E}_m^\top)^\top\|_F^2. \end{aligned}$$

□

The following proposition establishes an upper bound for the residual matrix function Frobenius norm of the approximate solution $X_m(t)$.

Proposition 3.4. *Let $X_m(t)$ be the approximate solution obtained from the m-th iteration of the extended block Hessenberg algorithm, and let $R_m(t)$ be the residual matrix function associated with $X_m(t)$. Then the residual norm satisfies the inequality*

$$\|R_m(t)\|_F \leq \sqrt{2(m+1)r \, \mathtt{max}(n,s)} \sqrt{\alpha_m(t) + \beta_m(t)} = \tilde{r}_m(t), \tag{26}$$

where $\alpha_m(t) = \|T^A_{m+1,m}(Y_m(t))_{m_r,:}\|_F^2$, $\beta_m(t) = \|(Y_m(t))_{:,m_r}(T^B_{m+1,m})^\top\|_F^2$, with $m_r = 2(m-1)r + 1 : 2mr$.

Proof. It was seen in Proposition 3.3 that

$$R_m(t) = \mathcal{V}_{m+1} \begin{pmatrix} 0_{2mr\times 2mr} & -Y_m(t)(T^B_{m+1,m}\mathbb{E}_m^\top)^\top \\ -T^A_{m+1,m}\mathbb{E}_m^\top Y_m(t) & 0_{2r\times 2r} \end{pmatrix} \mathcal{W}_{m+1}^\top. \tag{27}$$

Taking the norm of both sides of (27), and using $\|AB\|_F \le \|A\|_F\|B\|_F$, gives the following inequality

$$\|R_m(t)\|_F \le \|\mathcal{V}_{m+1}\|_F \|G_m\|_F \|\mathcal{W}_{m+1}^\top\|_F,$$

in which $G_m(t) = \begin{pmatrix} 0_{2mr\times 2mr} & -Y_m(t)(T^B_{m+1,m}\mathbb{E}_m^\top)^\top \\ -T^A_{m+1,m}\mathbb{E}_m^\top Y_m(t) & 0_{2r\times 2r} \end{pmatrix}$.

Now, recall that the matrices $\mathcal{V}_{m+1} \in \mathbb{R}^{n\times 2(m+1)r}$ and $\mathcal{W}_{m+1} \in \mathbb{R}^{s\times 2(m+1)r}$ are produced by permutation matrices. These matrices are lower triangular trapezoidal and their elements are less than or equal to one. Therefore, the Frobenius norm of the matrices $\mathcal{V}_{m+1}$ and $\mathcal{W}_{m+1}$ are bounded as follows

$$\|\mathcal{V}_{m+1}\|_F \le \sqrt{2(m+1)nr}\,\|\mathcal{V}_{m+1}\|_{\mathtt{max}} \le \sqrt{2(m+1)nr},$$
$$\|\mathcal{W}_{m+1}\|_F \le \sqrt{2(m+1)sr}\,\|\mathcal{W}_{m+1}\|_{\mathtt{max}} \le \sqrt{2(m+1)sr}.$$

From these inequalities, it follows that

$$\|R_m(t)\|_F \le \sqrt{2(m+1)r\,\mathtt{max}(n,s)}\sqrt{\alpha_m+\beta_m},$$

where

$$\alpha_m = \|T^A_{m+1,m}\mathbb{E}_m^\top Y_m(t)\|_F^2 = \|T^A_{m+1,m}(Y_m(t))_{m_r,:}\|_F^2,$$
$$\beta_m = \|Y_m(t)(T^B_{m+1,m}\mathbb{E}_m^\top)^\top\|_F^2 = \|(Y_m(t))_{:,m_r}(T^B_{m+1,m})^\top\|_F^2,$$

and $m_r = 2(m-1)r+1:2mr$. □

We sum up the above method for solving the large-scale Sylvester differential matrix equations in Algorithm 2.

4. **Low-rank approximate solutions to the large-scale Sylvester differential equations.**

In this section, we show how to obtain low-rank approximate solutions to the Sylvester differential equation (1) by projecting directly the initial problem onto the extended block Krylov subspace. We first apply the extended block Hessenberg process with maximum strategy Algorithm 1 to the pairs (A,E) and $(B^\top,F)$, respectively, to get the matrices $\mathcal{V}_m$, $\mathcal{W}_m$, $\mathcal{T}_m^A = \mathcal{V}_m^L A\mathcal{V}_m$, and $\mathcal{T}_m^B = \mathcal{W}_m^L B^\top \mathcal{W}_m$. Let $X_m(t)$ be low-rank approximate solutions of (1) that have the form

$$X_m(t) = \mathcal{V}_m Y_m(t)\mathcal{W}_m^\top, \tag{28}$$

where $\mathcal{V}_m = [V_1, V_2, \cdots, V_m]$, $\mathcal{W}_m = [W_1, W_2, \cdots, W_m]$ with $V_i \in \mathbb{R}^{n\times 2r}$, $W_i \in \mathbb{R}^{s\times 2r}$ and $Y_m(t)$ is a matrix function of size $2mr \times 2mr$. Moreover, suppose

Input: The matrices $A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{s\times s}$, $E \in \mathbb{R}^{n\times r}$, $F \in \mathbb{R}^{s\times r}$, the initial and final times t_0, T, a tolerance $tol > 0$, m_{max} a maximum number of iterations, k a step-size parameter and N the number of mesh points in the time partioning, τ the tolerance for the truncated SVD.

Output: The approximate solution $X_m(t_N)$.

Compute $h = \frac{T-t_0}{N}$.
for $m = 1, \ldots, m_{max}$ **do**
  Use the extended block Hessenberg Algorithm 1 to the pairs (A, E) and $(B^\top, F)$ to compute the bases $\mathcal{V}_m = [V_1, \cdots, V_m]$ and $\mathcal{W}_m = [W_1, \cdots, W_m]$ and also the the block upper-Hessenberg matrices $\mathcal{T}_m^A$ and $\mathcal{T}_m^B$.
  if $\mathtt{rem}(m, k) = 0$ **then**
    for $i = 1, \ldots, N$ **do**
      Compute $t_i = t_0 + i\,h$;
      Compute the factors $\widetilde{Z}_m^A(\tau) = e^{(t_i-\tau)\mathcal{T}_m^A} E_m$ and $\widetilde{Z}_m^B(\tau) = e^{(t_i-\tau)\mathcal{T}_m^B} F_m$ with $E_m = \mathcal{V}_m^L E$ and $F_m = \mathcal{W}_m^L F$ using the Matlab function `expm`;
      To obtain $Y_m(t_i)$, apply a quadrature formula to compute the integral $\int_{t_0}^{t_i} \widetilde{Z}_m^A(\tau)\big(\widetilde{Z}_m^B(\tau)\big)^\top d\tau$;
      Compute the SVD of $Y_m(t_i)$: $Y_m(t_i) = \widehat{U}\widehat{S}\widehat{V}^\top$, where $\widehat{S} = \mathtt{diag}(\sigma_1, \ldots, \sigma_{2mr})$, and $\sigma_1 \geq \ldots \geq \sigma_{2mr}$; Find the scalar ℓ such that $\sigma_\ell > \tau$ and let $\widehat{S}_\ell = \mathtt{diag}(\sigma_1, \ldots, \sigma_\ell)$; Form $\mathbb{Z}_m^A = \mathcal{V}_m \widehat{U}_\ell \widehat{S}_\ell^{1/2}$ and $\mathbb{Z}_m^B = \mathcal{W}_m \widehat{V}_\ell \widehat{S}_\ell^{1/2}$, then compute the approximate solution $X_m(t_i) \approx \mathbb{Z}_m^A (\mathbb{Z}_m^B)^\top$;
    end for
    if $\|R_m(t)\|_F < tol$ **then**
      Stop;
    end if
  end if
end for=0

Algorithm 2: The extended block Hessenberg (EBHess-exp) method.

that the residual $R_m(t) = AX_m(t) + X_m(t)B - EF^\top - \dot{X}_m(t)$ associated with the approximate solution $X_m(t)$ satisfies the Petrov-Galerkin condition

$$\mathcal{V}_m^L R_m(t)(\mathcal{W}_m^L)^\top = 0, \tag{29}$$

in which $\mathcal{V}_m^L$ and $\mathcal{W}_m^L$ are the left inverses of the matrices $\mathcal{V}_m$ and $\mathcal{W}_m$, respectively. Then regarding (28) and the Petrov-Galerkin condition (29) and using the fact that $\mathcal{V}_m^L \mathcal{V}_m = I_{2mr}$ and $\mathcal{W}_m^L \mathcal{W}_m = I_{2mr}$, we obtain the low-dimensional

Sylvester differential equation

$$\begin{aligned}\dot{Y}_m(t) &= \mathcal{V}_m^L A\mathcal{V}_m Y_m(t) + Y_m(t)(\mathcal{W}_m^L B^\top \mathcal{W}_m)^\top + E_m F_m^\top \\ &= \mathcal{T}_m^A Y_m(t) + Y_m(t)(\mathcal{T}_m^B)^\top + E_m F_m^\top. \end{aligned} \tag{30}$$

Using (19), the low-dimensional Sylvester differential equation can be expressed as

$$\dot{Y}_m(t) = \mathcal{T}_m^A Y_m(t) + Y_m(t)(\mathcal{T}_m^B)^\top + (\mathcal{E}_1\Lambda_{1,1}^A)(\mathcal{E}_1\Lambda_{1,1}^B)^\top. \tag{31}$$

The approximate solution of the aforementioned low-dimensional Sylvester differential equation can be computed by some integration method for instance the Rosenbrock method [16, 18, 25] or the backward differentiation formula (BDF) [16,18]. When these two methods are used, we will face a Sylvester matrix equation in each iteration, if the coefficients of the Sylvester matrix equation are large-scale, we can use the Krylov methods mentioned in [2,5,9,21,22], and for moderate size, the Bartels-Stewart [13] and the Hesenberg-Schur [17] methods are suitable. In this work, we have computed the approximate solutions of the low-dimensional Sylvester differential equation (31) using BDF which is stated in [18].

Similar to what is said in Proposition 3.4, the following proposition gives a result that allows us the computation of an upper bound of the Frobenius norm of the residual matrix function.

Proposition 4.1. *The upper bound of the Frobenius norm of the residual $R_m(t)$ associated with the approximation solution $X_m(t)$ obtained after m iterations of the LRSD-EBHess Algorithm satisfies the following relation*

$$\|R_m(t)\|_F \leq \sqrt{2(m+1)r\,\mathtt{max}(n,s)}\sqrt{\alpha_m(t)+\beta_m(t)} = \tilde{r}_m(t), \tag{32}$$

where $\alpha_m(t) = \|T_{m+1,m}^A(Y_m(t))_{m_r,:}\|_F^2$, $\beta_m(t) = \|(Y_m(t))_{:,m_r}(T_{m+1,m}^B)^\top\|_F^2$, with $m_r = 2(m-1)r+1:2mr$.

In addition, we consider the error matrix function $E_m(t)$ given by

$$E_m(t) = X^*(t) - X_m(t), \quad t \in [t_0, T],$$

where $X^*(t)$ and $X_m(t)$ are the exact solution and approximate solution of the Sylvester differential matrix equation (1), respectively.

The following proposition gives an upper bound for the error matrix function norm $E_m(t)$.

Proposition 4.2. *Assume that $X_m(t)$ is the approximate solution of the Sylvester differential matrix equation* (1) *obtained after m iterations of the LR-SDE-EHess algorithm. Then the error matrix function $E_m(t)$ associated with $X_m(t)$ satisfies*

$$\|E_m\|_\infty \leq \sqrt{2(m+1)r\,\mathtt{max}(n,s)}\left(\frac{1-e^{(T-t_0)\left(\|A\|_F+\|B\|_F\right)}}{\|A\|_F+\|B\|_F}\right)\sqrt{\widehat{\alpha}_m+\widehat{\beta}_m},$$

where $\widehat{\alpha}_m = \|T^A_{m+1,m}\|_F^2 \|(Y_m)_{m_r,:}\|_\infty^2$, $\widehat{\beta}_m = \|(Y_m)_{:,m_r}\|_\infty^2 \|(T^B_{m+1,m})^\top\|_F^2$.

Proof. From (1), we have

$$\begin{aligned}\dot{E}_m(t) &= \dot{X}^*(t) - \dot{X}_m(t)\\ &= A(X^*(t) - X_m(t)) + (X^*(t) - X_m(t))B + R_m(t),\end{aligned}$$

where $R_m(t) = AX_m(t) + X_m(t)B + EF^\top - \dot{X}_m(t)$ and $E_m(t_0) = 0$. Therefore, the matrix error function $E_m(t)$ is the exact solution of the following initial value problem

$$\begin{cases}\dot{E}_m(t) = AE_m(t) + E_m(t)B + R_m(t), \quad t \in [t_0, T],\\ E_m(t_0) = 0.\end{cases}$$

In view of relation (13), $E_m(t)$ can be written as the following integral formula

$$E_m(t) = \int_{t_0}^{t} e^{(t-\tau)A} R_m(\tau) e^{(t-\tau)B} d\tau.$$

Passing to the norm and using the fact that $\|MN\|_F \leq \|M\|_F \|N\|_F$ and $\|e^{\beta M}\|_F \leq e^{\beta \|M\|_F}$ for $\beta \geq 0$, we get

$$\begin{aligned}\|E_m(t)\|_F &\leq \int_{t_0}^{t} \|e^{(t-\tau)A}\|_F \|R_m(\tau)\|_F \|e^{(t-\tau)B}\|_F d\tau\\ &\leq \int_{t_0}^{t} \|R_m(\tau)\|_F \, e^{(t-\tau)\left(\|A\|_F + \|B\|_F\right)} \, d\tau\\ &\leq \max_{\tau\in[t_0,t]} \|R_m(\tau)\|_F \int_{t_0}^{t} e^{(t-\tau)\left(\|A\|_F + \|B\|_F\right)} \, d\tau\\ &\leq \left(\frac{1 - e^{(t-t_0)\left(\|A\|_F + \|B\|_F\right)}}{\|A\|_F + \|B\|_F}\right) \max_{\tau\in[t_0,t]} \|R_m(\tau)\|_F, \qquad (33)\end{aligned}$$

for all $t \in [t_0, T]$.

According to Proposition 3.4,

$$\begin{aligned}\|R_m\|_\infty &:= \max_{\tau\in[t_0,T]} \|R_m(\tau)\|_F\\ &\leq \sqrt{2(m+1)r\,\mathtt{max}(n,s)}\sqrt{\widehat{\alpha}_m + \widehat{\beta}_m}\end{aligned}$$

where $\widehat{\alpha}_m = \|T^A_{m+1,m}\|_F^2 \|(Y_m)_{m_r,:}\|_\infty^2$, $\widehat{\beta}_m = \|(Y_m)_{:,m_r}\|_\infty^2 \|(T^B_{m+1,m})^\top\|_F^2$. This provides the bound for $\|E_m\|_\infty$. □

The results described in this Section allow us to give Algorithm 3 of the LRSD-EBHess method.

In Algorithms 2 and 3, we first divided into the time interval $[t_0, T]$ into N sub-intervals of length h. Then we compute $Y_m(t_i)$, with $t_i = t_0 + i\,h$, for $i = 0, 1, \ldots, N$ and so the approximate solutions $X_m(t_i)$ that can be written as products two low-rank matrix functions. In the next, we calculate

Input: The matrices $A \in \mathbb{R}^{n\times n}, B \in \mathbb{R}^{s\times s}$, $E \in \mathbb{R}^{n\times r}$, $F \in \mathbb{R}^{s\times r}$, the initial and final times t_0, T, a tolerance $tol > 0$, m_{max} a maximum number of iterations, k a step-size parameter and N the number of mesh points in the time partioning, τ the tolerance for the truncated SVD.

Output: The approximate solution $X_m(t_N)$.

Compute $h = \frac{T-t_0}{N}$.
for $m = 1, \ldots, m_{max}$ **do**
  Use the extended block Hessenberg algorithm to the pairs (A, E) and $(B^\top, F)$ to compute the bases $\mathcal{V}_m = [V_1, \cdots, V_m]$ and $\mathcal{W}_m = [W_1, \cdots, W_m]$ and also the the block upper-Hessenberg matrices $\mathcal{T}_m^A$ and $\mathcal{T}_m^B$;
  if $\texttt{rem}(m, k) = 0$ **then**
    for $i = 1, \ldots, N$ **do**
      Compute $t_i = t_0 + i\,h$;
      To obtain $Y_m(t_i)$, apply the BDF method to solve the reduced Sylvester differential equation (31) with the step-size h, see [18];
      Compute $r_m(t_i)$ that is given as (24);
      Compute the SVD of $Y_m(t_i)$: $Y_m(t_i) = \widehat{U}\widehat{S}\widehat{V}^\top$, where $\widehat{S} = \texttt{diag}(\sigma_1, \ldots, \sigma_{2mr})$, and $\sigma_1 \geq \ldots \geq \sigma_{2mr}$;
      Find the scalar ℓ such that $\sigma_\ell > \tau$ and let $\widehat{S}_\ell = \texttt{diag}(\sigma_1, \ldots, \sigma_\ell)$;
      Form $\mathbb{Z}_m^A = \mathcal{V}_m \widehat{U}_\ell \widehat{S}_\ell^{1/2}$ and $\mathbb{Z}_m^B = \mathcal{W}_m \widehat{V}_\ell \widehat{S}_\ell^{1/2}$, then compute the approximate solution $X_m(t_i) \approx \mathbb{Z}_m^A (\mathbb{Z}_m^B)^\top$;
    end for
    if $\max\{r_m(t_1), \ldots, r_m(t_N)\} < tol$ **then**
      Stop;
    end if
  end if
end for
=0

Algorithm 3: The low-rank Sylvester differential extended block Hessenberg method (LRSD-EBHess).

the residual associated with the approximate solution $X_m(t_i)$, i.e., $R_m(t_i)$, for $i = 1, 2, \ldots, N$. If $\|R_m(T_N)\|_F < tol\,\|E\|_F\|F\|_F$, the aforementioned algorithms will be stopped. In such situations, $X_m(T_N)$ computed by Algorithms 2 and 3 is the reasonable approximate solution of the Sylvester differential equation.

5. Numerical results

In this section, some numerical examples are given to examine the efficiency and potential of our proposed method. We compare Algorithms 2 and 3 with the ones presented in [18].

It should be pointed out that the initial condition matrix X_0 is a zero matrix in all examples. Moreover, different time intervals $[t_0, T]$ are considered and $t_0 = 0$ is fixed, while T is specified in each example. The time interval [0,T] is divided into N sub-intervals of length $h = \frac{T-t_0}{N}$.

All the numerical examples were implemented in MATLAB and have been performed on an Intel(R) Core(TM) i5 with 2.67 GHz processing speed and 8 GB memory. The following Matlab functions are applied to execute the different algorithms used in this work.

- `expm` : This function allows us to compute the exponential of a square matrix. It is based on a scaling and squaring algorithm with a Padè approximation [23].
- `lyap` : This allows us to obtain the solution of Sylvester and Lyapunov matrix equations. The MATLAB command is as follows:

$$X = \texttt{lyap}(A, B, C),$$

  where X is the solution of the Sylvester matrix equation $AX+XB = C$.
- `rand` : This allows us to create a random matrix.

Furthermore, when the extended block Hessenberg process is used to obtain an approximate solution to the low-rank Sylvester differential equation, the iterations are stopped just after the dimension of the extended block Krylov subspace produced by the extended block Hessenberg process or the extended block Arnoldi process reaches a maximum value $m = m_{max} = 200$ or instantly when the Frobenius norm of the true residual norm at the final time, i.e., $\|R_m\|_F$ calculated by the algorithm is lower than $10^{-10}\alpha$, where $\alpha = \|E\|_F\|F\|_F$. Also, the factors of the right-hand side matrix, E, and F were generated randomly. Also, it is worth noting that we obtain $W = A^{-1}V$ by solving the matrix equation $AW = V$ in Algorithm 1 via the block GMRES method with the block Krylov subspace of dimension 5.

Example 5.1. *The main aim of this example is to compare the solution estimated by Algorithm 3 with the one acquired via the integral formula* (14) *and some classical ordinary differential equation's solvers, such as* `ode23s`, `ode15s`, `ode45`, `ode23tb` *and the algorithm based on the extended block Arnoldi process [19]. To this end, we consider the benchmark described in [15] that is briefly explained here.*

Assume that

$$P = \begin{pmatrix} 3 & 8 & -19 \\ -1 & -5 & 11 \\ 0 & -1 & 2 \end{pmatrix}, \quad Q = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix},$$

are two nilpotent matrices of the index $I_0 = 3$ and $P_0 \in \mathbb{R}^{p_0 \times p_0}$, $Q_0 \in \mathbb{R}^{q_0 \times q_0}$ are given. In this example, we select the matrices

$$P_0 = \texttt{gallery}(\texttt{'leslie'}, \texttt{p}_0), \quad Q_0 = \texttt{gallery}(\texttt{'minij'}, \texttt{q}_0).$$

Moreover, assume that the coefficient matrices A and B are defined as

$$A = \alpha\, I_n + P_0 \otimes P, \quad B = \beta\, I_s + Q_0 \otimes Q,$$

where $n = 3p_0$, $s = 3q_0$ and the real numbers α, β satisfy $\alpha + \beta < 0$.

In view of (14)*, the unique solution of the Sylvester differential matrix equation*

$$\begin{cases} \dot{X}(t) = AX(t) + X(t)B + EF^\top, \\ X(t_0) = X_0, \end{cases}$$

is given by

$$X^*(t) = e^{(t-t_0)A} X_0 e^{(t-t_0)B} + \int_{t_0}^{t} e^{(t-u)A} EF^\top e^{(t-u)B} du.$$

According to [15], it can be shown that $X^*(t)$ is as

(34)

$$X^*(t) = \sum_{i=0}^{I_0-1} \sum_{j=0}^{I_0-1} \Big((t-t_0)^{i+j} e^{(\alpha+\beta)(t-t_0)} M_{i,j}(X_0) + J_{i+j}(t) M_{i,j}(EF^\top) \Big),$$

where $J_\ell(t) = \int_{t_0}^{t} (t-u)^\ell e^{(\alpha+\beta)(t-u)} du$ and $M_{i,j}(C) = \frac{1}{i!j!}(P_0^i \otimes P^i) C (Q_0^j \otimes Q^j)$, for $i, j = 0, 1, \ldots, I_0 - 1$. Note that the integral formula $J_\ell(t)$ can be obtained by the following recursion

$$J_\ell(t) = \frac{1}{\alpha+\beta} \Big((t-t_0)^k e^{(\alpha+\beta)(t-t_0)} - \ell\, J_{\ell-1}(t) \Big), \ \ell \geq 1,$$

in which $J_0(t) = \frac{1}{\alpha+\beta} \Big(e^{(\alpha+\beta)(t-t_0)} - 1 \Big)$.

It is necessary to point out that the methods mentioned in this example compare from the perspective of the run time, the absolute error, and the relative error. As you observed from Table 2, the execution in seconds is denoted by the CPU, and the absolute error of the final time is denoted by $er(t_N)$, that is

$$er(t_N) = \|X_m(t_N) - X_m(t_{N-1})\|_F.$$

The numerical results reported in Tabel 2 show that the Algorithms based on extended block Krylov subspace have more speed convergence and are more accurate. In addition, we depict the behavior of the relative error norm

$$\frac{\|X_m(t_i) - X^*(t_i)\|_F}{\|X^*(t_i)\|_F},$$

with $t_i = t_0 + i\, h$, for $i = 1, 2, \ldots, N$. Note that $X^*(t_i)$ indicates the exact solution computed (34) *in time t_i.*

In Fig. 1, to compare the relative error Frobenius norm convergence, the $\texttt{log}_{10}$ *plot of the relative error norms in terms of the number of iterations are depicted for two different time intervals. As seen in this figure, the solutions obtained by the LRSD-EBHess and LRSD-EBA are more accurate than those given by* `ode15s`*,* `ode23s`*,* `ode45` *and* `ode23tb`*. Also, it should be highlighted*

TABLE 1. The CPU-time and the absolute error norm of Example 5.1 with $r = 3$, $k = 5$, $\alpha = -20$, $\beta = -6$ and $t_0 = 0$. To obtain $Y_m(t_i)$ in the LRSD-EBHess and LRSD-EBA methods, the BDF3 method is used

	$T = 2, N = 10$		$T = 10, N = 20$	
Method	*CPU(s)*	$er(t_N)$	*CPU(s)*	$er(t_N)$
LRSD-EBHess	*0.0468*	*3.1569e-03*	*0.0625*	*7.2656e-08*
LRSD-EBA	*0.0781*	*1.5958e-02*	*0.0156*	*1.2320e-10*
`ode15s`	*32.406*	*2.3385e-02*	*30.906*	*1.3620e-01*
`ode23s`	*2639.3*	*3.8844e-02*	*3066.0*	*2.4417e-01*
`ode45`	*1.8438*	*2.7184e-02*	*7.5625*	*8.2308e-02*
`ode23tb`	*41.609*	*3.1969e-02*	*93.562*	*1.0938e-01*

that the convergence curves of the `ode15s`, `ode23s`, `ode45` *and* `ode23tb` *methods are overlapped with each other.*

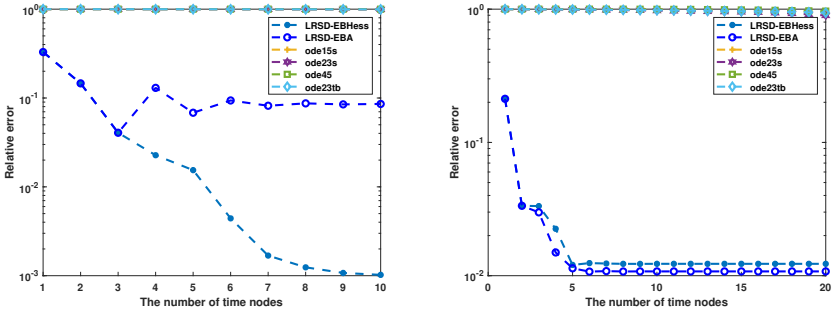

FIGURE 1. Example 5.1: Comparison between the relative error norms of the different methods. Left figure: $T = 2, N = 10$. Right figure: $T = 10, N = 20$.

Example 5.2. *In this example, we examine the performances of LRSD-EBHess, LRSD-EBA, EBHess-EXP, and EBA-EXP in two cases. In these cases, the coefficients matrices of the Sylvester differential matrix equation* (1) *are defined as different forms. In addition, using the MATLAB function* `rand`*, the right-hand side matrix is set to be* $EF^\top = \texttt{rand}(n, r) * \texttt{rand}(r, s)$.

Note that these methods compare from the viewpoint of the CPU time, the residual norm, the upper bound of the true residual norm $\tilde{r}_m(T)$ *and the seminorm* $r_m(T)$ *at the final time. As it is observed from Tables 2 and 3, the CPU time in seconds is denoted by the CPU, and the true residual norm of the final time is denoted by* $\|R_m(T)\|_F$.

Case 1 : *In this case, the matrix A is selected from the Matrix Market web server [14]. Again, the matrix B is derived from the discretization*

of the operator

$$L_B(u) = \Delta\, u - f_1^B(x,y)\frac{\partial u}{\partial x} - f_2^B(x,y)\frac{\partial u}{\partial y} - f_3^B(x,y)\, u,$$

on the unit on the unit square $\Omega = [0,1] \times [0,1]$ *with homogeneous Dirichlet bonudary conditions. The functions* $f_1^B(x,y), f_2^B(x,y)$ *and* $f_3^B(x,y)$ *are determined in Table 2.*

To produce the coefficient matrix B*, we applied the* `fdm_2d_matrix` *function from the LYAPACK toolbox [24] in the form*

$$B = \texttt{fdm\_2d\_matrix}(s_0, f_1^B(x,y), f_2^B(x,y), f_3^B(x,y)),$$

where s_0 *is the number of inner grid points in each direction when discretizing the operators* L_B*. This results* $B \in \mathbb{R}^{s\times s}$ *with* $s = s_0^2$*. We point out that in this case, we consider the parameter values* $r = 3$*,* $k = 1$*,* $n \in \{1104, 4960, 17758\}$*, and* $s = 400$*. Furthermore, we take the time interval* $[t_0, T] = [0, 2]$ *with the step-size* $h = 0.2$*.*

In Table 2, the number of iterations, CPU time, the true residual norm, the upper bound of the true residual norm and the semi-norm are displayed. As seen from Table 2, when we use the BDF1 method to obtain $Y_m(t_i)$*, for* $i = 1, 2, \ldots, N$*, the LRSD-EBHess method is superior to the LRSD-EBA method in terms of CPU time for all test problems. Moreover, it can be seen from Table 3 that the EBHess-EXP works better than EBA-EXP in CPU times except for the test problem* `memplus`*.*

TABLE 2. The number results for Case 1 of Example 5.2 with $r = 3$ and $k = 1$.

Test Problem	*Method*	*BDF*	*iter.*	*CPU(s)*	$\|R_m(T)\|_F$	$\tilde{r}_m(T)$	$r_m(T)$
$A =$ ***sherman4***		*1*	*9*	*1.2188*	*5.9687e-08*	*5.9687e-08*	*1.7387e-09*
$B =$ `fdm`$(\cos(xy), e^{y^2x}, 100)$	*LRSD-EBHess*	*2*	*9*	*1.5313*	*1.7284e-08*	*1.7284e-08*	*4.9755e-10*
$n = 1104, s = 400$		*3*	*9*	*1.1875*	*5.9700e-08*	*5.9700e-08*	*1.7391e-09*
		1	*9*	*1.3281*	*1.1697e-08*	*1.1697e-08*	*1.1697e-08*
	LRSD-EBA	*2*	*9*	*1.4531*	*5.2516e-09*	*5.2516e-09*	*5.2516e-09*
		3	*9*	*2.1719*	*1.1370e-08*	*1.1370e-08*	*1.1370e-08*
$A =$ ***add32***		*1*	*3*	*1.4063*	*2.3150e-09*	*2.3150e-09*	*2.0904e-11*
$B =$ `fdm`$(10xy, e^{yx^2}, 20y)$	*LRSD-EBHess*	*2*	*7*	*2.8281*	*5.1348e-09*	*5.1348e-09*	*1.8025e-11*
$n = 4960, s = 400$		*3*	*6*	*2.8750*	*1.5951e-08*	*1.5951e-08*	*8.0691e-11*
		1	*3*	*1.5156*	*4.8819e-10*	*4.8819e-10*	*4.8819e-10*
	LRSD-EBA	*2*	*6*	*2.7969*	*8.7041e-09*	*8.7041e-09*	*8.7041e-09*
		3	*6*	*3.1250*	*1.9173e-08*	*1.9173e-08*	*1.9173e-08*
$A =$ ***memplus***		*1*	*4*	*4.6406*	*8.6898e-11*	*8.6898e-11*	*2.4696e-11*
$B =$ `fdm`$(100y, yx^2, xy)$	*LRSD-EBHess*	*2*	*6*	*6.5000*	*9.6854e-09*	*9.6854e-09*	*3.0161e-11*
$n = 17758, s = 400$		*3*	*5*	*6.2813*	*7.9811e-08*	*7.9811e-08*	*2.2549e-10*
		1	*3*	*5.3125*	*3.6381e-08*	*3.6381e-08*	*3.6381e-08*
	LRSD-EBA	*2*	*5*	*8.0313*	*3.1467e-08*	*3.1467e-08*	*3.1467e-08*
		3	*5*	*9.7656*	*5.0977e-08*	*5.0977e-08*	*5.0977e-08*

TABLE 3. The number results for Case 1 of Example 5.2 with $r = 3$ and $k = 1$.

Test Problem	*Method*	*iter.*	*CPU(s)*	$\|R_m(T)\|_F$	$\tilde{r}_m(T)$	$r_m(T)$
$A =$ ***sherman4***	*EBHess-EXP*	*3*	*7.0781*	*2.2875e-10*	*2.2875e-10*	*2.9277e-12*
$B = \texttt{fdm}(\cos(xy), e^{y^2x}, 100)$						
$n = 1104, s = 400$	*EBA-EXP*	*4*	*17.688*	*5.9382e-09*	*5.9382e-09*	*5.9382e-09*
$A =$ ***add32***	*EBHess-EXP*	*3*	*7.6094*	*2.0922e-09*	*2.0922e-09*	*1.0296e-11*
$B = \texttt{fdm}(10xy, e^{yx^2}, 20y)$						
$n = 4960, s = 400$	*EBA-EXP*	*4*	*17.922*	*5.2030e-08*	*5.2030e-08*	*5.2030e-08*
$A =$ ***memplus***	*EBHess-EXP*	*3*	*9.3438*	*1.0541e-10*	*1.0541e-10*	*3.0370e-13*
$B = \texttt{fdm}(100y, yx^2, xy)$						
$n = 17758, s = 400$	*EBA-EXP*	*2*	*9.2188*	*4.1887e-13*	*4.1887e-13*	*4.1887e-13*

In Fig. 2, $\texttt{log}_{10}$ *plot of the true residual norm and upper bound of the true residual norm in terms of the number of iterations are represented at the final time when the matrix A is* `sherman4`*. As shown in Fig. 2, we see that the upper bound* $\tilde{r}_m(T)$ *given by* (32) *enables us to mimic the true residual norm* $\|R_m(T)\|_F$*.*

Fig. 3 depicts $\texttt{log}_{10}$ *plot of the true residual norm and the relative true residual norm in terms of the number of iterations at the final time when the matrix A is* `sherman4`*. As observed from Fig. 3, the EBHess-EXP outperforms the other methods, and the LRSD-EBHess and LRSD-EBA methods have the same convergence behavior.*

Case 2 : *In this case, the matrices A and B are generated by the operators*

$$L_A(u) = \Delta\, u - f_1^A(x,y)\frac{\partial u}{\partial x} - f_2^A(x,y)\frac{\partial u}{\partial y} - f_3^A(x,y)\, u,$$
$$L_B(u) = \Delta\, u - f_1^B(x,y)\frac{\partial u}{\partial x} - f_2^B(x,y)\frac{\partial u}{\partial y} - f_3^{\boldsymbol{B}}(x,y)\, u,$$

where

$$f_1^A(x,y) = (x + 10y^2), f_2^A(x,y) = \sqrt{2x^2 + y^2}, f_3^A(x,y) = x^2 - y^2,$$
$$f_1^B(x,y) = 10\, x\, y, f_2^B(x,y) = e^{-x^2-y^2}, f_3^B(x,y) = \frac{1}{1 + x^2 + y^2}.$$

Herein, the main goal is to verify the numerical behavior of the LRSD-EBHess, LRSD-EBA, EBHess-EXP, and EBA-EXP, when the parameter values r and T increase. To do these, we report two types of numerical results. The first is when $r \in \{1, 2, 3\}$ *and T is fixed and the second is when the final time* $T \in \{5, 10, 50, 100\}$ *and r is fixed. Also, we use BDF1 in the LRSD-EBHess, LRSD-EBA methods to obtain* $Y_m(t_i)$*, for* $i = 1, 2, \ldots, N$*.*

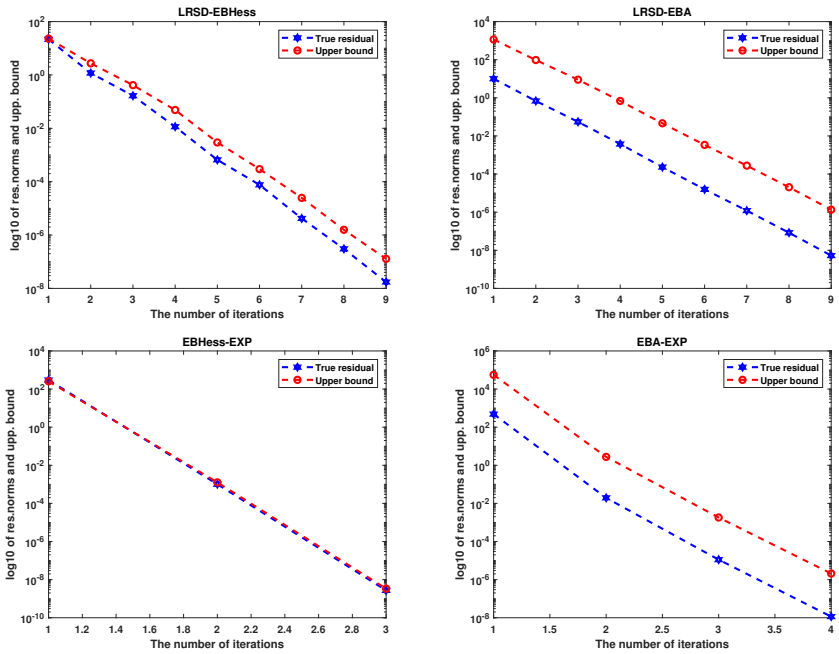

FIGURE 2. Example 5.2: Comparison between the true residual norm and the corresponding upper bound of the different methods for the first test problem.

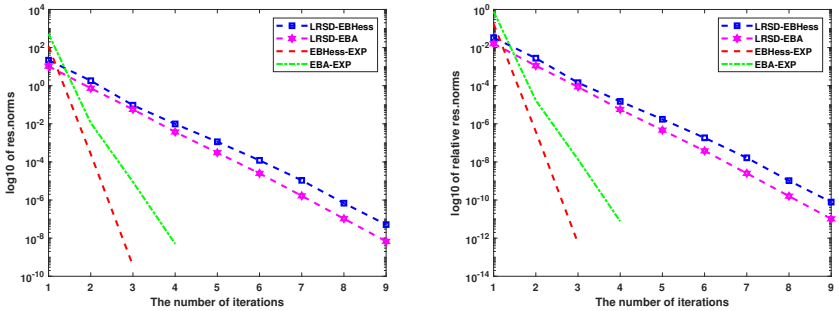

FIGURE 3. Example 5.2: Comparison between the true residual norm and the relative true residual norm of the different methods for the first test problem.

The numerical results listed in Table 4 demonstrate that when the rank of the matrices E and F increase, the run time of the LRSD-EBHess, LRSD-EBA, EBHess-EXP and EBA-EXP methods also increase.

TABLE 4. The numerical results for Case 2 of Example 5.2 with $n = 1600, s = 400, k = 5, N = 10, T = 2$. To obtain $Y_m(t_i)$ in the LRSD-EBHess and LRSD-EBA methods, the BDF1 method is used.

Method	r	*iter.*	*CPU(s)*	$\|R_m(T)\|_F$	$\tilde{r}_m(T)$	$r_m(T)$
LRSD-EBHess	1	20	0.6250	2.2209e-08	2.2209e-08	2.8013e-10
	2	10	0.4593	4.8700e-06	4.8700e-06	8.9718e-08
	3	10	0.5468	5.0479e-06	5.0479e-06	7.7234e-08
LRSD-EBA	1	15	0.5937	3.8052e-10	3.8052e-10	3.8052e-10
	2	10	0.5000	3.4696e-06	3.4696e-06	3.4696e-06
	3	10	0.5937	2.7033e-06	2.7033e-06	2.7033e-06
EBHess-EXP	1	5	2.5625	4.4018e-10	4.4018e-10	3.6135e-12
	2	5	5.6094	2.8866e-09	2.8866e-09	2.1051e-11
	3	5	9.1406	2.8782e-09	2.8782e-09	2.3186e-11
EBA-EXP	1	5	5.0156	1.1875e-11	1.1875e-11	1.1875e-11
	2	5	6.6563	1.2436e-11	1.2436e-11	1.2436e-11
	3	5	11.641	1.6606e-11	1.6606e-11	1.6606e-11

In Table 5, we report the numerical results of different methods when the final time T increases. It can be viewed that when T becomes large, the run time of the EBHess-EXP and EBA-EXP will be large. However, this is not true for methods LRSD-EBHess and LRSD-EBA.

TABLE 5. The numerical results for Case 2 of Example 5.2 with $n = 1600, s = 400, k = 5, N = 10$. To obtain $Y_m(t_i)$ in the LRSD-EBHess and LRSD-EBA methods, the BDF1 method is used.

Method	T	*iter.*	*CPU(s)*	$\|R_m(T)\|_F$	$\tilde{r}_m(T)$	$r_m(T)$
LRSD-EBHess	5	15	1.0938	7.1378e-10	7.1378e-10	1.5690e-11
	10	15	1.0938	5.8743e-09	5.8743e-09	8.6202e-11
	50	15	1.0781	1.3136e-09	1.3136e-09	3.4556e-11
	100	15	1.0938	6.5172e-08	6.5172e-08	1.1966e-09
LRSD-EBA	5	15	1.6563	1.7615e-10	1.7615e-10	1.7615e-10
	10	15	1.5313	1.2181e-10	1.2181e-10	1.2181e-10
	50	15	1.5469	1.6575e-10	1.6575e-10	1.6575e-10
	100	15	1.7344	2.1776e-10	2.1776e-10	2.1776e-10
EBHess-EXP	5	5	9.0938	1.6097e-09	1.6097e-09	1.2535e-11
	10	5	8.8906	2.6451e-10	2.6451e-10	2.1459e-12
	50	5	12.938	1.6224e-09	1.6224e-09	1.3135e-11
	100	5	13.813	5.6878e-10	5.6878e-10	6.4169e-12
EBA-EXP	5	5	15.234	1.5331e-11	1.5331e-11	1.5331e-11
	10	5	16.203	1.5993e-11	1.5993e-11	1.5993e-11
	50	5	18.984	1.4434e-11	1.4434e-11	1.4434e-11
	100	5	19.859	1.7531e-11	1.7531e-11	1.7531e-11

Example 5.3. *Here, we consider the time interval $[0, T]$ for $T = 1$ in which the number of sub-intervals is $N = 10$. The matrix A is selected from the Matrix Market web server [14]. We consider the particular case $B = A^\top$ and $F = E$*

and report the results obtained when solving low-rank differential Lyapunov equations. The obtained results for $r = 5$ are listed in Table 6.

Table 6 shows that CPU time in the EBHess-EXP method is less than the other methods. Moreover, its number of iterations is far less than the number of iterations of the LRSD-EBHess and LRSD-EBA iterative methods. If we compare the numerical behavior of the LRSD-EBHess and LRSD-EBA iterative methods, we can see that the proposed method LRSD-EBHess needs less execution time.

TABLE 6. The numerical results of Example 5.3 with $r = 5, k = 2, N = 10$. To obtain $Y_m(t_i)$ in the LRSD-EBHess and LRSD-EBA methods, the BDF1 method is used.

Test Problem	*Method*	*iter.*	*CPU(s)*	$\|R_m(T)\|_F$	$\tilde{r}_m(T)$	$r_m(T)$
$A =$ ***plat1919***	*LRSD-EBHess*	*14*	*9.3750*	*1.0677e-08*	*1.0677e-08*	*8.5437e-10*
$B = A^\top$	*LRSD-EBA*	*14*	*11.313*	*6.3409e-09*	*6.3409e-09*	*6.3409e-09*
$n = 1919, s = 1919$	*EBHess-EXP*	*4*	*9.6875*	*1.8054e-11*	*1.8054e-11*	*7.1955e-14*
	EBA-EXP	*4*	*13.953*	*2.1612e-14*	*2.1612e-14*	*2.1612e-14*
$A =$ ***psmigr_3***	*LRSD-EBHess*	*10*	*13.922*	*3.4190e-09*	*3.4190e-09*	*8.7934e-12*
$B = A^\top$	*LRSD-EBA*	*10*	*13.922*	*2.6176e-10*	*2.6176e-10*	*2.6176e-10*
$n = 3140, s = 3140$	*EBHess-EXP*	*4*	*12.500*	*2.8748e-16*	*2.8748e-16*	*6.8842e-19*
	EBA-EXP	*4*	*16.609*	*4.4529e-20*	*4.4529e-20*	*4.4529e-20*
$A =$ ***add32***	*LRSD-EBHess*	*6*	*22.031*	*7.7184e-12*	*7.7184e-12*	*3.6201e-14*
$B = A^\top$	*LRSD-EBA*	*6*	*22.438*	*3.5309e-12*	*3.5309e-12*	*3.5309e-12*
$n = 4960, s = 4960$	*EBHess-EXP*	*2*	*11.016*	*9.6317e-09*	*9.6317e-09*	*3.4505e-11*
	EBA-EXP	*2*	*13.875*	*1.6893e-10*	*1.6893e-10*	*1.6893e-10*

6. Conclusion

We have developed two techniques to solve the Sylvester differential matrix equation as well these methods can be used for solving the Lyapunov differential matrix equation. The first one is related to the exponential expression of the exact solution which can be derived by the Matlab function `expm` and a quadrature formula. The second approach is based on projecting the initial value problem onto an extended block Krylov subspace. In this approach, a low-dimensional Sylvester differential matrix equation which is solved using the BDF method is derived. This method is based on extended block Hessenberg with maximum strategy. we have also established a representation of the residual norm. Numerical test examples show that these methods in comparison with the method based on the extended block Arnoldi is more effective. In the future, our main goal is to estimate the approximate solution of these initial value problems exploiting deflation techniques, see [8, 9, 32].

7. Acknowledgements

We would like to thank the referees for the valuable remarks and their helpful suggestions.

References

[1] Abdaoui, I., Elbouyahyaoui, L., & Heyouni, H. (2020). The simpler block CMRH method for linear systems. Numerical Algorithms, 84, 1265-1293. https://doi.org/10.1007/s11075-019-00814-7

[2] Abidi, O., Heyouni, M., & Jbilou, K. (2017). On some properties of the extended block and global Arnoldi methods with applications to model reduction. Numerical Algorithms, 75, 285-304. https://doi.org/10.1007/s11075-016-0207-7

[3] Abou-Kandil, H., Freiling, G., Ionescu, V. & Jank, G. (2003). Matrix Riccati equations in control and systems theory. Birkhäuser.

[4] Addam, M., Heyouni, M., & Sadok, H. (2017). The block Hessenberg process for matrix equations. Electronic Transactions on Numerical Analysis, 46, 460-473.

[5] Agoujil, S., Bentbib, A. H., Jbilou, K. & Sadek, El M. (2014). A minimal residual norm method for large-scale Sylvester matrix equations. Electronic Transactions on Numerical Analysis, 43, 45-59. https://doi.org/10.1016/j.amc.2006.10.011

[6] Amato, F., Ambrosino, R., Ariola, M., Cosentino, C., & De Tommasi, G. (2014). Finite-Time stability and control. Springer.

[7] Antoulas, A. C. (2005). Approximation of Large-Scale Dynamical Systems. SIAM.

[8] Azizizadeh, N., Tajaddini, A., & Rafiei, R.(2023). Implicitly restarted global Krylov subspace methods for matrix equations $AXB = C$. Mathematical Sciences, https://doi.org/10.1007/s40096-023-00516-1

[9] Azizizadeh, N., Tajaddini, A., & Wu, G.(2018). Weighted and deflated global GMRES algorithms for solving large Sylvester matrix equations. Numerical Algorithms, 82, 155-181. https://doi.org/10.1007/s11075-018-0597-9

[10] Behr, M., Benner, P., & Heiland, J. (2019). Solution formulas for differential Sylvester and Lyapunov equations. Calcolo, 56(4), 51-84. https://doi.org/10.1007/s10092-019-0348-x

[11] Benner, P., & and Mena, H. (2004). BDF methods for large-scale differential Riccati equations. Proceedings of Mathematical Theory of Network and Systems, MTNS.

[12] Benner, P. & Mena, H. (2013). Rosenbrock methods for solving Riccati differential equations. IEEE Transactions on Automatic Control, 58(11), 2950-2957. https://doi.org/10.1109/TAC.2013.2258495

[13] Blanquer, I., Claramunt, H., Hernández, V., & Vidal, A. M. (1998). Solving the Generalized Lyapunov Equation by the Bartels-Stewart Method using Standard Software Libraries for Linear Algebra Computations. IFAC Proceedings Volumes, 31(18), 387-392.

[14] Boisvert, R., Pozo, R., Remington, K., Barrett, R., & Dongarra, J. (1997). The Matrix Market: A web resource for test matrix collections, in Quality of Numerical Software, Assessment and Enhancement, R. Boisvert, ed., Chapman & Hall, London, 125-137.

[15] Bouhamidi, A., Elbouyahyaoui, L., & Heyouni, M. (2024). The constant solution method for solving large-scale differential Sylvester matrix equations with time invariant coefficients. Numerical Algorithms, 96, 449-488. https://doi.org/10.1007/s11075-023-01653-3

[16] Butcher, J. C. (2008). Numerical methods for ordinary differential equations. John Wiley & Sons.

[17] Golub, G. H., Nash, S., & Van Loan, C. (1979). A Hessenberg-Schur method for the problem AX + XB = C. IEEE Transactions on Automatic Control, 24, 909-913. https://doi.org/10.1109/tac.1979.1102170

[18] Hached, M., & Jbilou, K. (2018). Computational Krylov-based methods for large-scale differential Sylvester matrix problems. Numerical Linear Algebra with Applications, 25(5), e2187. https://doi.org/10.1002/nla.2187

[19] Hached, M., & Jbilou, K. (2018). Numerical solutions to large-scale differential Lyapunov matrix equations. Numerical Algorithms, 79, 741-757. https://doi.org/10.1007/s11075-017-0458-y

[20] Hached, M., & Jbilou, K. (2020). Numerical methods for differential linear matrix equations via Krylov subspace methods. Journal of Computational and Applied Mathematics, 370, 112-647. https://doi.org/10.1016/j.cam.2019.112674
[21] Heyouni, M., & Jbilou, K. (2009). An extended block Arnoldi algorithm for large-scale solutions of the continuous-time algebraic Riccati equation. Electronic Transactions on Numerical Analysis, 33, 53-62. http://eudml.org/doc/130626
[22] Heyouni, M., Saberi-Movahed, F., & Tajaddini, A.(2019). On global Hessenberg based methods for solving Sylvester matrix equations. Computers & Mathematics with Applications, 77(1), 77-92. https://doi.org/10.1016/j.camwa.2018.09.015
[23] Higham, N. J. (2005). The scaling and squaring method for the matrix exponential revised. SIAM Journal on Matrix Analysis and Applications, 26(4), 1179-1193. https://doi.org/10.1137/04061101X
[24] Penzl, T. (2000). LYAPACK. A MATLAB Toolbox for large Lyapunov and Riccati equations, Model Reduction Problems, and Linear-Quadratic Optimal Control Problems.
[25] Rosenbrock, H. H., (1963). Some general implicit processes for the numerical solution of differential equations. The Computer Journal, 5(4), 329-330. https://doi.org/10.1093/comjnl/5.4.329
[26] Saad, Y. (1989). Numerical solution of large Lyapunov equations. Research Institute for Advanced Computer Science, NASA Ames Research Center.
[27] Saad, Y. (1992). Analysis of some Krylov subspace approximations to the matrix exponential operator, SIAM Journal on Numerical Analysis, 29, 209-228. https://doi.org/10.1137/0729014
[28] Sadek, E. M., Bentbib, A., Sadek, L., & Alaoui, H. (2020). Global extended Krylov subspace methods for large-scale differential Sylvester matrix equations. Journal of Applied Mathematics and Computing, 62, 157-177. https://doi.org/10.1007/s12190-019-01278-7
[29] Sadek, L., Sadek, El M. & Alaoui, H. T. (2022). On Some Numerical Methods for Solving Large Differential Nonsymmetric Stein Matrix Equations, Mathematical and Computational Applications, 27(4: 69), https://doi.org/10.3390/mca27040069
[30] Sadok, H. (1999). CMRH: A new method for solving nonsymmetric linear systems based on the Hessenberg reduction algorithm. Numerical Algorithms 20(4), 303-321. https://doi.org/10.1023/A:1019164119887
[31] Simoncini, V. (2007). A new iterative method for solving large-scale Lyapunov matrix equations. SIAM Journal on Scientific Computing, 29(3), 1268-1288. https://doi.org/10.1137/06066120X
[32] Tajaddini, A., Wu, G., Saberi-Movahed, F., & Azizizadeh, N.(2021). Two New Variants of the Simpler Block GMRES Method with Vector Deflation and Eigenvalue Deflation for Multiple Linear Systems. Journal of Scientific Computing, 86(9), 1-33. https://doi.org/10.1007/s10915-020-01376-w

Azita Tajaddini
Orcid number: 0000-0003-2513-194X
Department of Applied Mathematics
Faculty of Mathematics and Computer
Shahid Bahonar University of Kerman
Kerman, Iran
Email address: atajadini@uk.ac.ir