Solving Coupled Tensor Equations via Higher Order LSQR Methods

Masoud Hajarian

Abstract. Tensors have a wide application in control theory, data mining, chemistry, information sciences, documents analysis and medical engineering. The material here is motivated by the development of the efficient numerical methods for solving the coupled tensor equations

$$\begin{align*}
A_1 \ast_M X \ast_N B_1 + C_1 \ast_M Y \ast_N D_1 &= E_1, \\
A_2 \ast_M X \ast_N B_2 + C_2 \ast_M Y \ast_N D_2 &= E_2,
\end{align*}$$

with Einstein product. We propose the tensor form of the LSQR methods to find the solutions $X$ and $Y$ of the coupled tensor equations. Finally we give some numerical examples to illustrate that our proposed methods are able to accurately and efficiently find the solutions of tensor equations with Einstein product.

1. Introduction

Tensors are becoming increasingly used to describe and solve several problems of documents analysis [7], psychometrics [27], chemometrics [38] and medical engineering [35], for more details see [10, 21, 34]. Research related to tensors has increased dramatically in recent years [3, 4, 8, 9, 11, 13, 14, 20, 42]. Solving linear systems in higher dimensions is one of the most important research topics in tensors [2, 19, 31]. Ding and Wei [15] generalized the Jacobi, Gauss-Seidel and Newton methods to solve multilinear systems with $M$-tensors. From solving multi-linear $M$-equations is equivalent to solving nonlinear systems of equations where the involving functions are $P$-functions, He et al. derived a Newton-type algorithm for solving multi-linear $M$-equations [25]. Based on the rank-1 approximation of the coefficient tensor, Xie et al. obtained a new tensor method for solving symmetric $M$-tensor systems [45]. In the work of Dolgov and Savostyanov [16], the alternating minimal energy (AMEn) methods were introduced for solving high-dimensional symmetrical positive definite (SPD) linear systems. In [44], two neural network models were develop for solving nonsingular multi-linear tensor system. Wang et al. derived continuous time neural network and modified continuous time neural networks to solve a multi-linear system with $M$-tensors [43]. In [18], the greedy Tucker approximation algorithm was generalized for solving a tensor linear equation. Brazell et al. [5] introduced a higher order biconjugate gradient (HOBG) algorithm for solving a multilinear system of equations with Einstein product. Very recently, high order tensor equations such as the Stein tensor equation, the Lyapunov tensor equation and the Sylvester tensor equation have attracted great attention [12, 30, 37]. For example, Xu and Wang extended the biconjugate gradients (Bi-CG) and...
biconjugate residual (Bi-CR) algorithms for finding the solution of the high order Stein tensor equation [46]. In [29], the sensitivity of the continuous Lyapunov tensor equation was investigated. Influenced by Brazell et al.’s paper [5], the tensor equations with Einstein product have drawn much attention [26, 39]. For instance, Wang and Xu [41] proposed a conjugate gradient least squares (CGLS) algorithm to solve some tensor equations with Einstein product. In the following we will concentrate on the coupled tensor equations

\[
\begin{align*}
\mathcal{A}_1 \ast_M X \ast_N B_1 + C_1 \ast_M Y \ast_N D_1 &= E_1, \\
\mathcal{A}_2 \ast_M X \ast_N B_2 + C_2 \ast_M Y \ast_N D_2 &= E_2,
\end{align*}
\]

(1)

with Einstein product where \( \mathcal{A}_1, \mathcal{A}_2 \in \mathbb{R}^{i_1 \times \cdots \times i_M \times j_1 \times \cdots \times j_M} \), \( B_1, B_2 \in \mathbb{R}^{j_1 \times \cdots \times j_M \times k_1 \times \cdots \times k_M} \), \( C_1, C_2 \in \mathbb{R}^{j_1 \times \cdots \times j_M \times l_1 \times \cdots \times l_M} \), \( D_1, D_2 \in \mathbb{R}^{i_1 \times \cdots \times i_M \times k_1 \times \cdots \times k_M} \), \( E_1, E_2 \in \mathbb{R}^{i_1 \times \cdots \times i_M \times l_1 \times \cdots \times l_M} \) and \( X \in \mathbb{R}^{i_1 \times \cdots \times i_M \times p_1 \times \cdots \times p_M} \) and \( Y \in \mathbb{R}^{i_1 \times \cdots \times i_M \times q_1 \times \cdots \times q_M} \). The coupled tensor equations (1) have not been dealt with yet. In this article we aim to efficiently find the solutions of (1). Three numerical examples will be discussed in Section 3. Finally we give some conclusions in Section 4.

1.1. Notation and preliminary definitions

In this subsection, we present the notation system used in this article and the basic concepts of tensors. For positive integers \( l_1, l_2, \ldots, l_M \), an order \( M \) tensor \( \mathcal{A} = (a_{i_1, \ldots, i_M})_{i_1, \ldots, i_M} \) is a multidimensional array with \( L = l_1 l_2 \cdots l_M \) entries. We use \( \mathbb{R}^{i_1 \times \cdots \times i_M} \) to represent the set of the order \( M \) tensors of dimension \( l_1 \times \cdots \times l_M \) with entries from \( \mathbb{R} \). We now propose the Einstein product of two tensors. Given two tensors \( \mathcal{A} \in \mathbb{R}^{i_1 \times \cdots \times i_M \times l_1 \times \cdots \times l_M} \) and \( \mathcal{B} \in \mathbb{R}^{i_1 \times \cdots \times i_M \times j_1 \times \cdots \times j_M} \), the Einstein product of \( \mathcal{A} \) and \( \mathcal{B} \) is denoted by \( \mathcal{A} \ast_M \mathcal{B} \) [5, 17] and is defined by

\[
(\mathcal{A} \ast_M \mathcal{B})_{i_1, \ldots, i_M} = \sum_{j_1, \ldots, j_M} a_{i_1, \ldots, i_M} b_{j_1, \ldots, j_M}.
\]

The Einstein product of tensors has important applications in theory of relativity and continuum mechanics [5, 17, 28]. When \( M = N = L = 1 \), the Einstein product reduces to the standard matrix multiplication. The symbols

\[
(\mathcal{A}^T)_{i_1, j_1, \ldots, j_M} = (\mathcal{A})_{i_1, j_2, \ldots, j_M} \in \mathbb{R}^{i_1 \times \cdots \times i_M \times j_1 \times \cdots \times j_M},
\]

\[
\text{tr} (\mathcal{A}) = \sum_{i_1, \ldots, i_M} a_{i_1, \ldots, i_M},
\]

and

\[
\| \mathcal{A} \| = \left( \sum_{i_1, \ldots, i_M} (a_{i_1, \ldots, i_M})^2 \right)^{1/2},
\]

will, respectively, denote the transpose, the trace and the Frobenius norm of a given tensor \( \mathcal{A} \in \mathbb{R}^{i_1 \times \cdots \times i_M \times j_1 \times \cdots \times j_M} \). The inner product of two tensors \( \mathcal{A}, \mathcal{B} \in \mathbb{R}^{i_1 \times \cdots \times i_M \times j_1 \times \cdots \times j_M} \) is defined by

\[
\langle \mathcal{A}, \mathcal{B} \rangle = \text{tr}(\mathcal{B}^T \ast_M \mathcal{A}).
\]

Tensors \( \mathcal{A} \) and \( \mathcal{B} \) are called orthogonal if \( \langle \mathcal{A}, \mathcal{B} \rangle = 0 \). Given \( \mathcal{A}, \mathcal{B}, X \in \mathbb{R}^{i_1 \times \cdots \times i_M \times j_1 \times \cdots \times j_M}, C \in \mathbb{R}^{i_1 \times \cdots \times i_M \times k_1 \times \cdots \times k_M}, \]

\( Y \in \mathbb{R}^{k_1 \times \cdots \times k_M \times l_1 \times \cdots \times l_M} \) and \( \lambda \in \mathbb{R} \), it can be checked that [39, 41]

\[
\langle \mathcal{A}, \mathcal{B} \rangle = \text{tr}(\mathcal{B}^T \ast_M \mathcal{A}) = \text{tr}(\mathcal{A} \ast_N \mathcal{B}^{T}) = \text{tr}(\mathcal{A}^{T} \ast_M \mathcal{B}) = (\langle \mathcal{B}, \mathcal{A} \rangle),
\]

\[
\langle \lambda \mathcal{A}, \mathcal{B} \rangle = \lambda(\langle \mathcal{A}, \mathcal{B} \rangle),
\]

\[
\langle X, C \ast_M Y \rangle = (C^{T} \ast_M X, Y),
\]

\[
(\mathcal{C} \ast_M Y)^{T} = Y^{T} \ast_M \mathcal{C}.
\]
Definition 1. [41] Define the transformation \( \phi_{12} : \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N} \to \mathbb{R}^{I_1 \times J} \) with \( I = I_1 I_2 \ldots I_M, J = J_1 J_2 \ldots J_N \) and \( \phi_{12}(A) = A \) defined component-wise as

\[
(A)_{i_1 \ldots i_M} \to A_{i_1 \ldots i_M j_1 \ldots j_N},
\]

where \( A \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_M \times J_1 \times J_2 \times \ldots \times J_N}, A \in \mathbb{R}^{I_1 \times J}, s = i_m + \sum_{k=1}^{M-1} ((i_k - 1) \prod_{r=k+1}^{M} J_r) \) and \( t = j_n + \sum_{k=1}^{N-1} ((j_k - 1) \prod_{r=k+1}^{N} J_r) \).

Here is an example related to the above definition. Consider the tensor \( A \in \mathbb{R}^{2 \times 3 \times 2} \) as follows

\[
A(:, 1, 1) = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}, \quad A(:, 2, 1) = \begin{pmatrix} 7 & 8 & 9 \\ 10 & 11 & 12 \end{pmatrix},
\]

\[
A(:, 3, 1) = \begin{pmatrix} 13 & 14 & 15 \\ 16 & 17 & 18 \end{pmatrix}, \quad A(:, 1, 2) = \begin{pmatrix} 19 & 20 & 21 \\ 22 & 23 & 24 \end{pmatrix},
\]

\[
A(:, 2, 2) = \begin{pmatrix} 25 & 26 & 27 \\ 28 & 29 & 30 \end{pmatrix}, \quad A(:, 3, 2) = \begin{pmatrix} 31 & 32 & 33 \\ 34 & 35 & 36 \end{pmatrix}.
\]

It follows from Definition 1 that

\[
\phi_{12}(A) = A = \begin{pmatrix} 1 & 7 & 13 & 19 & 25 & 31 \\ 2 & 8 & 14 & 20 & 26 & 32 \\ 3 & 9 & 15 & 21 & 27 & 33 \\ 4 & 10 & 16 & 22 & 28 & 34 \\ 5 & 11 & 17 & 23 & 29 & 35 \\ 6 & 12 & 18 & 24 & 30 & 36 \end{pmatrix} \in \mathbb{R}^{6 \times 6}.
\]

Remark 1. It is worth mentioning that we can define the inverse function of \( \phi_{12} : \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_M \times J_1 \times J_2 \times \ldots \times J_N} \to \mathbb{R}^{I_1 \times J} \) as follows

\[
\phi_{12}^{-1} : \mathbb{R}^{I_1 \times J} \to \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_M \times J_1 \times J_2 \times \ldots \times J_N},
\]

with \( A_{st} \to (A)_{i_1 \ldots i_M j_1 \ldots j_N}, \) where the \( s \)-th column of the matrix \( A \) consists the \( s \)-th element in the set \( \{A(:, \ldots , 1, \ldots , N) \} \forall j_1, \ldots , j_N \). Here we sort all the elements in this set in lexicographic order, that is from \((1, \ldots , 1)\) to \((j_1, \ldots , j_N)\).

Lemma 1. [41] For \( A \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_M \times J_1 \times J_2 \times \ldots \times J_N}, X \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_M \times K_1 \times K_2 \times \ldots \times K_N}, \) and \( C \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_M \times K_1 \times \ldots \times K_N} \) we have

\[
A *_M X = C \iff \phi_{12}(A) *_{12k} (X) = \phi_{12k}(C).
\]

Using the above lemma, we can obtain the following well-known proposition which gives the solvability conditions and the expression of the solutions to the coupled tensor equations (1). In the following, we set

\[
A = \begin{pmatrix} \phi_{QD}(B_1)^T \otimes \phi_{12k}(A_1) & \phi_{QD}(B_1)^T \otimes \phi_{12k}(C_1) \\ \phi_{QD}(B_2)^T \otimes \phi_{12k}(A_2) & \phi_{QD}(B_2)^T \otimes \phi_{12k}(C_2) \end{pmatrix},
\]

\[
x = \begin{pmatrix} \text{vec}(\phi_{12k}(X)) \\ \text{vec}(\phi_{ST}(Y)) \end{pmatrix}, \quad b = \begin{pmatrix} \text{vec}(\phi_{12k}(E_1)) \\ \text{vec}(\phi_{12k}(E_2)) \end{pmatrix}.
\]

**Proposition 1.** The coupled tensor equations (1) have a unique solution pair \((X, Y)\) if and only if \(\text{rank}(A, b) = \text{rank}(A)\) and \(A\) has a full column rank. In that case, the solution pair \((X, Y)\) of (1) can be expressed as

\[
\begin{pmatrix} \text{vec}(\phi_{12k}(X)) \\ \text{vec}(\phi_{ST}(Y)) \end{pmatrix} = (A^T A)^{-1} A^T b.
\]
2. Higher order LSQR methods for (1)

Because there is extensive literature on iterative methods to efficiently and effectively solve the unsymmetric linear systems \([6, 36, 40]\), recently the development of iterative methods for solving linear matrix equations has become increasingly popular \([22–24]\). The main purpose of this section is to propose higher order LSQR methods for solving the coupled tensor equations (1). In this section, we begin by recalling the LSQR methods for solving the unsymmetric linear systems \(Ax = b\). As is well known, by using the Golub-Kahan bidiagonalization process, two types of the LSQR method were constructed in \([32]\) to compute an approximation solution of the linear systems \(Ax = b\) and unconstrained least-squares problem \(\min \|Ax - b\|\). First we present the pseudocode for the LSQR method in the following. Two types of the LSQR method can be summarized as follows \([32, 33]\).

Algorithm 1. Type 1 of LSQR method

\[
\tau(0) = 1; \xi(0) = -1; \omega(0) = 0; \epsilon(0) = 0; z(0) = 0; \beta(1) = \|b\|; \beta(1)u(1) = b; \alpha(1) = \|A^T u(1)\|; \alpha(1)v(1) = A^T u(1);
\]

For \(i = 1, 2, \ldots, \) until convergence, do:

\[
\xi(i) = -\xi(i-1)\beta(i)/\alpha(i); z(i) = z(i-1) + \xi(i)\omega(i); \epsilon(i) = (\tau(i-1) - \beta(i)\epsilon(i-1))/\alpha(i); \omega(i) = \omega(i-1) + \epsilon(i)\nu(i); \beta(i+1) = \|\epsilon(i)\|; \beta(i+1)u(i+1) = \epsilon(i)u(i+1); \tau(i) = \tau(i-1)\alpha(i)/\beta(i+1); \alpha(i+1) = \|A^T u(i+1) - \beta(i+1)\nu(i)\|; \alpha(i+1)v(i+1) = A^T u(i+1) - \beta(i+1)\nu(i); \gamma(i) = \beta(i+1)\xi(i)/\beta(i)\epsilon(i) - \tau(i); x(i) = z(i) - \gamma(i)\omega(i).
\]

Algorithm 2. Type 2 of LSQR method

\[
\theta(1) = \|A^T b\|; \theta(1)p(1) = A^T b; \rho(1) = \|Ax(1)\|; \rho(1)p(1) = Ax(1); \omega(1) = v(1)/\rho(1); \xi(1) = \theta(1)/\rho(1); x(1) = \xi(1)\omega(1);
\]

For \(i = 1, 2, \ldots, \) until convergence, do:

\[
\theta(i+1) = \|A^T p(i) - \rho(i)v(i)\|; \theta(i+1)v(i+1) = A^T p(i) - \rho(i)v(i); \rho(i+1) = \|Ax(i+1) - \theta(i+1)p(i)\|; \rho(i+1)p(i+1) = Ax(i+1) - \theta(i+1)p(i); \omega(i+1) = (v(i+1) - \theta(i+1)\omega(i))/\rho(i+1); \xi(i+1) = -\xi(i)\theta(i+1)/\rho(i+1); x(i+1) = x(i) + \xi(i+1)\omega(i+1).
\]

Theorem 1. \([33]\) LSQR algorithms return the minimum-norm solution.

It can be easily see that the scalars \(\alpha(i) \geq 0, \beta(i) \geq 0, \rho(i) \geq 0\) and \(\theta(i) \geq 0\) are selected to make \(\|u(i)\|_2 = 1\) and \(\|v(i)\|_2 = 1\), respectively. Also the stopping criterion can be considered as \(\|b - Ax(i)\|_2 < \epsilon\) where \(\epsilon\) is a small positive number (for more details see \([33]\)). It is worth noting that Algorithms 1 and 2 can be applied for solving the linear system \(Ax = b\) with the parameters \((6)\) and \((7)\). But obviously this system is a large system of equations and computing solution of such systems is a major computational challenge. Therefore it would be preferable to formulate Algorithms 1 and 2 in the tensor form. In order to obtain the tensor form of Algorithms 1 and 2, we substitute the parameters \((6)\) and \((7)\) into the above algorithms and then we obtain new parameters. It can be obtained that

\[
\alpha(1)v(1) = A^T u(1) = \begin{pmatrix}
\phi_{PQ}(B_1)^T \otimes \phi_{LK}(A_1) & \phi_{QT}(D_1)^T \otimes \phi_{LS}(C_1) \\
\phi_{PQ}(B_2)^T \otimes \phi_{LK}(A_2) & \phi_{QT}(D_2)^T \otimes \phi_{LS}(C_2)
\end{pmatrix} u(1),
\]

\[
\beta(i+1)u(i+1) = Ax(i) - \alpha(i)v(i) = \begin{pmatrix}
\phi_{PQ}(B_1)^T \otimes \phi_{LK}(A_1) & \phi_{QT}(D_1)^T \otimes \phi_{LS}(C_1) \\
\phi_{PQ}(B_2)^T \otimes \phi_{LK}(A_2) & \phi_{QT}(D_2)^T \otimes \phi_{LS}(C_2)
\end{pmatrix} v(i) - \alpha(i)u(i),
\]

\[
\alpha(i+1)v(i+1) = A^T u(i+1) - \beta(i+1)v(i) = \begin{pmatrix}
\phi_{PQ}(B_1)^T \otimes \phi_{LK}(A_1) & \phi_{QT}(D_1)^T \otimes \phi_{LS}(C_1) \\
\phi_{PQ}(B_2)^T \otimes \phi_{LK}(A_2) & \phi_{QT}(D_2)^T \otimes \phi_{LS}(C_2)
\end{pmatrix} w(i+1) - \beta(i+1)v(i),
\]

\[
\theta(1)v(1) = A^T b = \begin{pmatrix}
\phi_{PQ}(B_1)^T \otimes \phi_{LK}(A_1) & \phi_{QT}(D_1)^T \otimes \phi_{LS}(C_1) \\
\phi_{PQ}(B_2)^T \otimes \phi_{LK}(A_2) & \phi_{QT}(D_2)^T \otimes \phi_{LS}(C_2)
\end{pmatrix}
\begin{pmatrix}
\text{vec}(\phi_{LQ}(E_1)) \\
\text{vec}(\phi_{LQ}(E_2))
\end{pmatrix},
\]

\[
\theta(i+1)v(i+1) = A^T p(i) - \beta(i+1)v(i) = \begin{pmatrix}
\phi_{PQ}(B_1)^T \otimes \phi_{LK}(A_1) & \phi_{QT}(D_1)^T \otimes \phi_{LS}(C_1) \\
\phi_{PQ}(B_2)^T \otimes \phi_{LK}(A_2) & \phi_{QT}(D_2)^T \otimes \phi_{LS}(C_2)
\end{pmatrix} p(i) - \beta(i+1)v(i),
\]
\[ p(i+1)p(i+1) = \rho p(i+1) - \theta p(i+1)p(i) = \left( \phi_{\text{PC}}(B_1) \otimes \phi_{\text{KL}}(A_1) \right) \phi_{\text{QT}}(D_1) \otimes \phi_{\text{LS}}(C_1) \left( \phi_{\text{PC}}(B_2) \otimes \phi_{\text{KL}}(A_2) \right) \phi_{\text{QT}}(D_2) \otimes \phi_{\text{LS}}(C_2). \]  

In view of (9)-(11), we define new parameters as follows:

\[ p(i) = \begin{pmatrix} \text{vec}(\phi_{\text{LQ}}(P_1(i))) \\ \text{vec}(\phi_{\text{LQ}}(P_2(i))) \end{pmatrix}, \quad u(i) = \begin{pmatrix} \text{vec}(\phi_{\text{LQ}}(U_1(i))) \\ \text{vec}(\phi_{\text{LQ}}(U_2(i))) \end{pmatrix}, \quad r_0 = \begin{pmatrix} \text{vec}(\phi_{\text{LQ}}(\text{vec}(R_1(0)))) \\ \text{vec}(\phi_{\text{LQ}}(\text{vec}(R_2(0)))) \end{pmatrix}, \quad r(i) = \begin{pmatrix} \text{vec}(\phi_{\text{LQ}}(R_1(i))) \\ \text{vec}(\phi_{\text{LQ}}(R_2(i))) \end{pmatrix}. \]  

\[ v(i) = \begin{pmatrix} \text{vec}(\phi_{\text{EP}}(V_1(i))) \\ \text{vec}(\phi_{\text{ST}}(V_2(i))) \end{pmatrix}, \quad z(i) = \begin{pmatrix} \text{vec}(\phi_{\text{EP}}(Z_1(i))) \\ \text{vec}(\phi_{\text{ST}}(Z_2(i))) \end{pmatrix}, \quad q(i) = \begin{pmatrix} \text{vec}(\phi_{\text{EP}}(Q_1(i))) \\ \text{vec}(\phi_{\text{ST}}(Q_2(i))) \end{pmatrix}. \]  

where

\[ P_1(i), P_2(i), U_1(i), U_2(i), R_1(i), R_2(i), \tilde{R}_1(0), \tilde{R}_2(0) \in \mathbb{R}^{L \times \cdots \times L \times M \times X_1 \times \cdots \times X_N}, \]  

and

\[ V_1(i), Z_1(i), W_1(i), X(i) \in \mathbb{R}^{k_1 \times \cdots \times k_n \times \cdots \times k_N}, \quad V_2(i), Z_2(i), W_2(i), Y(i) \in \mathbb{R}^{L_1 \times \cdots \times L_N \times X_1 \times \cdots \times X_N}. \]  

Now using the above results and Remark 1, we can get the tensor forms of Algorithms 1 and 2 to solve the coupled tensor equations (1). The resulting algorithms are given as Algorithms 3 and 4, respectively.

**Algorithm 3. Type 1 of LSQR method for (1)**

\[ \tau(0) = 1; \quad \xi(0) = -1; \quad W'_1(0) = 0; \quad W_2(0) = 0; \quad w(0) = 0; \quad Z_1(0) = 0; \quad Z_2(0) = 0; \]

\[ \beta(1) = \left( ||E_1||^2 + ||E_2||^2 \right)^{1/2}; \quad U_1(1) = E_1/\beta(1); \quad U_2(1) = E_2/\beta(1); \]

\[ \alpha(1) = \left[ ||A_1 \otimes M U_1(1) \otimes N B_2 ||^2 + ||A_2 \otimes M U_2(1) \otimes N B_2 ||^2 + ||C_1 \otimes M U_1(1) \otimes N D_1 ||^2 + ||C_2 \otimes M U_2(1) \otimes N D_2 ||^2 \right]^{1/2}; \]

\[ \begin{align*} V_1(1) &= \left[ A_1^T \otimes M U_1(1) \otimes N B_2^T + A_2^T \otimes M U_2(1) \otimes N B_2^T \right]/\alpha(1); \\ V_2(1) &= \left[ C_1^T \otimes M U_1(1) \otimes N D_1^T + C_2^T \otimes M U_2(1) \otimes N D_2^T \right]/\alpha(1) \end{align*} \]

For \( i = 1, 2, \ldots \), until convergence, do:

\[ \begin{align*} \xi(i) &= -\xi(i-1)\beta(i)/\alpha(i); \\ Z_1(i) &= Z_1(i-1) + \xi(i)V_1(i); \\ Z_2(i) &= Z_2(i-1) + \xi(i)V_2(i); \\ w(i) &= (w(i-1) - \beta(i)w(i-1))/\alpha(i); \\ W'_1(i) &= W'_1(i-1) + w(i)V_1(i); \\ W_1(i) &= W_1(i-1) + w(i)V_1(i); \\ W'_2(i) &= W'_2(i-1) + w(i)V_2(i); \\ W_2(i) &= W_2(i-1) + w(i)V_2(i); \\ \beta(i+1) &= \left[ ||A_1 \otimes M U'_1(i) \otimes N B_1 + C_1 \otimes M V'_2(i) \otimes N D_1 - \alpha(i)U_1(i) ||^2 \\ + ||A_2 \otimes M U_2(i) \otimes N D_2 - \alpha(i)U_2(i) ||^2 \right]^{1/2}; \\ U_1(i+1) &= \left[ A_1 \otimes M V'_1(i) \otimes N B_1 + C_1 \otimes M V'_2(i) \otimes N D_1 - \alpha(i)U_1(i) \right]/\beta(i+1); \\ U_2(i+1) &= \left[ A_2 \otimes M U_1(i) \otimes N B_2 + C_2 \otimes M V'_2(i) \otimes N D_2 - \alpha(i)U_2(i) \right]/\beta(i+1); \\ \gamma(i) &= -\beta(i+1)/\beta(i); \\ \alpha(i+1) &= \left[ ||A_1^T \otimes M U'_1(i+1) \otimes N B_1^T + A_2^T \otimes M U_2(i+1) \otimes N B_2^T - \beta(i+1)V_1(i) ||^2 \\ + ||C_1^T \otimes M U_1(i+1) \otimes N D_1^T + C_2^T \otimes M U_2(i+1) \otimes N D_2^T - \beta(i+1)V_2(i) ||^2 \right]^{1/2}; \\ V_1(i+1) &= \left[ A_1^T \otimes M U'_1(i+1) \otimes N B_1^T + A_2^T \otimes M U_2(i+1) \otimes N B_2^T - \beta(i+1)V_1(i) \right]/\alpha(i+1); \\ V_2(i+1) &= \left[ C_1^T \otimes M U_1(i+1) \otimes N D_1^T + C_2^T \otimes M U_2(i+1) \otimes N D_2^T - \beta(i+1)V_2(i) \right]/\alpha(i+1); \\ \gamma(i) &= \beta(i+1)/\beta(i); \\ X(i) &= Z_1(i) - \gamma(i)W_1(i); \\ Y(i) &= Z_2(i) - \gamma(i)W_2(i). \end{align*} \]

**Algorithm 4. Type 2 of LSQR method for (1)**

\[ \begin{align*} \theta(1) &= \left( ||A_1 \otimes M E_1 \otimes N B_1^T + A_2 \otimes M E_2 \otimes N B_2^T ||^2 + ||C_1 \otimes M E_1 \otimes N D_1^T + C_2 \otimes M E_2 \otimes N D_2^T ||^2 \right)^{1/2}; \\ \begin{align*} V_1(1) &= \left[ A_1^T \otimes M E_1 \otimes N B_1^T + A_2^T \otimes M E_2 \otimes N B_2^T \right]/\theta(1); \end{align*} \]
First we consider the Sylvester tensor equation
with
\[ V_2(1) = [C^T_M X_N B + C_M Y_N D] \]
\[ \rho(1) = \sqrt{\|A_1 M V_1(1) N B_1 + C_1 M V_2(1) N D_1\|^2 + \|A_2 M V_1(1) N B_2 + C_2 M V_2(1) N D_2\|^2} / \theta(1); \]
\[ P_1(1) = [A_1 M V_1(1) N B_1 + C_1 M V_2(1) N D_1]/\rho(1); \]
\[ P_2(1) = [A_2 M V_1(1) N B_2 + C_2 M V_2(1) N D_2]/\rho(1); \]
\[ W_1(1) = V_1(1)/\rho(1); \]
\[ W_2(1) = V_2(1)/\rho(1); \]
\[ \xi(1) = \theta(1)/\rho(1); \]
\[ \chi(1) = \xi(1) W_1(1); \]
\[ \gamma(1) = \chi(1) W_2(1); \]

For \( i = 1, 2, ..., \) until convergence, do:
\[ \theta(i + 1) = \sqrt{\|A_1 M V_1(i) N B_1 + C_1 M V_2(i) N D_1 - \rho(i) V_1(i)\|^2 + \|A_2 M V_1(i) N B_2 + C_2 M V_2(i) N D_2 - \rho(i) V_2(i)\|^2} / \theta(i + 1); \]
\[ V_1(i + 1) = [A_1 M V_1(i) N B_1 + C_1 M V_2(i) N D_1 - \rho(i) V_1(i) ]/\theta(i + 1); \]
\[ V_2(i + 1) = [A_2 M V_1(i) N B_2 + C_2 M V_2(i) N D_2 - \rho(i) V_2(i) ]/\theta(i + 1); \]
\[ \rho(i + 1) = \sqrt{\|A_1 M V_1(i + 1) N B_1 + C_1 M V_2(i + 1) N D_1 - \theta(i + 1) P_1(i)\|^2 + \|A_2 M V_1(i + 1) N B_2 + C_2 M V_2(i + 1) N D_2 - \theta(i + 1) P_2(i)\|^2} / \theta(i + 1); \]
\[ P_1(i + 1) = [A_1 M V_1(i + 1) N B_1 + C_1 M V_2(i + 1) N D_1 - \theta(i + 1) P_1(i) ]/\theta(i + 1); \]
\[ P_2(i + 1) = [A_2 M V_1(i + 1) N B_2 + C_2 M V_2(i + 1) N D_2 - \theta(i + 1) P_2(i) ]/\theta(i + 1); \]
\[ W_1(i + 1) = (V_1(i + 1) - \theta(i + 1) W_1(i))/\rho(i + 1); \]
\[ W_2(i + 1) = (V_2(i + 1) - \theta(i + 1) W_2(i))/\rho(i + 1); \]
\[ \xi(i + 1) = -\xi(i) \theta(i + 1)/\rho(i + 1); \]
\[ \chi(i + 1) = \chi(i) + \xi(i + 1) W_1(i + 1); \]
\[ \gamma(i + 1) = \gamma(i) + \xi(i + 1) W_2(i + 1); \]

**Stopping criterion.** As a stopping criterion, we choose to satisfy
\[ \sqrt{\sum_{k=1}^{2} \|E_k - A_k M X_N B_k - C_k M Y(i) N D_k\|^2} = 0 \]
So our convergence criterion for the LSQR methods becomes
\[ \sqrt{\sum_{k=1}^{2} \|E_k - A_k M X(i) N B_k - C_k M Y(i) N D_k\|^2} \leq \text{tol}, \]
where \( \text{tol} \) is a chosen fixed threshold.

3. Numerical experiments

To numerically test the effectiveness and efficiency of the LSQR methods, we give two numerical examples. The calculations were carried out in MATLAB [1].

First we consider the Sylvester tensor equation
\[ A M X N B + C M Y N D = E, \]
with
\[ A = -10 \times \text{tenrand}([3 2 4 3]) \in \mathbb{R}^{3 \times 2 \times 4 \times 3}, \quad B = 20 \times \text{tenrand}([3 4 5 2]) \in \mathbb{R}^{3 \times 4 \times 5 \times 2}, \]
\[ C = -30 \times \text{tenrand}([3 2 5 2]) \in \mathbb{R}^{3 \times 2 \times 5 \times 2}, \quad D = 40 \times \text{tenrand}([2 5 5 2]) \in \mathbb{R}^{2 \times 5 \times 5 \times 2}, \]
and
\[ E = \text{tenrand}([3 2 5 2]) \in \mathbb{R}^{3 \times 2 \times 5 \times 2}. \]
The results obtained for this example are depicted in Figure 1 where

\[ r(i) = \sqrt{\log \| E - A X(i) * N B - C Y(i) * N D \|^2}. \]

As the second example, we study the coupled tensor equations (1) with

\[ A_1 = 22 \times \text{tenrand}(3 2 4 3) \in \mathbb{R}^{3 \times 2 \times 4 \times 3}, \quad B_1 = 12 \times \text{tenrand}(3 4 5 2) \in \mathbb{R}^{3 \times 4 \times 5 \times 2}, \]
\[ C_1 = 45 \times \text{tenrand}(3 2 5 2) \in \mathbb{R}^{3 \times 2 \times 5 \times 2}, \quad D_1 = 55 \times \text{tenrand}(2 5 5 2) \in \mathbb{R}^{2 \times 5 \times 5 \times 2}, \]
\[ A_2 = 66 \times \text{tenrand}(3 2 4 3) \in \mathbb{R}^{3 \times 2 \times 4 \times 3}, \quad B_2 = 11 \times \text{tenrand}(3 4 5 2) \in \mathbb{R}^{3 \times 4 \times 5 \times 2}, \]
\[ C_2 = 33 \times \text{tenrand}(3 2 5 2) \in \mathbb{R}^{3 \times 2 \times 5 \times 2}, \quad D_2 = 40 \times \text{tenrand}(2 5 5 2) \in \mathbb{R}^{2 \times 5 \times 5 \times 2}, \]

and

\[ E_1 = \text{tenrand}(3 2 5 2) \in \mathbb{R}^{3 \times 2 \times 5 \times 2}, \quad E_2 = \text{tenrand}(3 2 5 2) \in \mathbb{R}^{3 \times 2 \times 5 \times 2}. \]

The results of numerical results are illustrated in Figure 2 where

\[ r(i) = \log \sum_{k=1}^{2} \| E_k - A_k X(i) * N B_k - C_k Y(i) * N D_k \|^2. \]

The numerical experiments confirmed the accuracy and efficiency of Algorithms 1 and 2.

4. Conclusions

In this paper, we developed two tensor types of the LSQR method for computing the solutions of the coupled tensor equations (1). The resulting LSQR algorithms were easy to implement and simple to use. Finally, we presented numerical results which illustrate the efficiency of the LSQR methods.

Acknowledgments

The author wishes to express his gratitude to the editor and an anonymous reviewer for helpful remarks and suggestions.
Figure 2: Plot of the residual of Algorithms 1 and 2.

References


