Advances in Applied Mathematics 应用数学进展, 2025, 14(4), 1007-1018
Published Online April 2025 in Hans. https://www.hanspub.org/journal/aam
https://doi.org/10.12677/aam.2025.144223

Hans汉斯

求解大规模线性问题的张量GMRES算法

王仕伟*，杨　志，冷震北

重庆对外经贸学院数学与计算机学院，重庆

收稿日期：2025年3月28日；录用日期：2025年4月23日；发布日期：2025年4月30日

摘　要

彩色图像和视频通常可以被描述为高阶张量。本文基于三阶张量t-积，讨论了Krylov子空间方法用以解决图像恢复中的大规模线性问题。本文通过张量GMRES算法构建Krylov子空间，将大规模线性问题转换为小规模问题，且构建的子空间始终保持张量的空间结构。数值例子和彩色图像修复的应用说明了算法的有效性。

关键词

大规模线性问题，Krylov子空间方法，t-积，GMRES

Tensor GMRES Algorithm for Solving Large-Scale Linear Problems

Shiwei Wang*, Zhi Yang, Zhenbei Leng

School of Mathematics and Computer Science, Chongqing College of International Business and Economics, Chongqing

Received: Mar. 28th, 2025; accepted: Apr. 23rd, 2025; published: Apr. 30th, 2025

Abstract

Color images and video sequences can typically be characterized as higher-order tensors. This paper investigates Krylov subspace methods based on the third-order tensor t-product for solving large-scale linear systems arising in image restoration. This paper employs the tensor GMRES algorithm to construct the Krylov subspace, effectively reducing large-scale linear problems to manageable small-scale formulations, while consistently preserving the spatial architecture of tensors within the constructed subspace. Numerical experiments and applications in color image inpainting demonstrate the efficacy of the proposed methodology.

*通讯作者。

文章引用: 王仕伟, 杨志, 冷震北. 求解大规模线性问题的张量 GMRES 算法[J]. 应用数学进展, 2025, 14(4): 1007-1018.
DOI: 10.12677/aam.2025.144223

Keywords

Large-Scale Linear Problems, Krylov Subspace Methods, t-Product, GMRES

1. 引言

随着人工智能和大数据技术的飞速发展，数据规模和复杂性呈指数级增长，传统的向量和矩阵形式的数据表示已无法有效捕获高维数据的内在结构。在此背景下，张量作为高维数据的自然表达形式，广泛应用于推荐系统、图像处理、自然语言处理、生物信息学等领域。然而，大规模张量数据的处理和分析面临计算复杂性高、存储成本昂贵等挑战，这对高效算法的设计提出了更高要求。

在高阶张量分析的研究中，张量积是构建代数运算和分解模型的核心工具。传统的张量积形式包括模积(tensor mode product) [1]-[3]、哈达玛积(Hadamard product) [4] [5]和爱因斯坦积(Einstein product) [6] [7]等。虽然这些张量积在特定场景中具有一定优势，但在处理多维数据的代数一致性和计算效率方面存在局限性。2011 年，Kilmer 等人[8]首次提出基于三阶张量间的 t-积(t-product)，作为一种基于离散傅里叶变换(DFT)的创新张量积定义，近年来引起了广泛关注。t-积在多维数据处理和张量代数中具有独特优势[9]。首先，t-积能够利用离散傅里叶变换将卷积操作简化为逐元素乘法，大幅降低计算复杂度，同时保留张量的多维结构。张量 t-积的突出优势之一在于它支持一系列线性代数操作，例如 t-SVD、t-张量逆等，为张量分解和低秩逼近提供了强有力的工具。张量在不同应用广泛的领域，特别是彩色图像、视频恢复或压缩[1] [2] [5] [6] [10]-[17]。张量在现代科学中的其他应用，如信号处理[7] [18]，数据挖掘[8]，张量互补问题，计算机视觉，更多细节见[9]。最近的张量方法被用于数值求解[19]中的偏微分方程。

在图像处理中，随着图像数据维度的增加，尤其是在处理彩色图像、视频以及其他多维数据时，传统的线性模型往往难以准确捕捉复杂的图像结构。当图像数据不完整、受损或被噪声污染时，求解过程可能无法获得稳定和精确的结果，这时张量求解的离散化过程便可能面临不适定问题。本文主要研究彩色图像与视频恢复等图像处理中的大规模线性离散问题[1] [10] [11] [20]-[22]，可将其表述为

$$\mathcal{A} *_L \vec{\mathcal{X}} = \vec{\mathcal{B}}, \mathcal{A} = [a]_{i,j,k=1}^{n_1 \times n_2 \times n_3} \in \mathbb{R}^{n_1 \times n_2 \times n_3}, \vec{\mathcal{B}} \in \mathbb{R}^{n_2 \times 1 \times n_3}. \tag{1.1}$$

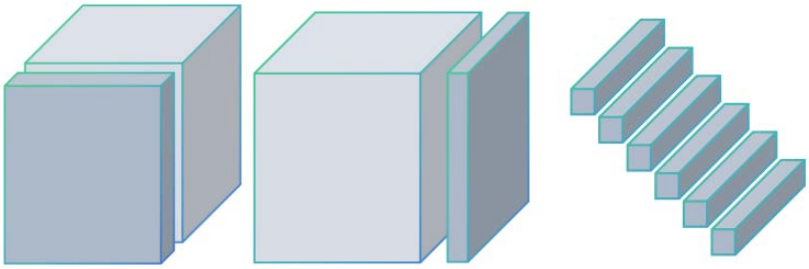
或张量最小二乘问题

$$\min_{\vec{\mathcal{X}} \in \mathbb{R}^{n_1 \times n_4 \times n_3}} \left\| \mathcal{A} *_L \mathcal{X} - \mathcal{B} \right\|_F^2, \tag{1.2}$$

其中 $\mathcal{A} = [a]_{i,j,k=1}^{n_1,n_2,n_3} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ 和 $\mathcal{B} \in \mathbb{R}^{n_1 \times n_4 \times n_3}$ 为已知三阶张量，$*_L$ 为张量间可逆线性变换 L 的乘积[23]，$\|\cdot\|_F$ 表示张量的 Frobenius 范数。若 L 为快速傅里叶变换(FFT)时，$*_L$ 为 t-积。

2. 预备知识

对于三阶张量 $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$，可以通过不同的分割方式得到其各类切片，本文使用 $\mathcal{A}_{(:,:,k)}$ 或 A_k 表示张量 $\mathcal{A}$ 的第 k 个正面切片，$\mathcal{A}_{(:,j,:)}$ 或 $\vec{\mathcal{A}}_j$ 表示张量 $\mathcal{A}$ 的第 j 个侧面切片，$\mathcal{A}_{(i,j,:)}$ 表示为张量管，这几种张量结构如图 1 所示：



(a) (b) (c)

Figure 1. (a) frontal slices $\mathcal{A}_{(:,:,k)}$, (b) lateral slices $\mathcal{A}_{(:,j,:)}$, (c) tube fibers $\mathcal{A}_{(i,j,:)}$

图 1. (a) 正面切片 $\mathcal{A}_{(:,:,k)}$，(b) 横向切片 $\mathcal{A}_{(:,j,:)}$，(c) 管 $\mathcal{A}_{(i,j,:)}$

在本节预备一些 t-积的定义和性质，是基于离散傅里叶变换(DFT)所定义的，给定向量 $v\in\mathbb{C}^n$，假设 F_n 是 $n\times n$ DFT 矩阵，那么

$$\overline{v}=F_n v\in\mathbb{C}^n$$

其中 F_n 的元素是复数形式，其分量被定义为

$$\left(F_n\right)_{ij}=\omega^{(i-1)(j-1)},i,j=1,\cdots,n.$$

这里 $\omega=\mathrm{e}^{-2\pi\mathrm{i}/n},\mathrm{i}^2=-1$。

给定 $\mathcal{A}\in\mathbb{R}^{n_1\times n_2\times n_3}$ 是一个三阶张量，使用三个算子 bcirc、unfold 和 fold，即

$$\mathrm{bcirc}\left(\mathcal{A}\right)=\begin{bmatrix}A_1 & A_{n_3} & A_{n_3-1} & \cdots & A_2\\ A_2 & A_1 & A_{n_3} & \cdots & A_3\\ \vdots & \vdots & \ddots & \ddots & \vdots\\ A_{n_3} & A_{n_3-1} & \cdots & A_2 & A_1\end{bmatrix}\in\mathbb{R}^{n_1n_3\times n_2n_3},$$

$$\mathrm{unfold}\left(\mathcal{A}\right)=\begin{bmatrix}A_1\\ A_2\\ \vdots\\ A_{n_3}\end{bmatrix}\in\mathbb{R}^{n_1n_3\times n_2},\mathrm{fold}\left(\mathrm{unfold}\left(\mathcal{A}\right)\right)=\mathcal{A}.$$

定义 2-1 张量 $\mathcal{A}\in\mathbb{R}^{n_1\times n_2\times n_3}$ 和 $\mathcal{B}\in\mathbb{R}^{n_2\times n_4\times n_3}$ 之间的 t-积被定义为

$$\mathcal{C}=\mathcal{A}*\mathcal{B}=\mathrm{fold}\left(\mathrm{bcirc}\left(\mathcal{A}\right)\mathrm{unfold}\left(\mathcal{B}\right)\right)\in\mathbb{R}^{n_1\times n_4\times n_3}.$$

对于三阶张量 $\mathcal{B}\in\mathbb{R}^{n_2\times n_4\times n_3}$，其第 j 个张量侧面切片为 $\vec{\mathcal{B}}_j\in\mathbb{R}^{n_2\times 1\times n_3}$，也称侧面切片为张量列，则有

$$\mathcal{B}=\left[\vec{\mathcal{B}}_1,\vec{\mathcal{B}}_2,\cdots,\vec{\mathcal{B}}_{n_4}\right].$$

定义 2-2 由三阶张量 $\mathcal{A}\in\mathbb{R}^{n_1\times n_2\times n_3}$ 和张量列 $\vec{\mathcal{B}}\in\mathbb{R}^{n_2\times 1\times n_3}$ 生成的 k 维 tKrylov 张量，定义如下

$$\mathcal{K}_k\left(\mathcal{A},\vec{\mathcal{B}}\right)=\left[\vec{\mathcal{B}},\mathcal{A}\vec{\mathcal{B}},\mathcal{A}^2\vec{\mathcal{B}},\cdots,\mathcal{A}^{k-1}\vec{\mathcal{B}}\right].$$

设 $\hat{\mathcal{A}}$ 是在张量 $\mathcal{A}\in\mathbb{R}^{n_1\times n_2\times n_3}$ 的所有 3-模管上应用 DFT 得到的张量。用 Matlab 命令 fft 得到 $\hat{\mathcal{A}}=\mathrm{fft}\left(\mathcal{A},[\,],3\right)$，并运用快速傅里叶反变换 ifft 得到 $\mathcal{A}=\mathrm{ifft}\left(\hat{\mathcal{A}},[\,],3\right)$。得益于张量 t-积的特殊循环结构，可将两个张量的 t-积投影到傅里叶变换域中展开计算，即对于两个尺寸合适的张量 $\mathcal{A}\in\mathbb{R}^{n_1\times n_2\times n_3}$ 和 $\mathcal{B}\in\mathbb{R}^{n_2\times n_4\times n_3}$ 可在 MATLAB 中快速计算，如算法 2-1 所示。

算法 2-1. 基于 fft 的 t-积

输入：$\mathcal{A}\in\mathbb{R}^{n_1\times n_2\times n_3}$ 和 $\mathcal{B}\in\mathbb{R}^{n_2\times n_4\times n_3}$

输出：$\mathcal{C}=\mathcal{A}*\mathcal{B}\in\mathbb{R}^{n_1\times n_4\times n_3}$

$\hat{\mathcal{A}}=\text{fft}\left(\mathcal{A},[\,],3\right)$；$\hat{\mathcal{B}}=\text{fft}\left(\mathcal{B},[\,],3\right)$

For $k=1,2,\cdots,n_3$

$A_k=\hat{\mathcal{A}}\left(:,:,k\right);B_k=\hat{\mathcal{B}}\left(:,:,k\right)$

$\hat{\mathcal{C}}\left(:,:,k\right)=A_kB_k$

End

$\mathcal{C}=\text{ifft}\left(\hat{\mathcal{C}},[\,],3\right)$

定义 2-3 若尺寸为 $n_1\times n_1\times n_3$ 的张量 $\mathcal{I}$ 的第 1 个正面切片是一个单位矩阵，其他索引位置元素都为零，则称张量 $\mathcal{I}$ 为 t-积下的单位张量。

特别注意，管标量 $\vec{e}_1$ 表示 $1\times1\times n_3$ 的单位张量，其形式为一个管状向量，除索引(1, 1, 1)的元素为 1 外，其他位置都为 0。

定义 2-4 若 $\mathcal{Q}\in\mathbb{R}^{m\times m\times n}$ 为正交张量，则满足 $\mathcal{Q}^{\mathrm{T}}\mathcal{Q}=\mathcal{Q}\mathcal{Q}^{\mathrm{T}}=\mathcal{I}$。

若 $\mathcal{Q}$ 是一个正交张量，则对任意满足尺寸的张量 $\mathcal{A}$ 都有 $\|\mathcal{Q}*\mathcal{A}\|_F=\|\mathcal{A}\|_F$ 或 $\|\mathcal{A}*\mathcal{Q}\|_F=\|\mathcal{A}\|_F$。

定义 2-5 若张量 $\mathcal{A}\in\mathbb{R}^{n_1\times n_1\times n_3}$ 可逆，则张量 $\mathcal{A}$ 的逆记作 $\mathcal{A}^{-1}$，满足

$$\mathcal{A}*\mathcal{A}=\mathcal{A}^{-1}*\mathcal{A}=\mathcal{I}.$$

对于非零张量 $\vec{\mathcal{X}}\in\mathbb{R}^{n_1\times1\times n_3}$，可以将其分解为一个归一化张量 $\vec{\mathcal{D}}\in\mathbb{R}^{n_1\times1\times n_3}$ 与管标量 $\boldsymbol{d}\in\mathbb{R}^{1\times1\times n_3}$ 的 t-积，如下

$$\vec{\mathcal{X}}=\vec{\mathcal{D}}*\boldsymbol{d},$$

算法 2-2 为张量列的归一化分解过程。

算法 2-2. 归一化(Normalization)

输入：非零张量 $\vec{\mathcal{X}}\in\mathbb{R}^{m\times1\times n}$, tol

输出：$\vec{\mathcal{D}}$，$\boldsymbol{d}$ ($\vec{\mathcal{X}}=\vec{\mathcal{D}}*\boldsymbol{d}$), 且 $\left\|\vec{\mathcal{D}}\right\|_F=1$.

1. 计算 $\vec{\mathcal{D}}=\text{fft}\left(\vec{\mathcal{X}},[\,],3\right)$
2. 对于 $j=1,2,\cdots,n$
3. 计算 $\boldsymbol{d}_j\leftarrow\left\|\mathcal{D}_j\right\|_2$
4. 如果 $\boldsymbol{d}_j>\text{tol}$ 则

计算 $\vec{\mathcal{D}}_j\leftarrow\frac{1}{\boldsymbol{d}_j}\vec{\mathcal{D}}_j$

5. 否则

计算 $\vec{\mathcal{D}}_j\leftarrow\textbf{randn}\left(m,1\right),\boldsymbol{d}_j\leftarrow\left\|\mathcal{D}_j\right\|_2$

6. 计算 $\vec{\mathcal{D}}=\text{fft}\left(\vec{\mathcal{X}},[\,],3\right)$

张量 QR (tQR)的分解由 Kilmer 等人[9]描述。$\mathcal{A}\in\mathbb{R}^{l\times m\times n}$ 的 QR 分解表示为

$$\mathcal{A}=\mathcal{Q}*\mathcal{R},$$

其中张量 $\mathcal{Q}\in\mathbb{R}^{l\times m\times n}$ 是部分正交的，$\mathcal{R}\in\mathbb{R}^{m\times m\times n}$ 的每个正面切片是一个上三角形。算法 2-3 的 tQR 分解是

在傅里叶域实现 tQR 分解。

算法 2-3. tQR 分解[1]

输入：$\mathcal{A}\in\mathbb{R}^{l\times m\times n}$

输出：$\mathcal{Q}\in\mathbb{R}^{l\times m\times n},\mathcal{R}\in\mathbb{R}^{m\times m\times n},\mathcal{A}=\mathcal{Q}*\mathcal{R}$

$\hat{\mathcal{A}}=\text{fft}\left(\mathcal{A},[\,],3\right)$

For $j=1,2,\cdots,n$

$\hat{\mathcal{A}}(:,:,k)=QR$

$\hat{\mathcal{Q}}(:,:,k)=Q,\hat{\mathcal{R}}(:,:,k)=R$

End

$\mathcal{Q}=\text{ifft}\left(\hat{\mathcal{Q}},[\,],3\right)$，$\mathcal{R}=\text{ifft}\left(\hat{\mathcal{R}},[\,],3\right)$.

3. 张量全局-GMRES 算法

首先介绍了 El Guide 等人[20]使用的其他定义，设三阶张量 $\mathcal{B}\in\mathbb{R}^{n_2\times n_4\times n_3}$ 与向量 $y=\left[y_1,y_2,\cdots,y_k\right]^{\mathrm{T}}\in\mathbb{R}^k$，El Guide 等人将定义了一个积 $\circledast$ 为

$$\mathcal{B}\circledast y=\sum_{j=1}^{k}y_j\vec{\mathcal{B}}_j$$

另外，若使三阶张量 $\mathbb{B}_k$ 表示为

$$\mathbb{B}_k=\left[\mathcal{B}_1,\mathcal{B}_2,\cdots,\mathcal{B}_k\right]\in\mathbb{R}^{n_2\times n_4k\times n_3},\mathcal{B}_j\in\mathbb{R}^{n2\times n_4\times n_3},j=1,\cdots,k.$$

则

$$\mathbb{B}_k\circledast y=\sum_{j=1}^{k}y_j\mathcal{B}_j.$$

定义 3-1 设 $z\in\mathbb{R}^{1\times1\times n_3}$，那么 z 的管秩是其非零傅里叶系数的个数。如果 z 的秩等于 n_3，它是可逆的，用 z^{-1} 表示 z 的逆：$z*z^{-1}=z^{-1}*z=e$，其中 e 是单位张量管。

考虑张量线性方程组

$$\mathcal{A}*\mathcal{X}=\mathcal{B}\tag{3.1}$$

这里 $\mathcal{A}\in\mathbb{R}^{n_1\times n_1\times n_3},\mathcal{X}\in\mathbb{R}^{n_1\times n_4\times n_3}$ 和 $\mathcal{B}\in\mathbb{R}^{n_1\times n_4\times n_3}$。由张量 $\mathcal{A}$ 和 $\mathcal{V}\in\mathbb{R}^{n_1\times n_4\times n_3}$ 生成的 k 维 Krylov 子空间表示为

$$\mathcal{K}_k\left(\mathcal{A},\mathcal{V}\right)=\left[\mathcal{V},\mathcal{A}*\mathcal{V},\mathcal{A}^2*\mathcal{V},\cdots,\mathcal{A}^{k-1}*\mathcal{V}\right].$$

可得到出张量全局-Arnoldi 算法，如算法 3-1 所示。

算法 3-1. 张量全局-Arnoldi 算法

输入：$\mathcal{A}\in\mathbb{R}^{n_1\times n_1\times n_3},\mathcal{V}\in\mathbb{R}^{n_1\times n_4\times n_3}$

输出：k 维 Krylov 子空间 $\mathcal{K}_k\left(\mathcal{A},\mathcal{V}\right)$

$a=\left\|\mathcal{V}\right\|_F$；$\mathcal{V}_1=\dfrac{\mathcal{V}}{a}$

For $j=1,2,\cdots,k$

$\mathcal{J}=\mathcal{A}*\mathcal{V}_j$

For $i=1,2,\cdots,j$

$R_{i,j}=\left\langle\mathcal{V}_i,\mathcal{J}\right\rangle;\mathcal{J}=\mathcal{J}-R_{i,j}*\mathcal{V}_i$

End
$R_{j+1,j}=\|\mathcal{J}\|_F$
If $R_{j+1,j}=0$
Break
EndIf
$\mathcal{V}_{j+1}=\mathcal{J}/R_{j+1,j}$.
End

由算法 3-1 得到的张量 $\mathbb{V}_k$ 是 $n_1\times n_4k\times n_3$ 张量，且

$$\mathbb{V}_k=[\mathcal{V}_1,\mathcal{V}_2,\cdots,\mathcal{V}_k],$$

$\bar{R}_k$ 是 R 的元素所组成的 $(k+1)\times k$ 的上海森伯格矩阵(Hesenberg)， R_k 是由 $\bar{R}_k$ 删去最后一行所得到的矩阵。将 $R_{:,j}$ 表示矩阵 R_k 的第 j 列，则

$$\mathbb{V}_k\circledast R_k=\left[\mathbb{V}_k\circledast R_{:,1},\mathbb{V}_k\circledast R_{:,2},\cdots,\mathbb{V}_k\circledast R_{:,k}\right].$$

命题 3-1 设 $\mathbb{V}_k$ 是由 $[\mathcal{V}_1,\mathcal{V}_2,\cdots,\mathcal{V}_k]$ 定义的张量，其中 $\mathcal{V}\in\mathbb{R}^{n_1\times n_4\times n_3}$ 由张量全局-Arnoldi 算法定义，那么对任一 $y=[y_1,y_2,\cdots,y_k]^{\mathrm{T}}\in\mathbb{R}^k$ 都有

$$\|\mathbb{B}_k\circledast y\|_F=\|y\|_F.$$

设 $\mathcal{X}_0\in\mathbb{R}^{n_1\times n_4\times n_3}$ 是一个任意的初始猜测，则相应的剩余残差张量 $\mathcal{R}_0=\mathcal{B}-\mathcal{A}*\mathcal{X}_0$。若经过 m 次迭代后得到的近似解 $\mathcal{X}_m$ ，则相应地将问题(3.1)转换为寻找最小化问题的最优解：

$$\|\mathcal{R}_m\|_F=\min_{\mathcal{X}\in\mathcal{X}_0+\mathcal{K}_m(\mathcal{A},\mathcal{R}_0)}\left\{\|\mathcal{B}-\mathcal{A}*\mathcal{X}\|_F\right\}.$$

设 $\mathcal{X}_m=\mathcal{X}_0+\mathbb{V}_m\circledast y, y\in\mathbb{R}^m$ ，然后

$$\mathcal{R}_m=\mathcal{B}-\mathcal{A}*\mathcal{X}_m=\mathcal{B}-\mathcal{A}*(\mathcal{X}_0+\mathbb{V}_m\circledast y)=\mathcal{R}_0-(\mathcal{A}*\mathbb{V}_m)\circledast y.$$

则得到问题

$$\|\mathcal{R}_m\|_F=\min_{y\in\mathbb{R}^m}\left\{\|\mathcal{R}_0-(\mathcal{A}*\mathbb{V}_m)\circledast y\|_F\right\}. \tag{3.2}$$

事实上，由于 $\mathcal{R}_0=\|\mathcal{R}_0\|_F\mathcal{V}_1$ 和 $\mathcal{V}_1=\mathcal{V}_{m+1}\circledast e_1$， e_1 是 $\mathbb{R}^{m+1}$ 中的第一个标准基向量，利用命题 3-1，得到

$$\begin{aligned}
&\|\mathcal{R}_0-(\mathcal{A}\mathbb{V}_m)\circledast y\|_F\\
&=\|\mathcal{R}_0-(\mathbb{V}_{m+1}\circledast\bar{R}_m)\circledast y\|_F\\
&=\|\|\mathcal{R}_0\|_F(\mathbb{V}_{m+1}\circledast e_1)-(\mathbb{V}_{m+1}\circledast\bar{R}_m)\circledast y\|_F\\
&=\|\mathbb{V}_{m+1}\circledast(\|\mathcal{R}_0\|_F e_1-\bar{R}_m y)\|_F\\
&=\|\|\mathcal{R}_0\|_F e_1-\bar{R}_m y\|_2.
\end{aligned}$$

最后

$$\mathcal{X}_m=\mathcal{X}_0+\mathbb{V}_m\circledast y$$

其中

$$y=\arg\min_{y\in\mathbb{R}^m}\|\|\mathcal{R}_0\|_F e_1-\bar{R}_m y\|_2.$$

综上得到了将大规模问题(3.1)转换为小规模问题张量全局-GMRES 算法(算法 3-2)。

算法 3-2. 张量全局-GMRES 算法(G-GMRES)

输入： $\mathcal{A}\in\mathbb{R}^{n_1\times n_1\times n_3},\mathcal{X}_0,\mathcal{B}\in\mathbb{R}^{n_1\times n_4\times n_3}$, itermax, tol, 子空间维数 m.

输出： 问题(3.1)近似解 $\mathcal{X}_m$.

For $k=1,2,\cdots$ until itermax

计算 $\mathcal{R}_0=\mathcal{B}-\mathcal{A}*\mathcal{X}_0$

使用张量全局-Arnoldi 算法用计算 $\mathbb{V}_k$ 和 $\bar{R}_k$.

计算 $y=\arg\min_{y\in\mathbb{R}^m}\left\|\|\mathcal{R}_0\|_F e_1-\bar{R}_m y\right\|_2$.

$\mathcal{X}_m=\mathcal{X}_0+\mathbb{V}_m\circledast y$

$\mathcal{R}_m=\mathcal{B}-\mathcal{A}*\mathcal{X}_m$

If $\|\mathcal{R}_0\|_F\le$ tol

Break

End If

$\mathcal{X}_0=\mathcal{X}_m$

End

4. 张量 Krylov 子空间算法

对问题(1.1)的直接求解所花费的代价会随着系统的维度增加而呈现几何倍的增长，可能会导致程序崩溃，因此将问题(1.1)转换为小规模问题是可能是一种有效的方式。本文发展了形式问题(1.1)的最小二乘问题的 t-积 Arnoldi (t-Arnoldi)和 t-积 GMRES 方法。该方法将用于说明相对于矢量化或矩阵化张量方程的一般情况下，张量化的潜在优势。

对于问题(3.1)中 $\mathcal{B}\in\mathbb{R}^{n_1\times n_4\times n_3}$ ，可视为 n_4 个独立的子问题，即

$$\vec{\mathcal{X}}_j=\arg\min_{\vec{\mathcal{X}}\in\mathbb{R}^{n_1\times n_4\times n_3}}\left\|\mathcal{A}*\vec{\mathcal{X}}-\vec{\mathcal{B}}_j\right\|_F^2,\ j=1,2,\cdots,n_4.$$

令 $\vec{\mathcal{X}}=\mathcal{V}_k*\vec{\mathcal{Y}}\left(k\ll n\right)$，则有

$$\arg\min_{\vec{\mathcal{Y}}\in\mathbb{R}^{k\times 1\times n}}\left\|\mathcal{A}*\mathcal{V}_k*\vec{\mathcal{Y}}-\vec{\mathcal{B}}\right\|_F^2. \tag{4.1}$$

其中 $\mathcal{V}_k$ 中的张量列两两相互正交，且 $\mathcal{V}_k^{\mathrm{T}}*\mathcal{V}_k=\mathcal{I}$ 。问题(4.1)的正规方程为

$$\mathcal{V}_k^{\mathrm{T}}*\left(\mathcal{A}^{\mathrm{T}}*\mathcal{A}\right)*\mathcal{V}_k*\vec{\mathcal{Y}}_\mu=\mathcal{V}_k^{\mathrm{T}}*\mathcal{A}^{\mathrm{T}}*\vec{\mathcal{B}}. \tag{4.2}$$

则可以得到问题(4.1)的解为

$$\vec{\mathcal{Y}}_\mu=\left(\mathcal{V}_k^{\mathrm{T}}*\left(\mathcal{A}^{\mathrm{T}}*\mathcal{A}\right)*\mathcal{V}_k\right)^{-1}*\mathcal{V}_k^{\mathrm{T}}*\mathcal{A}^{\mathrm{T}}*\vec{\mathcal{B}}. \tag{4.3}$$

4.1. 张量 Arnoldi 算法

首先介绍一个带 t-Arnoldi 过程的算法。将大规模问题(3.1)简化为小规模问题。算法 4-1 所描述的 t-Arnoldi 过程将张量 $\mathcal{A}\in\mathbb{R}^{n_1\times n_1\times n_3}$ 简化为一个上海森伯格张量(tHessenberg)，其每个正面切片都是一个上海森伯格矩阵。

算法 4-1. t-Arnoldi 分解算法

输入： $\mathcal{A}\in\mathbb{R}^{n_1\times n_1\times n_3},\vec{\mathcal{B}}\in\mathbb{R}^{n_1\times 1\times n_3}$

输出：k 维 Krylov 子空间 $\mathcal{K}_k\left(\mathcal{A},\mathcal{V}\right)$

$\left[\vec{\mathcal{Q}}_1,z_1\right]=\text{Normalization}\left(\vec{\mathcal{B}}\right)$

For $j=1,2,\cdots,k$

$\vec{\mathcal{W}}=\mathcal{A}*\vec{\mathcal{Q}}_j$

For $i=1,2,\cdots,j$

$h_{i,j}=\mathcal{Q}_j^{\mathrm{T}}*\vec{\mathcal{W}};\vec{\mathcal{W}}=\vec{\mathcal{W}}-\mathcal{Q}_j^{\mathrm{T}}*\vec{\mathcal{W}}$

End

$\left[\vec{\mathcal{Q}}_{j+1},h_{j+1,j}\right]=\mathrm{Normalization}\left(\vec{\mathcal{W}}\right)$

End

假设 t-Arnoldi 过程的步数小以避免分解，当 k 选择足够小以使得对任何次对角管标量 $h_{j+1,j}$ 都不可逆，则称算法 4-1 为 t-Arnoldi 分解算法。

$$\mathcal{A}*\mathcal{Q}_k=\mathcal{Q}_{k+1}*\bar{\mathcal{H}}_k$$

其中

$$\bar{\mathcal{H}}_k=\begin{bmatrix} h_{11} & & & & h_{1,k} \\ h_{21} & h_{22} & & & \\ & h_{32} & h_{33} & & \\ & & \ddots & \ddots & \\ & & & h_{k,k-1} & h_{k,k} \\ & & & & h_{k+1,k} \end{bmatrix}\in\mathbb{R}^{(k+1)\times k\times n_3}.$$

而 $\mathcal{Q}_k\in\mathbb{R}^{n_1\times k\times n_3}$ 形成 t-Krylov 子空间的标准正交张量基。

4.2. 张量 GMRES 算法

在 t-Arnoldi 分解算法中，生成了张量列正交的张量 $\mathcal{Q}_k$，若将 $\mathcal{Q}_k$ 的列作为子空间的基，则有

$$\vec{\mathcal{Y}}=\arg\min_{\vec{\mathcal{Y}}\in\mathbb{R}^{k\times 1\times n}}\left\|\mathcal{A}*\mathcal{Q}_k*\vec{\mathcal{Y}}-\vec{\mathcal{B}}\right\|_F^2$$

且 $\mathcal{A}*\mathcal{Q}_k=\mathcal{Q}_{k+1}*\bar{\mathcal{H}}_k$，则

$$\vec{\mathcal{Y}}=\arg\min_{\vec{\mathcal{Y}}\in\mathbb{R}^{k\times 1\times n}}\left\|\mathcal{Q}_{k+1}\bar{\mathcal{H}}_k*\vec{\mathcal{Y}}-\vec{\mathcal{B}}\right\|_F^2.$$

由于 $\vec{\mathcal{B}}=\vec{\mathcal{Q}}_1*z_1$，可以得到

$$\mathcal{Q}_{k+1}^{\mathrm{T}}*\vec{\mathcal{B}}=\vec{e}_1*z_1\in\mathbb{R}^{(k+1)\times 1\times n_3}.$$

其中 $\vec{e}_1$ 是一个管标量，除其索引位置(1, 1, 1)元素为 1 外的其他位置元素都为 0。又因为 $\mathcal{Q}_k$ 张量列正交，那么可以将求解问题(4.1)转化为求解

$$\vec{\mathcal{Y}}=\arg\min_{\vec{\mathcal{Y}}\in\mathbb{R}^{k\times 1\times n}}\left\|\bar{\mathcal{H}}_k*\vec{\mathcal{Y}}-\vec{e}_1*z_1\right\|_F^2.$$

将张量 GMRES 算法的过程在算法 4-2 中进行描述。

算法 4-2. 张量 GMRES 算法

输入： $\mathcal{A}\in\mathbb{R}^{n_1\times n_1\times n_3},\mathcal{B}\in\mathbb{R}^{n_1\times n_4\times n_3}$, itermax, tol

输出： 问题(3.1)近似解 $\mathcal{X}$.

$\left\|\mathcal{R}_0\right\|_F\le\mathrm{tol}$

For $j=1,\cdots,n_4$

计算 $\left[\vec{\mathcal{Q}}_1,z_1\right]=\text{Normalization}\left(\vec{\mathcal{B}}_j\right)$

$\left\|\mathcal{R}_0\right\|_F\le\left\|z_1\right\|_F$

For $k=1,2,\cdots$ until 收敛

使用张量 Arnoldi 算法用计算 $\mathcal{Q}_k,\mathcal{Q}_{k+1}$ 和 $\bar{\mathcal{H}}_k$.

计算 $\vec{\mathcal{Y}}=\underset{\vec{\mathcal{Y}}\in\mathbb{R}^{k\times1\times n}}{\arg\min}\left\|\bar{\mathcal{H}}_k*\vec{\mathcal{Y}}-\vec{e}_1z_1\right\|_F^2$.

$\vec{\mathcal{X}}_j=\mathcal{Q}*\vec{\mathcal{Y}}\left(k\ll n\right)$

$\mathcal{R}_k=\vec{\mathcal{B}}_j-\mathcal{A}*\vec{\mathcal{X}}_j$

End For

$\mathcal{X}_{(:,j,:)}=\vec{\mathcal{X}}_j$

End For

5. 数值实例

本节的数值实例的实验部分的环境配置均如表 1 所述。

Table 1. System configuration specifications
表 1. 环境配置

环境	配置
CPU	Inter(R) Core(H) i7-11800H @2.30GHz
GPU	NVIDIA GeForce RTX 3060
操作系统	WINDOWS10
内存	16 GB
MATLAB	2018a

例子 5.1. 数值恢复

本例比较张量全局-GMRES 算法和张量 GMRES 算法两种算法在对称正定张量系统中的计算性能，在 MATLAB 中 Hansen [24]的正则化工具中的函数 baart 生成矩阵

$$A_1=\text{baart}\left(400\right),A_2=\text{gallery}\left(\text{prolate}',400,\alpha\right).$$

设 $\alpha=0.45$，张量 $\mathcal{A}$ 满足

$$\mathcal{A}_{(:,:,i)}=A_1\left(i,1\right)A_2,i=1,2,\cdots,400.$$

$\mathcal{A}$ 的正面切片的条件数都大于 10^{15}。因此张量 $\mathcal{A}$ 是高度离散化的，令真实数据 $\vec{\mathcal{X}}_{true}\in\mathbb{R}^{400\times1\times400}$ 的元素全为 1，并通过 $\mathcal{A}*\vec{\mathcal{X}}_{true}=\vec{\mathcal{B}}$ 生成数据张量。

经典的最小二乘问题为

$$\vec{x}^*:=\underset{\vec{x}\in\mathbb{R}^{mn\times1}}{\arg\min}\left\|\left(A\otimes A\right)\vec{x}-\vec{b}\right\|_F^2,\tag{5.1}$$

这里的 A 是模糊矩阵，而 $\vec{b}$ 是 $\vec{\mathcal{B}}$ 向量化后的结果，问题(5.1)是问题(1.2)矩阵化后的结果。

表 2 给出了 GMRES、G-GMRES 和 tGMRES 算法在数字恢复实验中得到的解的相对误差，所需的迭代次数(子空间维数 k)，以及计算时间。表中的数据显示的是 GMRES 算法求解经典最小化问题(5.1)，以及 G-GMRES 和 tGMRES 算法求解 t-积结构的问题(1.2)的数据恢复情况。在数字恢复实验中，令收敛

的条件是相邻两次迭代解的相对误差小于 1e−06，则 GMRES 算法得到解与真解的相对误差要高于 G-GMRES 和 tGMRES 算法，这是因为 G-GMRES 和 tGMRES 算法的 t-积结构保持了数据的空间结构性，而 G-GMRES 算法使用的时间多于 tGMRES 算法，且 tGMRES 在构建子空间时，构建的子空间维数更小，得到稳定解时的迭代解的相对误差更小。

Table 2. Performance comparison of G-GMRES and tGMRES algorithms in digital restoration experiment
表 2. G-GMRES 和 tGMRES 算法在数字恢复实验中的性能比较

算法	迭代数	相对误差	CPU 时间(s)
GMRES	213	8.71e−06	103.23
G-GMRES	65	2.03e−06	346.45
tGMRES	44	8.25e−07	236.32

例子 5.2. 彩色图像恢复

此例显示了模糊彩色图像辣椒在 G-GMRES 和 tGMRES 算法下的恢复情况。原始图像数据在 MATLAB 中存储为张量 $\mathcal{X}_{or} \in \mathbb{R}^{256\times256\times3}$，将其转换为张量列的形式存储，即 $\mathcal{X}_{true} \in \mathbb{R}^{256\times3\times256}$，其中 $\mathcal{X}_{true}$ 的第 j 个张量列的数据由 $\mathcal{X}_{or}$ 中的第 j 个正面切片的数据构成。对于模糊张量 $\mathcal{A} \in \mathbb{R}^{256\times256\times256}$，定义为

$$z = \left[exp\left(-\left([0:band-1]^2\right)\Big/\left(2\sigma^2\right)\right), zeros\left(1, N-band\right)\right],$$

$$A = toeplitz\left(z\right), \mathcal{A}_{(:,:,i)} = \frac{1}{2\pi\sigma} A\left(i,1\right) A, i = 1,\cdots,256.$$

令 $N = 256, \sigma = 2.5, \text{band} = 12$，其中张量 $\mathcal{A}$ 的前 12 个正面切片的条件数大于 10^7，其余切片的条件数皆为无限。

表 3 显示了在例子 5.2 的彩色图像恢复实验中，G-GMRES 和 tGMRES 算法的性能比较，包括相对误差和计算时间。在彩色图像恢复实验中，可以得到和例子 5.1 类似的结果，而 G-GMRES 算法涉及到扁平化，tGMRES 算法相较于 G-GMRES 算法始终保持 t-积结构，保持了彩色图片三通道间的可能存在的空间结构性，这使其得到解时的迭代解的相对误差更小。

Table 3. Performance comparison of G-GMRES and tGMRES algorithms in color image restoration experiment
表 3. G-GMRES 和 tGMRES 算法在彩色图像恢复实验中的性能比较

算法	相对误差	CPU 时间(s)
G-GMRES	6.32e−06	862.56
tGMRES	5.11e−07	635.24

原图　模糊图　G-GMRES 恢复图　tGMRES 恢复图

Figure 2. The restoration effect of color images
图 2. 彩色图像的恢复效果图

图 2 给出了彩色图像辣椒和被模糊的图像，以及使用 G-GMRES 和 tGMRES 算法得到的恢复图像。

6. 结论

传统的图像恢复方法通常需要将模糊图像进行矩阵化或者向量化，这可能破坏了图像各个通道间可能存在的空间结构性，从而导致恢复质量不佳，而全局构建子空间的方法会占用更多的存储空间，在计算时会将误差扩大，使得解的质量下降。张量 t-积结构会避免图像进行矩阵化或者向量化，且张量切片形式构建的子空间，占用的空间更少，计算时间更短。

基金项目

资助项目 1：KYZK2024010：图像处理中的张量离散不适定问题及其高性能迭代算法研究。
资助项目 2：KYZK2024027：变分不等式与不动点问题非单调步长的惯性投影算法。
资助项目 3：KYZK2024003：变分不等式与不动点问题在 NASH 群体博弈的应用研究。

参考文献

[1] Kolda, T.G. and Bader, B.W. (2009) Tensor Decompositions and Applications. *SIAM Review*, **51**, 455-500. https://doi.org/10.1137/07070111x

[2] Tucker, L.R. (1966) Some Mathematical Notes on Three-Mode Factor Analysis. *Psychometrika*, **31**, 279-311. https://doi.org/10.1007/bf02289464

[3] De Lathauwer, L., De Moor, B. and Vandewalle, J. (2000) A Multilinear Singular Value Decomposition. *SIAM Journal on Matrix Analysis and Applications*, **21**, 1253-1278. https://doi.org/10.1137/s0895479896305696

[4] Goodfellow, I., Bengio, Y. and Courville, A. (2016) Deep Learning. MIT Press.

[5] He, K., Zhang, X., Ren, S. and Sun, J. (2015) Delving Deep into Rectifiers: Surpassing Human-Level Performance on Imagenet Classification. 2015 *IEEE International Conference on Computer Vision* (*ICCV*), Santiago, 7-13 December 2015, 1026-1034. https://doi.org/10.1109/iccv.2015.123

[6] Rasmussen, C.E. and Williams, C.K.I. (2005) Gaussian Processes for Machine Learning. The MIT Press. https://doi.org/10.7551/mitpress/3206.001.0001

[7] Paszke, A., Gross, S., Massa, F., *et al*. (2019) PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Advances in Neural Information Processing Systems* (*NeurIPS*), Vancouver, 8-14 December 2019.

[8] Kilmer, M.E. and Martin, C.D. (2011) Factorization Strategies for Third-Order Tensors. *Linear Algebra and Its Applications*, **435**, 641-658. https://doi.org/10.1016/j.laa.2010.09.020

[9] Braman, K. (2010) Third-Order Tensors as Linear Operators on a Space of Matrices. *Linear Algebra and Its Applications*, **433**, 1241-1253. https://doi.org/10.1016/j.laa.2010.05.025

[10] Cichocki, A. and Amari, S.I. (2010) Tensor Decompositions for Signal Processing Applications: From Two-Way to Multi-Way Component Analysis. *IEEE Transactions on Signal Processing*, **58**, 1226-1241.

[11] Vasilenko, D. and Savich, A. (2016) Multidimensional Signal Processing Using Tensor Decomposition: A Survey. *IEEE Transactions on Signal Processing*, **64**, 1263-1275.

[12] Reichel, L. and Ugwu, U.O. (2021) The Tensor Golub-Kahan-Tikhonov Method Applied to the Solution of Ill-Posed Problems with a T-Product Structure. *Numerical Linear Algebra with Applications*, **29**, e2412. https://doi.org/10.1002/nla.2412

[13] Ugwu, U.O. and Reichel, L. (2021) Tensor Regularization by Truncated Iteration: A Comparison of Some Solution Methods for Large-Scale Linear Discrete Ill-Posed Problems with a T-Product. arXiv preprint arXiv:2110.02485

[14] Zheng, M. and Ni, G. (2023) Approximation Strategy Based on the T-Product for Third-Order Quaternion Tensors with Application to Color Video Compression. *Applied Mathematics Letters*, **140**, Article 108587. https://doi.org/10.1016/j.aml.2023.108587

[15] Khaleel, H.S., Mohd Sagheer, S.V., Baburaj, M. and George, S.N. (2018) Denoising of Rician Corrupted 3D Magnetic Resonance Images Using Tensor-SVD. *Biomedical Signal Processing and Control*, **44**, 82-95. https://doi.org/10.1016/j.bspc.2018.04.004

[16] Cichocki, A., Mandic, D., De Lathauwer, L., Zhou, G., Zhao, Q., Caiafa, C., *et al*. (2015) Tensor Decompositions for Signal Processing Applications: From Two-Way to Multiway Component Analysis. *IEEE Signal Processing Magazine*,

32, 145-163. https://doi.org/10.1109/msp.2013.2297439

[17] Zhang, J., Saibaba, A.K., Kilmer, M.E. and Aeron, S. (2018) A Randomized Tensor Singular Value Decomposition Based on the T-Product. *Numerical Linear Algebra with Applications*, **25**, e2179. https://doi.org/10.1002/nla.2179

[18] Hao, N., Kilmer, M.E., Braman, K. and Hoover, R.C. (2013) Facial Recognition Using Tensor-Tensor Decompositions. *SIAM Journal on Imaging Sciences*, **6**, 437-463. https://doi.org/10.1137/110842570

[19] Kilmer, M.E., Braman, K., Hao, N. and Hoover, R.C. (2013) Third-Order Tensors as Operators on Matrices: A Theoretical and Computational Framework with Applications in Imaging. *SIAM Journal on Matrix Analysis and Applications*, **34**, 148-172. https://doi.org/10.1137/110837711

[20] El Guide, M., El Ichi, A., Jbilou, K. and Sadaka, R. (2021) On Tensor GMRES and Golub-Kahan Methods via the T-Product for Color Image Processing. *The Electronic Journal of Linear Algebra*, **37**, 524-543. https://doi.org/10.13001/ela.2021.5471

[21] Song, H., Wang, S. and Huang, G. (2023) Tensor Conjugate-Gradient Methods for Tensor Linear Discrete Ill-Posed Problems. *AIMS Mathematics*, **8**, 26782-26800. https://doi.org/10.3934/math.20231371

[22] Wang, S., Huang, G. and Yin, F. (2024) Tensor Conjugate Gradient Methods with Automatically Determination of Regularization Parameters for Ill-Posed Problems with T-Product. *Mathematics*, **12**, Article 159. https://doi.org/10.3390/math12010159

[23] Kernfeld, E., Kilmer, M. and Aeron, S. (2015) Tensor-Tensor Products with Invertible Linear Transforms. *Linear Algebra and Its Applications*, **485**, 545-570. https://doi.org/10.1016/j.laa.2015.07.021

[24] Hansen, P.C. (1998) Rank-Deficient and Discrete Ill-Posed Problems. Society for Industrial and Applied Mathematics. https://doi.org/10.1137/1.9780898719697