

求解张量绝对值方程的非精确 LM 方法

马昌凤, 谢亚君

(福州外语外贸学院 大数据学院; 数据科学与智能计算重点实验室, 福州 350202)

摘要:通过引入互补函数将张量绝对值问题重新表述为张量互补问题. 针对重构的张量互补问题, 建立了自适应非精确 LM 算法, 并证明了算法的收敛性. 数值实验结果表明所提出的算法是有效的.

关键词:张量绝对值方程; 非精确 LM 方法; 收敛性分析; 数值实验

中图分类号: O241

文献标志码: A

考虑如下的张量绝对方程(TAVE): 寻找向量 $x \in \mathbf{R}^n$ 满足

$$\mathbf{A}x^{p-1} + \mathbf{B} |x|^{[q-1]} = b, \quad (1)$$

其中 \mathbf{A} 是 p 阶 n 维张量, \mathbf{B} 是 q 阶 n 维张量, 且 p, q 均为偶数. $b \in \mathbf{R}^n$ 为已知向量. 向量 $|x|^{[q-1]}$ 定义为

$$|x|^{[q-1]} = (|x_1|^{[q-1]}, \dots, |x_n|^{[q-1]})^T.$$

当 $p=q=2$ 时, 方程(1)退化为下面的(向量)绝对值方程(AVE):

$$\mathbf{A}x + \mathbf{B} |x| = b. \quad (2)$$

关于方程(2)的数值算法, 已经存在大量研究工作, 可参见文献[1-5]. 特别地, 当 $\mathbf{B}=0$ 时, TAVE(1)退化为多线性系统^[6-9], 它在数据挖掘和数值偏微分中发现了许多重要的应用.

最近, DU 等^[10]考虑了(2)式的另一种特殊情况, 其中设 \mathbf{B} 为负的 p 阶 n 维单位张量(即 $\mathbf{B} |x|^{q-1}$ 退化为 $-|x|^{[p-1]}$), 得到如下方程

$$\mathbf{A}x^{m-1} - |x|^{[m-1]} = b, \quad (3)$$

这等价于广义张量互补问题. 求解此类问题, 由于潜在的非线性(或多重线性), 为 AVE 系统量身定制的所有理论和算法一般不能直接应用于 TAVE(3).

最近, SONG 等^[11]引入了一类互补问题, 称为张量互补问题, 其中所涉及的函数是由 $n \geq 2$ 的齐次多项式定义的. 已知张量互补问题是线性互补问题和一类非线性互补问题的推广. 张量互补问题在 n 人非合作博弈和非线性压缩感知中有许多应用. 最近, 人们对张量互补问题的理论越来越感兴趣, 例如解集的结构和问题的可解性等, 可参见文献[12-14]. 在文献[15]中考虑了另一种与张量有关的互补问题, 称为张量特征值互补问题. 在文献[16]中, 表明 AVE 等效于广义线性互补问题. 类似地, 可以证明 TAVE 等价于广义张量互补问题. 尽管已经提出了一些关于求解 AVE 的数值方法, 但是由于 TAVE(1)本质上是一个非线性方程组, 因此很难推广这些算法去求解 TAVE. 值得一提的是, Levenberg-Marquardt(LM)方法是求解非线性方程的重要算法之一, 因此可以用于求解 TAVE(1).

本文, 在文献[9]的工作上进一步提出自适应的非精确 LM 方法来求解张量绝对值方程, 并且比较了两种算法的求解效率, 理论证明了所提出方法数值是全局超线性收敛, 数值结果表明了算法的有效性.

收稿日期: 2022-03-21; **修回日期:** 2022-05-10.

基金项目: 国家自然科学基金(11901098); 福建省自然科学基金(2020J05034).

作者简介: 马昌凤(1962-), 男, 湖南邵阳人, 福州外语外贸学院教授, 博士, 主要研究方向为数值代数及其应用, E-mail: mcf@fzfu.edu.cn.

通信作者: 谢亚君(1980-), 男, 福建泉州人, 福州外语外贸学院教授, 博士, 主要研究方向为数值代数和图像处理, E-mail: xyj@fzfu.edu.cn.

1 非精确自适应 LM 方法

本节,首先证明 TAVE(3)等价于双线性规划和广义的张量互补问题.首先介绍如下定义.

定义 1 设 \mathbf{A} 是 m 阶 n 维实张量, $x, b \in \mathbf{R}^n$. 定义

$$F(x) = (\mathbf{A} + \mathbf{I})x^{m-1} - b, G(x) = (\mathbf{A} - \mathbf{I})x^{m-1} - b.$$

广义张量互补问题就是寻找向量 $x \in \mathbf{R}^n$ 满足

$$F(x) \geq 0, G(x) \geq 0, F(x)^T G(x) = 0. \quad (4)$$

称下面的非线性规划为双多线性规划问题

$$\min\{F(x)^T G(x) \mid F(x) \geq 0, G(x) \geq 0\} = 0. \quad (5)$$

定理 1 令 $\mathbf{A} \in T(m, n), b \in \mathbf{R}^n$, 那么 TAVE(3) 等价于广义张量互补问题(4) 和双多线性规划(5), 其中 $T(m, n)$ 表示 m 阶 n 维实张量的集合.

证明 显然广义张量互补问题(4)等价于双多线性规划问题(5), 即(4) \Leftrightarrow (5).

所以只需要证明(3) \Leftrightarrow (5).事实上,由于 $|x|^{[m-1]} \geq x^{[m-1]}, |x|^{[m-1]} \geq -x^{[m-1]}$, 因此,由(3)可得:

$$(\mathbf{A} + \mathbf{I})x^{m-1} - b \geq 0, (\mathbf{A} - \mathbf{I})x^{m-1} - b \geq 0.$$

这就意味着 x 是(5)的可行解.因为 $|x|^{[m-1]} = \mathbf{A}x^{m-1} - b \Leftrightarrow [(\mathbf{A} + \mathbf{I})x^{m-1} - b]^T [(\mathbf{A} - \mathbf{I})x^{m-1} - b] = 0$, 于是有

$$|x|^{[m-1]} = \mathbf{A}x^{m-1} - b \Leftrightarrow \min\{F(x)^T G(x) \mid F(x) \geq 0, G(x) \geq 0\} = 0.$$

这就完成了定理证明.

为了求解 TAVE(3),通过上面的定理,提出自适应的 LM 算法用于求解广义的张量互补问题(4).利用 Fischer-Burmeister 函数 ϕ_{FB} , 可以将(4)再生为如下方程:

$$H(x) := \begin{pmatrix} \phi_{FB}(F_1(x), G_1(x)) \\ \vdots \\ \phi_{FB}(F_n(x), G_n(x)) \end{pmatrix} = 0.$$

这里, x 是(4)的解当且仅当 $H(x) = 0$.进一步,由于 $H(x)$ 各个部分组成是强半光滑的,所以 $H(x)$ 是强半光滑的(参见文献[17]).

定理 2 设 \mathbf{A} 是 m 阶 n 维对称张量,则函数 $H(x)$ 是强半光滑的.进一步对任意的 $x \in \mathbf{R}^n$, 有

$$\partial H(x) \subseteq D_a(x)JF(x) + D_b(x)JG(x),$$

其中, $D_a(x) = \text{diag}(a_i(x)), D_b(x) = \text{diag}(b_i(x))$ 是 $\mathbf{R}^{n \times n}$ 对角矩阵, 对应元素

$$(a_i(x), b_i(x)) \in \partial_{\psi_{FB}}(F_i(x), G_i(x)),$$

其中, $\partial_{\psi_{FB}}(F_i(x), G_i(x))$ 是 $\partial_{\psi_{FB}}(a, b)$ 中的 (a, b) 由 $(F_i(x), G_i(x))$ 替换, 并且 $JF(x)$ 和 $JG(x)$ 由下给出 $JF(x) = (\mathbf{A} + \mathbf{I})x^{m-2}, JG(x) = (\mathbf{A} - \mathbf{I})x^{m-2}$.

为了给出求解 $H(x) = 0$ 的算法,定义如下的价值函数

$$\Psi(x) = \frac{1}{2} \|H(x)\|^2.$$

接下来给出价值函数的一个性质.

性质 1 设 \mathbf{A} 是 m 阶 n 维对称张量,那么价值函数 $\Psi(x)$ 是连续可微的并且 $\nabla \Psi(x) = \mathbf{Q}^T H(x)$, 对任意的 $\mathbf{Q} \in \partial H(x)$.

现在提出自适应 LM 算法用于求解半光滑方程系统 $H(x) = 0$, 这实际上是非光滑非精确 LM 类方法.为了确保算法能全局收敛,极小化光滑函数 Ψ 的线搜索被提出,因为张量的数据结构规模比较大,因此非精确的版本的 LM 方法更加适合张量问题,算法如下.

算法 1 (自适应非精确 LM 算法)

步骤 1 给定初始点 x_0 , 选取 $\beta \in (0, \frac{1}{2}), p > 2, \epsilon \geq 0, \gamma \in (0, 1), \rho \in (0, 1), \delta \in (0, 2], \sigma \in (0, 1)$,

置 $k := 0$.

步骤 2 若 $\|H(x_k)\| \leq \varepsilon$, 停算. 否则计算 $Q_k \in \partial H(x_k)$.

步骤 3 计算 LM 参数

$$\mu_k = \|H(x_k)\|^\delta.$$

非精确求解 LM 方程得到近似解 d_k , 使满足

$$(Q_k^T Q_k + \mu_k I) d_k = -Q_k^T H_k + r_k. \tag{6}$$

若如下不等式成立

$$\|H(x_k + d_k)\| \leq \gamma \|H(x_k)\|, \tag{7}$$

则令 $x_{k+1} = x_k + d_k$, 转步骤 5. 否则, 若成立

$$\nabla \Psi(x_k)^T d_k > -\rho \|d_k\|^p, \tag{8}$$

则令 $d_k = -\nabla \Psi(x_k)$.

步骤 4 Armijo 搜索. 令 m_k 是满足下面不等式的最小的非负整数 m :

$$\Psi(x_k + \beta^m d_k) \leq \Psi(x_k) + \sigma \beta^m \nabla \Psi(x_k) d_k,$$

置 $\alpha_k = \beta^{m_k}, x_{k+1} = x_k + \alpha_k d_k$.

步骤 5 置 $k := k + 1$, 转步骤 2.

注 1 在步骤 3 中检查搜索方向 d_k 是否满足(8)式. 这是因为 d_k 不是 LM 方程的精确解, 因此 d_k 可能不是价值函数的下降方向. 如果 d_k 不是下降方向, 则将 d_k 重置为负梯度方向. 因此算法中 d_k 的选择始终是使得价值函数下降的方向, 这样步骤 4 中的线搜索得以明确定义.

下面分析算法 1 的全局收敛性, 假设算法 1 生成无限序列 x_k . 类似于文献[18]中定理 2.1 和定理 2.2 的证明, 可以得到如下的两个定理.

定理 3 设 $\{x_k\}$ 是由算法 1 生成的序列, 如果残差向量 r_k 满足如下条件

$$\|r_k\| \leq \min\{\eta \|Q_k^T H_k\|, \nu_k \|Q_k^T H_k\|^\delta\},$$

其中, $\eta \in (0, 1), \nu_k = o(\text{dist}(x_k, X^*))$, 那么对任意 $\{x_k\}$ 的聚点是 Ψ 的稳定点, 进一步, 如果 $\{x_k\}$ 的聚点 $\{x^*\}$ 是 $H(x) = 0$ 的解, 则 $\{\text{dist}(x_k, X^*)\}$ 超线性收敛到 0.

定理 4 设 $\{x_k\}$ 是由算法 1 生成的序列, $\delta = 2$. 若残差向量 r_k 满足如下条件

$$\lambda \|r_k\| \leq \min\{\eta \|Q_k^T H_k\|, \nu_k \|Q_k^T H_k\|^\delta\},$$

其中, $\eta \in (0, 1), \nu_k = o(\text{dist}(x_k, X^*)^2)$, 则 $\{x_k\}$ 的任意聚点都是 Ψ 的稳定点. 进一步, 如果 $\{x_k\}$ 的聚点 $\{x^*\}$ 是 $H(x) = 0$ 的解, 则 $\{\text{dist}(x_k, X^*)\}$ 二次收敛到 0.

下面再对算法 1 作一些说明. 在算法 1 的执行过程中, 主要计算(6)式当 $r_k = 0$ 时的近似值. 注意到方程总是可解的. 事实上, 由于 $\mu_k > 0$, 那么矩阵 $Q_k^T Q_k + \mu_k I$ 是对称正定矩阵, 因此(6)式一定可解. 现在考虑如何计算 $Q_k \in \partial H(x_k)$, 选择在第 k 步迭代, 通过定理 2, 矩阵 $Q_k \in \partial H(x_k)$ 的元素可以通过如下公式计算. 令

$$\Lambda = \{i : F_i(x_k) = 0 = G_i(x_k)\}$$

是一个退化指数集合, 定义 $z \in \mathbf{R}^n$ 是一个向量, 其分量 $z_i = 1$, 若 $i \in \Lambda$; 否则 $z_i = 0$. 那么矩阵 Q_k 可以如下定义 $Q_k = A(x_k) JF(x_k) + B(x_k) JG(x_k)$. 其中 A 和 B 是 $n \times n$ 矩阵其第 i 个对角元素, 可分别如下给出:

$$A_{ii}(x_k) = \begin{cases} 1 - \frac{F_i(x_k)}{\sqrt{F_i^2(x_k) + G_i^2(x_k)}}, & \text{若 } i \notin \Lambda, \\ 1 - \frac{\nabla F_i(x_k)^T z}{\sqrt{(\nabla F_i(x_k)^T z)^2 + (\nabla G_i(x_k)^T z)^2}}, & \text{若 } i \in \Lambda. \end{cases}$$

$$B_{ii}(x_k) = \begin{cases} 1 - \frac{G_i(x_k)}{\sqrt{F_i^2(x_k) + G_i^2(x_k)}}, & \text{若 } i \notin \Lambda, \\ 1 - \frac{\nabla G_i(x_k)^T z}{\sqrt{(\nabla F_i(x_k)^T z)^2 + (\nabla G_i(x_k)^T z)^2}}, & \text{若 } i \in \Lambda. \end{cases}$$

后面实验部分用这个公式计算 Q_k .

2 数值实验

本节, 将用一个数值算例来说明用算法 1 求解张量绝对值问题(TAVE)的可行性及有效性. 数值实验在 MATLAB R2016a 软件上运行, 个人计算机的运行环境为 2.40 GHz 中央处理器(Intel 2(R) Core (TM)), 8.0 GB 内存以及 Windows 10 操作系统.

在整个计算实验中, 算法 1 使用的参数, 取 $\delta = 1, \epsilon = 10^{-6}, \beta = 0.7, \sigma = 0.4, \gamma = 0.95, \rho = 10^{-8}, p = 2.1$. 终止条件为 $\|H(x_k)\| \leq 10^{-6}$.

例 1 首先随机生成一个四阶四维的张量, 其元素属于 $[0, 1]$, 然后将其对称化得到非负张量 B . 令

$$c = a + (1 + 0.01) \max_{1 \leq i \leq n} (Be^3)_i, \quad (9)$$

其中 $e = (1, 1, 1, 1)^T, a \in (0, +\infty)$. 因为 $\max_{1 \leq i \leq n} (Be^3)_i \geq \rho(B)$, 这样 c 的选择确保 $c > \rho(B) + 1$. 那么令 $A = cI - B$ 满足 $A - I$ 是强 M 张量. 张量 B 和张量 A 见表 1 和表 2.

表 1 随机对称非负张量 $B = (b_{i_1 i_2 i_3 i_4})$

Tab. 1 Random symmetric nonnegative tensors $B = (b_{i_1 i_2 i_3 i_4})$

$b_{2411} = 0.082\ 2$	$b_{3411} = 0.026\ 3$	$b_{4211} = 0.082\ 2$	$b_{4311} = 0.026\ 3$	$b_{4411} = 0.003\ 0$	$b_{1421} = 0.082\ 2$	$b_{2221} = 0.114\ 0$
$b_{2321} = 0.082\ 1$	$b_{3221} = 0.082\ 1$	$b_{4121} = 0.082\ 2$	$b_{4421} = 0.025\ 4$	$b_{1431} = 0.026\ 3$	$b_{2231} = 0.082\ 1$	$b_{4131} = 0.026\ 3$
$b_{4431} = 0.052\ 6$	$b_{1241} = 0.082\ 2$	$b_{1341} = 0.026\ 3$	$b_{1441} = 0.003\ 0$	$b_{2141} = 0.082\ 2$	$b_{2441} = 0.025\ 4$	$b_{1241} = 0.082\ 2$
$b_{1341} = 0.026\ 3$	$b_{1441} = 0.003\ 0$	$b_{2141} = 0.082\ 2$	$b_{2441} = 0.025\ 4$	$b_{3141} = 0.026\ 3$	$b_{3441} = 0.052\ 6$	$b_{4141} = 0.003\ 0$
$b_{4241} = 0.025\ 4$	$b_{4341} = 0.052\ 6$	$b_{1412} = 0.082\ 2$	$b_{2212} = 0.114\ 0$	$b_{2312} = 0.082\ 1$	$b_{3212} = 0.082\ 1$	$b_{4112} = 0.082\ 2$
$b_{4412} = 0.025\ 4$	$b_{1222} = 0.114\ 0$	$b_{1322} = 0.082\ 1$	$b_{2122} = 0.114\ 0$	$b_{2322} = 0.087\ 0$	$b_{3122} = 0.082\ 1$	$b_{3222} = 0.087\ 0$
$b_{1232} = 0.082\ 1$	$b_{2132} = 0.082\ 1$	$b_{2232} = 0.087\ 0$	$b_{3332} = 0.121\ 5$	$b_{3432} = 0.031\ 8$	$b_{1232} = 0.082\ 1$	$b_{2132} = 0.082\ 1$
$b_{2232} = 0.087\ 0$	$b_{3332} = 0.121\ 5$	$b_{3432} = 0.031\ 8$	$b_{4332} = 0.031\ 8$	$b_{4432} = 0.081\ 9$	$b_{1142} = 0.082\ 2$	$b_{1442} = 0.025\ 4$
$b_{3342} = 0.031\ 8$	$b_{3442} = 0.081\ 9$	$b_{4142} = 0.025\ 4$	$b_{4342} = 0.081\ 9$	$b_{1413} = 0.026\ 3$	$b_{2213} = 0.082\ 1$	$b_{4113} = 0.026\ 3$
$b_{4413} = 0.052\ 6$	$b_{1223} = 0.082\ 1$	$b_{2123} = 0.082\ 1$	$b_{2223} = 0.087\ 0$	$b_{3323} = 0.121\ 5$	$b_{3423} = 0.031\ 8$	$b_{4323} = 0.031\ 8$
$b_{4423} = 0.081\ 9$	$b_{2333} = 0.121\ 5$	$b_{2433} = 0.031\ 8$	$b_{3233} = 0.121\ 5$	$b_{4233} = 0.031\ 8$	$b_{1143} = 0.026\ 3$	$b_{1443} = 0.052\ 6$
$b_{2343} = 0.031\ 8$	$b_{2443} = 0.081\ 9$	$b_{3243} = 0.031\ 8$	$b_{4143} = 0.052\ 6$	$b_{4243} = 0.081\ 9$	$b_{1214} = 0.082\ 2$	$b_{1314} = 0.026\ 3$
$b_{1414} = 0.003\ 0$	$b_{2114} = 0.082\ 2$	$b_{2414} = 0.025\ 4$	$b_{3114} = 0.026\ 3$	$b_{3414} = 0.052\ 6$	$b_{4141} = 0.003\ 0$	$b_{4214} = 0.025\ 4$
$b_{4314} = 0.052\ 6$	$b_{1124} = 0.082\ 2$	$b_{1424} = 0.025\ 4$	$b_{3324} = 0.031\ 8$	$b_{3424} = 0.081\ 9$	$b_{4124} = 0.025\ 4$	$b_{4324} = 0.081\ 9$
$b_{1134} = 0.026\ 3$	$b_{1434} = 0.052\ 6$	$b_{2334} = 0.031\ 8$	$b_{2434} = 0.081\ 9$	$b_{3234} = 0.031\ 8$	$b_{4134} = 0.052\ 6$	$b_{4234} = 0.081\ 9$
$b_{1144} = 0.003\ 0$	$b_{1244} = 0.025\ 4$	$b_{1344} = 0.052\ 6$	$b_{2144} = 0.025\ 4$	$b_{2344} = 0.081\ 9$	$b_{3144} = 0.052\ 6$	$b_{3244} = 0.081\ 9$

这个例子主要用于观察算法 1 的迭代过程. 随机生成对称张量 $B \in S^{[4,4]}$ 和随机向量 $x^* \in \mathbf{R}^4$ 且张量 B 和向量 x^* 元素取值均在 $[0, 1]$. 为了确保方程只有唯一的解, 计算 $b = Ax^*{}^{m-1} - |x^*|^{[m-1]}$. 这里取 $a = 3$, 从而计算出 $c = 4.899\ 8$. 为了检验本文算法 1 的有效性, 将其与文献[9]中算法 3.1 进行对比发现, 要达到相同的精度, 本文的算法 1 在迭代次数上并不具有优势, 但 CPU 时间是占优势的, 参见表 3 中的数值结果. 分析其原因, 文献[9]中算法 3.1 是精确 LM 算法, 每一迭代步需要精确求解 LM 方程, 这是比较耗费时间的. 另外从表 4 可以看到 $\|H(x_k)\|$ 随着迭代次数 k 的增加会快速趋于 0. 此外, $\|\Psi(x_k)\|$ 随着迭代次数 k 的增

加亦会快速趋于 0.这说明了算法具有良好的收敛性.

此外,选择不同的 b 值来测试算法的收敛结果.实验中式(9)中的参数 a 取维 15,数值结果见表 5.其中 x_s 表示方程的解,Iter 表示迭代次数.

表 2 基于张量 B 的对称张量 $A = (a_{i_1 i_2 i_3 i_4})$

Tab. 2 Symmetric tensors $A = (a_{i_1 i_2 i_3 i_4})$ based on the tensor B

$a_{1111}=4.899\ 8$	$a_{2411}=-0.082\ 2$	$a_{3411}=-0.026\ 3$	$a_{4211}=-0.082\ 2$	$a_{4311}=-0.026\ 3$	$a_{4411}=-0.003\ 0$	$a_{1421}=-0.082\ 2$
$a_{2221}=-0.114\ 0$	$a_{2321}=-0.082\ 1$	$a_{3221}=-0.082\ 1$	$a_{4121}=-0.082\ 2$	$a_{4421}=-0.025\ 4$	$a_{1431}=-0.026\ 3$	$a_{2231}=-0.082\ 1$
$a_{4131}=-0.026\ 3$	$a_{4431}=-0.052\ 6$	$a_{1241}=-0.082\ 2$	$a_{1341}=-0.026\ 3$	$a_{1441}=-0.003\ 0$	$a_{2141}=-0.082\ 2$	$a_{2441}=-0.025\ 4$
$a_{1241}=-0.082\ 2$	$a_{1341}=-0.026\ 3$	$a_{1441}=-0.003\ 0$	$a_{2141}=-0.082\ 2$	$a_{2441}=-0.025\ 4$	$a_{3141}=-0.026\ 3$	$a_{3441}=-0.052\ 6$
$a_{4141}=-0.003\ 0$	$a_{4241}=-0.025\ 4$	$a_{4341}=-0.052\ 6$	$a_{1412}=-0.082\ 2$	$a_{2212}=-0.114\ 0$	$a_{2312}=-0.082\ 1$	$a_{3212}=-0.082\ 1$
$a_{4112}=-0.082\ 2$	$a_{4412}=-0.025\ 4$	$a_{1222}=-0.114\ 0$	$a_{1322}=-0.082\ 1$	$a_{2122}=-0.114\ 0$	$a_{2222}=4.899\ 8$	$a_{2322}=-0.087\ 0$
$a_{3122}=-0.082\ 1$	$a_{3222}=-0.087\ 0$	$a_{1232}=-0.082\ 1$	$a_{2132}=-0.082\ 1$	$a_{2232}=-0.087\ 0$	$a_{3332}=-0.121\ 5$	$a_{3432}=-0.031\ 8$
$a_{1232}=-0.082\ 1$	$a_{2132}=-0.082\ 1$	$a_{2232}=-0.087\ 0$	$a_{3332}=-0.121\ 5$	$a_{3432}=-0.031\ 8$	$a_{4332}=-0.031\ 8$	$a_{4432}=-0.081\ 9$
$a_{1142}=-0.082\ 2$	$a_{1442}=-0.025\ 4$	$a_{3342}=-0.031\ 8$	$a_{3442}=-0.081\ 9$	$a_{4142}=-0.025\ 4$	$a_{4342}=-0.081\ 9$	$a_{1413}=-0.026\ 3$
$a_{2213}=-0.082\ 1$	$a_{4113}=-0.026\ 3$	$a_{4413}=-0.052\ 6$	$a_{1223}=-0.082\ 1$	$a_{2123}=-0.082\ 1$	$a_{2223}=-0.087\ 0$	$a_{3323}=-0.121\ 5$
$a_{3423}=-0.031\ 8$	$a_{4323}=-0.031\ 8$	$a_{4423}=-0.081\ 9$	$a_{2333}=-0.121\ 5$	$a_{2433}=-0.031\ 8$	$a_{3233}=-0.121\ 5$	$a_{4233}=-0.031\ 8$
$a_{1143}=-0.026\ 3$	$a_{1443}=-0.052\ 6$	$a_{2343}=-0.031\ 8$	$a_{2443}=-0.081\ 9$	$a_{3243}=-0.031\ 8$	$a_{3333}=4.899\ 8$	$a_{4143}=-0.052\ 6$
$a_{4243}=-0.081\ 9$	$a_{1214}=-0.082\ 2$	$a_{1314}=-0.026\ 3$	$a_{1414}=-0.003\ 0$	$a_{2114}=-0.082\ 2$	$a_{2414}=-0.025\ 4$	$a_{3114}=-0.026\ 3$
$a_{3414}=-0.052\ 6$	$a_{4141}=-0.003\ 0$	$a_{4214}=-0.025\ 4$	$a_{4314}=-0.052\ 6$	$a_{1124}=-0.082\ 2$	$a_{1424}=-0.025\ 4$	$a_{3324}=-0.031\ 8$
$a_{3424}=-0.081\ 9$	$a_{4124}=-0.025\ 4$	$a_{4324}=-0.081\ 9$	$a_{1134}=-0.026\ 3$	$a_{1434}=-0.052\ 6$	$a_{2334}=-0.031\ 8$	$a_{2434}=-0.081\ 9$
$a_{3234}=-0.031\ 8$	$a_{4134}=-0.052\ 6$	$a_{4234}=-0.081\ 9$	$a_{1144}=-0.003\ 0$	$a_{1244}=-0.025\ 4$	$a_{1344}=-0.052\ 6$	$a_{2144}=-0.025\ 4$
$a_{2344}=-0.081\ 9$	$a_{3144}=-0.052\ 6$	$a_{3244}=-0.081\ 9$	$a_{4444}=4.899\ 8$			

表 3 算法 1 与文献[9]中算法 3.1 比较

Tab. 3 Algorithm 1 is compared with algorithm 3.1 in Reference [9]

算法	迭代数(k)	CUP 时间/s	$\ H(x_k)\ $ 的值	算法	迭代数(k)	CUP 时间/s	$\ H(x_k)\ $ 的值
文献[9]中算法 3.1	7	0.158 4	3.1573e-07	本文中算法 1	8	0.023 9	5.9422e-07

表 4 算法 1 的迭代过程

Tab. 4 The iterative process of algorithm 1

迭代数(k)	x_k	$\ H(x_k)\ $	迭代数(k)	x_k	$\ H(x_k)\ $
0	(0.709 400 0,0.754 700 0,0.276 000 0,0.679 700 0) ^T	1.316 3	5	(0.775 324 9,0.600 577 4,0.538 544 9,0.802 225 1) ^T	0.001 1
1	(0.842 230 7,0.697 481 1,1.242 378 9,0.913 229 9) ^T	6.327 0	6	(0.775 350 1,0.600 652 8,0.538 973 6,0.802 259 5) ^T	9.292 8e-05
2	(0.794 216 7,0.651 927 3,0.840 643 9,0.830 473 9) ^T	1.520 7	7	(0.775 348 1,0.600 646 7,0.538 940 0,0.802 256 7) ^T	7.427 1e-06
3	(0.780 552 9,0.615 707 9,0.622 814 9,0.809 236 1) ^T	0.286 6	8	(0.775 348 3,0.600 647 2,0.538 941 8,0.802 256 9) ^T	5.942 2e-07
4	(0.775 697 3,0.601 688 0,0.544 857 4,0.802 732 8) ^T	0.017 3			

3 结 论

本文提出了一个求解连续张量绝对值方程的非精确自适应 LM 算法.分析了算法的全局收敛性和局部二次收敛性.并通过数值实例来验证所做的理论分析及有效性和可行性.数值结果表明,本文所提出的算法是有效的.

表 5 在不同 b 值下算法 1 的收敛效果Tab. 5 Convergence results of the algorithm 1 under different b values

b	x_s	Iter	$\ H(x_k)\ $
(6.519 3;0.291 6;0.397 8;0.687 7) ^T	(0.748 7;0.290 9;0.332 6;0.364 7) ^T	7	8.326 9e-08
(0.519 3;4.291 6;1.397 8;0.687 7) ^T	(0.349 4;0.658 9;0.461 9;0.369 2) ^T	8	1.787 6e-07
(12.519 2;4.291 6;0.397 8;1.687 7) ^T	(0.927 4;0.665 8;0.368 4;0.500 4) ^T	13	1.945 7e-07
(3.810 5;5.659 2;4.160 0;1.269 5) ^T	(0.547 0;0.296 3;0.744 6;0.188 9) ^T	10	8.288 8e-08
(14.672 6;5.015 9;8.730 8;0.820 2) ^T	(0.984 7;0.715 6;0.838 9;0.433 2) ^T	11	9.887 2e-08
(8.819 5;9.291 3;0.307 5;0.690 8) ^T	(0.831 0;0.850 5;0.343 3;0.397 9) ^T	8	1.103 6e-07
(9.229 2;8.943 1;-0.244 9;3.442 5) ^T	(0.845 4;0.839 8;0.212 8;0.618 7) ^T	9	5.936 9e-07
(13.701 0;3.311 5;-0.087 1;-0.009 3) ^T	(0.954 2;0.606 5;0.155 3;0.225 4) ^T	8	6.732 3e-07
(5.307 4;15.341 8;-0.055 3;13.961 4) ^T	(0.723 7;1.003 8;0.353 5;0.969 2) ^T	9	9.186 4e-07

参 考 文 献

- [1] MANGASARIAN O L, MEYER R R. Absolute value equations[J]. Linear Algebra Appl, 2006, 419: 359-367.
- [2] PROKOPYEV O. On equivalent reformulations for absolute value equations[J]. Comput Optim Appl, 2009, 44: 363-372.
- [3] NOOR M A, IQBAL J, AL-SAID E. Residual iterative method for solving absolute value equations[J]. Abstr Appl Anal, 2012, 21: 121-129.
- [4] NOOR M A, IQBAL J, KHATTRI S, et al. A new iterative method for solving absolute value equations[J]. Int J Phys Sci, 2011, 6: 1793-1797.
- [5] NOOR M A, IQBAL J, NOOR K I, et al. On an iterative method for solving absolute value equations[J]. Optim Lett, 2012, 6: 1027-1033.
- [6] DING W, WEI Y. Solving multilinear systems with M-tensors[J]. J Sci Comput, 2016, 68: 689-715.
- [7] WANG X, CHE M, WEI Y. Neural networks based approach solving multi-linear systems with M-tensors[J]. Neurocomputing, 2019, 351: 33-42.
- [8] LI X T, NG M K. Solving sparse non-negative tensor equations, algorithms and applications[J]. Front Math China, 2015, 10: 649-680.
- [9] LV C Q, MA C F. A Levenberg-Marquardt method for solving semi-symmetric tensor equations [J]. J Comput Appl Math, 2018, 332: 13-25.
- [10] DU S, ZHANG L, CHEN C, et al. Tensor absolute value equations[J]. Sci China Math, 2018, 61: 1695-1710.
- [11] SONG Y, QI L. Tensor complementarity problem and semi-positive tensors[J]. J Optim Theory Appl, 2016, 169: 1069-1078.
- [12] HUANG Z, QI L. Formulating an n-person noncooperative game as a tensor complementarity problem[J]. Comput Optim Appl, 2017, 66: 557-576.
- [13] LUO Z Y, QI L, XIU N H. The sparse solutions to Z-tensor complementarity problems[J]. Optim Lett, 2017, 11: 471-482.
- [14] SONG Y, YU G. Properties of solution set of tensor complementarity problem[J]. J Optim Theory Appl, 2016, 170: 85-96.
- [15] CHEN H, CHEN Y, LI G, et al. A semidefinite program approach for computing the maximum eigenvalue of a class of structured tensors and its applications in hypergraphs and copositivity test[J]. Numer Linear Algebra Appl, 2018, 25(1): e2125.
- [16] PROKOPYEV O. On equivalent reformulations for absolute value equations[J]. Comput Optim Appl, 2009, 44: 363-372.
- [17] CLARKE F H. Optimization and Nonsmooth Analysis[M]. New York: Wiley, 1983.
- [18] DAN H, YAMASHITA N, FUKUSHIMA M. Convergence properties of the inexact Levenberg-Marquardt type under local error bound conditions[J]. Optim Meth Soft, 2001, 17: 605-626.

The inexact LM method for tensor absolute value equation

Ma Changfeng, Xie Yajun

(School of Big Data; Key Laboratory of Data Science and Intelligent Computing,
Fuzhou University of International Studies and Trade, Fuzhou 350202, China)

Abstract: In this paper, the tensor absolute value equations is reformulated into tensor complementary problem by introducing complementary function. For the reformulated tensor complementary problem, an adaptive inexact Levenberg-Marquardt (LM) algorithm is proposed. And convergence theorem of the algorithm is proved. Numerical experiments show that the proposed algorithm is effective.

Keywords: tensor absolute value equation; inexact LM algorithm; convergence analysis; numerical experiment