具有一般对称性的广义典范多元张量分解

Alex Mulrooney，*IEEE* 学生会员，和 David Hong，*IEEE* 会员

arXiv:2601.05335v1 [math.NA] 8 Jan 2026

摘要—**典范多元（CP）张量分解是发现张量数据中潜在低维结构的主力算法。传统 CP 分解通过根据最小二乘损失将低秩张量拟合到数据上来实现这一点。广义 CP（GCP）分解对这一方法进行了推广，允许使用更一般的损失函数；例如，这类函数可能更适合对二元数据和计数数据建模，或提高对离群值的鲁棒性。然而，GCP 分解并未明确考虑张量中的任何对称性，而这种对称性在现代应用中十分常见。例如，沿时间维度堆叠动态网络的邻接矩阵所形成的张量，自然会在对应图节点的两个模态上呈现对称性。本文提出对称广义 CP（SymGCP）分解，支持一般形式的对称性，即任意子集模态上的对称性。SymGCP 通过在分解中强制满足相应的对称性来处理对称结构。我们推导了 SymGCP 的梯度，使其能够借助现有张量核，通过整体优化高效计算。梯度的形式还导出了多种随机近似，使我们能够开发可扩展至大型张量的随机 SymGCP 算法。我们通过在合成数据和真实数据上开展多种实验，展示了所提出的 SymGCP 算法的实用性。**

索引词—**广义典范多元（GCP）张量分解、CANDECOMP、PARAFAC、对称张量分解。**

I. 引言

张量分解技术是分析数据张量的基础工具，即用于分析以多维数组形式组织的数据。特别是，典范多元（CP）张量分解 [1]–[3] 为揭示数据中潜在的低秩信号提供了强有力的工具。CP 分解通过用秩为 1 的分量张量之和来近似数据，从而得到 Kruskal 张量。在许多情况下，由于这些低秩张量具有唯一性，每个分量都能捕捉数据中的某种潜在现象。CP 分解已应用于众多领域，包括信号处理、计算机视觉、神经科学和深度学习 [4]。

传统的 CP 分解采用最小二乘损失来拟合数据，即使用平方误差来衡量低秩近似对数据张量的拟合程度。针对这种情形，人们开发了许多成熟的算法；概述请参见 [4] 。然而，在某些应用中，其他（非最小二乘）损失函数可能更为合适。尤其值得关注的是用于计数数据的非负最小二乘损失 [5]–[7],泊松损失 [8]–[10], ，以及用于二元数据的伯努利损失 [11], [12]。

A. Mulrooney 和 D. Hong 任职于特拉华大学电气与计算机工程系，地址为美国特拉华州纽瓦克市，邮编 19716。通讯作者：David Hong（电子邮件：hong@udel.edu）。

然而，在各种情形下，采用这些选项之外的其他损失函数也很自然，例如更准确地刻画噪声统计特性，或增强对离群值的鲁棒性。因此，近期出现了若干允许使用更一般损失函数的方法。具体而言，广义 CP（GCP）分解 [13] 通过用户指定的一般损失函数，将低秩张量拟合到数据。[13]提出了一种整体梯度优化方法。他们推导了 GCP 的梯度，并指出这些梯度具有一系列矩阵化张量与 Khatri-Rao 积的乘积（MTTKRP）的形式，而对此已有高度优化的张量核。为将该方法扩展到超大张量， [14]为 GCP 开发了随机梯度，并采用随机梯度方法进行优化。其他方法包括基于适用于各选定损失函数的 Bregman 散度的随机镜像下降算法[15], [16],，以及二阶高斯-牛顿法 [17]。

除了损失函数的选择之外，将有关数据张量的知识纳入分解的另一种方式，是在分解中加入约束。一个重要的约束，也是本文的重点，是分解各模态之间的对称性。在现代数据张量中，对称性自然存在，例如，考虑与高阶统计矩相对应的张量时便是如此。这类张量在所有模态上都是对称的。对称性的另一个例子，是将动态图随时间变化的邻接矩阵堆叠起来形成的张量。这类张量在对应于图节点的两个模态上表现出对称性。自然地，人们希望得到具有相应对称性的分解。针对传统 CP 分解以及通常的最小二乘损失，已有大量研究致力于这一问题。Kolda [18] 提出了一种针对对称 CP 的基于梯度的优化方法，采用最小二乘损失（包括非负最小二乘损失）。最近， [19]提出了一种用于矩张量的专门对称CP分解算法，该算法利用张量结构，高效地对函数值和梯度进行隐式计算。另一种近期方法是子空间幂方法 [20], ，它在矩阵化数据张量的各个子空间中依次寻找最佳秩一张量。迄今为止，整体研究重点一直是计算（最小二乘）对称 CP 分解，其中对称性涵盖张量的所有模态。

本文提出了适用于 CP 分解的算法，既支持一般损失函数（不局限于常见的最小二乘损失），也支持一般形式的对称性（不局限于张量所有模态上的常见对称性）。为此，我们首先定义了一般的张量对称性概念，涵盖了在其指标子集的置换下取值相等的张量。这推广了通常的

张量对称性概念，即在所有指标的置换下取值相等。随后，我们提出对称 GCP（SymGCP）方法，并开发一种基于梯度的整体优化方法，用于将分解拟合到数据。为使该方法适用于大型张量，我们最后开发了一种高效的随机梯度方法，与 [14]类似。我们通过合成数据集和真实数据集展示了所提出的 SymGCP 算法的实用性，这些数据集涵盖多种数据类型，包括二元数据和计数数据。

第二节回顾了张量的相关符号和背景。第三节定义了本文所考虑的一般形式的张量对称性。第四节对所提出的 SymGCP 分解进行了表述。随后，第五节推导了相应的梯度公式，以实现高效计算；第六节则推导了随机梯度，通过随机优化进一步提升可扩展性。最后，第七节和第八节通过合成数据和真实数据上的实验展示 SymGCP，第九节给出结论。

II. 张量符号

本节介绍张量及张量分解的符号和运算，这些内容将在本文后续部分中使用。

我们使用小写非粗体字母（例如 z）表示标量，使用小写粗体字母（例如 $\boldsymbol{y}$）表示向量，使用大写粗体字母（例如 $\boldsymbol{A}$）表示矩阵，并使用花体粗体字母（例如 $\boldsymbol{\mathcal{X}}$）表示张量。我们记作 $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$，表示 $\boldsymbol{\mathcal{X}}$ 是一个具有 N 个模态的实值张量，其沿模态 n 的大小为 I_n。我们将张量 $\boldsymbol{\mathcal{X}}$在索引 $(i_1, \ldots, i_N)$ 处的元素记作 $x_{i_1,\ldots,i_N}$。有时，我们也使用多重索引 $i = (i_1, \ldots, i_N)$ 将元素记作 x_i。

我们将张量 $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ 的向量化表示为

$$\operatorname{vec}(\boldsymbol{\mathcal{X}}) = \begin{bmatrix} x_{1,1,\ldots,1} \\ x_{2,1,\ldots,1} \\ \vdots \\ x_{I_1,1,\ldots,1} \\ x_{1,2,\ldots,1} \\ \vdots \\ x_{I_1,I_2,\ldots,I_N} \end{bmatrix} \in \mathbb{R}^{I_1 \cdots I_N} \tag{1}$$

张量 $\boldsymbol{\mathcal{X}}$ 沿模式 n 的纤维将行和列的概念推广到了高阶张量；每条纤维都是一个向量，由沿给定模式收集的元素构成，即它们是形如 $\boldsymbol{\mathcal{X}}_{i_1,\ldots,i_{n-1},:,i_{n+1},\ldots,i_N} \in \mathbb{R}^{I_n}$ 的切片。张量 $\boldsymbol{\mathcal{X}}$的模式-n 展开记为 $\boldsymbol{X}_{(n)}$，是一个大小为 $I_n \times \prod_{j \neq n} I_j$ 的矩阵，由张量 $\boldsymbol{\mathcal{X}}$ 所有模式-n纤维作为列水平堆叠而成。

我们用 $\circ$ 表示外积，用 $\langle \cdot, \cdot \rangle$ 表示内积，用 $\otimes$表示克罗内克积，用 $\odot$表示 Khatri-Rao 积（即按列计算的克罗内克积），并用 $*$表示 Hadamard 积（即逐元素积）。我们用 $[n]$表示集合 $\{1, 2, \ldots, n\}$，其中 n为正整数。

最后，我们将 Kruskal 张量表示为

$$[\![\boldsymbol{\lambda}; \boldsymbol{A}_1, \ldots, \boldsymbol{A}_N]\!] = \sum_{j=1}^{r} \lambda_j \cdot (\boldsymbol{A}_1)_{:,j} \circ \cdots \circ (\boldsymbol{A}_N)_{:,j}, \tag{2}$$

其中 $\boldsymbol{\lambda} \in \mathbb{R}^r$ 是权重向量，$\boldsymbol{A}_n \in \mathbb{R}^{I_n \times r}$ 是每个 $n \in [N]$ 的模式 -n 因子矩阵。该低秩张量是 CP 分解的输出。

III. 具有一般对称性的对称张量

本节介绍我们所考虑的张量对称性的一般形式，为本文提出的对称 GCP 方法（第 IV 节）做铺垫。第 III-A 节介绍常用的完全张量对称性，第 III-B 节介绍我们所考虑的更一般的张量对称性形式，第 III-C 节则介绍相应的对称 Kruskal 张量概念。

A. 完全张量对称性

在介绍我们所考虑的一般张量对称性形式之前，我们先简要回顾通常研究的传统张量对称性形式。具体而言，“对称张量”一词通常指在其所有指标的任意置换下均等价的张量。对于一个N阶张量$\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$，这意味着

$$\forall_{\pi \in \Pi([N])} \; \forall_{i \in [I_1] \times \cdots \times [I_N]} \quad x_{\pi(i)} = x_i, \tag{3}$$

其中，$\Pi([N])$ 表示作用在 $[N]$上的对称群，即 $[N]$的所有置换构成的集合，而置换 $\pi \in \Pi([N])$ 将张量指标 $i = (i_1, \ldots, i_N)$ 置换为

$$\pi(i) = (i_{\pi_1}, \ldots, i_{\pi_N}). \tag{4}$$

这等价地可以写成

$$\forall_{\pi \in \Pi([N])} \quad \operatorname{permutedims}(\boldsymbol{\mathcal{X}}, \pi) = \boldsymbol{\mathcal{X}}, \tag{5}$$

其中，permutedims 根据 π对$\boldsymbol{\mathcal{X}}$ 的模态进行置换。用群论的语言来说，这种逐模态置换是与我们的语境相关的$\boldsymbol{\mathcal{X}}$ 上的 π 群作用，而这里的对称性是指$\boldsymbol{\mathcal{X}}$ 在 $\Pi([N])$下保持不变。

请注意，这种对称性概念要求 $I_1 = \cdots = I_N$。本文将此类张量称为“完全对称”，以区别于下文描述的更一般的对称形式。

B. 一般张量对称性

现在，我们定义本文关注的一般张量对称性概念。具体而言，受动态图等应用的启发，我们希望张量对称性概念允许仅在部分模态之间具有对称性。请注意，将对称邻接矩阵沿第三个模态堆叠而成的张量，在前两个模态之间是对称的，但在全部三个模态之间并不对称。因此，我们放宽了这一要求：张量的元素不必在其所有指标的任意置换下都相等，而只需在某些指定指标子集的任意置换下相等。

举一个具体例子，假设我们有一个三阶张量$\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$，其前两个模态具有对称性。这意味着

$$\forall_{i \in [I_1] \times [I_2] \times [I_3]} \quad x_{i_1,i_2,i_3} = x_{i_2,i_1,i_3}. \tag{6}$$

换言之，对于 $i_3 \in [I_3]$ ，前切片 $\boldsymbol{\mathcal{X}}_{:,:,i_3}$ 都是对称矩阵。这种对称性可以通过枚举张量保持不变的所有置换来系统地编码；在这种情况下，这些置换为$\{(1,2,3),(2,1,3)\}$。也就是说，我们允许模态 1 和 2 任意置换，同时固定模态 3。

换言之，我们将这里的模态划分为若干个“等价”模态子集，对这些子集中的模态进行任意置换都不会改变张量。对于我们的例子，由于张量在模态 1 和 2 之间具有对称性，因此所有模态组成的集合 $\{1,2,3\}$ 的划分为 $\{1,2\}\sqcup\{3\}$。模态 1 和 2 的所有置换构成对称群 $\Pi(\{1,2\})$，而模态 3 的所有置换（即平凡置换）构成对称群 $\Pi(\{3\})$。将它们组合起来，就得到相关置换群

$$\Pi(\{1,2\}) \times \Pi(\{3\}) = \{(1,2,3),(2,1,3)\}, \tag{7}$$

其中 $\times$ 表示直积。用群论的术语来说，这是一个置换群，也是 $\Pi([N])$ 的子群。

将我们的例子抽象化，自然可以得到以下张量对称性的一般概念。

定义 1 (张量对称性)。 张量 $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1\times\cdots\times I_N}$ 关于其模态划分 $\mathcal{I}_1 \sqcup \cdots \sqcup \mathcal{I}_K = [N]$ 对称，该划分将模态分为 K 个单元，当且仅当

$$\forall_{\pi\in\Pi(\mathcal{I}_1)\times\cdots\times\Pi(\mathcal{I}_K)} \quad \forall_{i\in[I_1]\times\cdots\times[I_N]} \quad x_{\pi(i)} = x_i, \tag{8}$$

其中， $\Pi(\mathcal{I}_1) \times \cdots \times \Pi(\mathcal{I}_K) \subseteq \Pi([N])$ 是仅置换划分中每个单元内指标的所有置换所构成的群。用逐模态置换表示，我们有

$$\forall_{\pi\in\Pi(\mathcal{I}_1)\times\cdots\times\Pi(\mathcal{I}_K)} \quad \operatorname{permutedims}(\boldsymbol{\mathcal{X}},\pi) = \boldsymbol{\mathcal{X}}. \tag{9}$$

该定义要求每个单元中的维度相匹配，即对于任意 j、$k \in \mathcal{I}_\ell$ 和 $\ell \in [K]$， $I_j = I_k$。

这种张量对称性的一般概念涵盖了此前研究的几种情况:

- **完全对称**张量对应于只有一个单元且包含所有指标的划分，即 $\mathcal{I}_1 = [N]$。
- **非对称** 张量对应于每个指标各自构成一个单元的划分，即对于所有 $j \in [N]$， $\mathcal{I}_j = \{j\}$。
- **INDSCAL** [2] 强制三阶张量的前两个模态具有对称性，并对应于划分 $\mathcal{I}_1 \sqcup \mathcal{I}_2 = \{1,2\} \sqcup \{3\}$。

C. 具有一般对称性的对称 *Kruskal* 张量

现在，针对我们所关注的一般张量对称性概念（定义 1），我们定义式（2）中（非对称）Kruskal 张量的一个对称类比。总体思路是：只需约束每个单元中的因子矩阵彼此匹配，即可约束 Kruskal 张量具有适当的对称性。具体而言，给定模态的一个划分 $\mathcal{I}_1\sqcup\cdots\sqcup\mathcal{I}_K = [N]$，我们考虑如下形式的 Kruskal 张量

$$[\![\boldsymbol{\lambda}; \boldsymbol{A}_1,\ldots,\boldsymbol{A}_N]\!] \quad \text{s.t.} \quad \forall_{\ell\in[K]}\,\forall_{j,k\in\mathcal{I}_\ell} \ \boldsymbol{A}_j = \boldsymbol{A}_k. \tag{10}$$

实际上，我们不必为每个模态分别设置因子矩阵，而只需为划分中的每个单元设置一个因子矩阵。由此自然引出对称 Kruskal 张量的以下概念及相应的记号。

定义 2 （对称 Kruskal 张量）。 相对于 N 模态的一个划分 $\mathcal{I}_1 \sqcup \cdots \sqcup \mathcal{I}_K = [N]$，对称 ***Kruskal*** 张量表示为

$$\begin{aligned}[\![\boldsymbol{\lambda}; \mathcal{I}_1 \to \boldsymbol{A}_1,\ldots,\mathcal{I}_K \to \boldsymbol{A}_K]\!] &= [\![\boldsymbol{\lambda}; \boldsymbol{A}_{\sigma_1},\ldots,\boldsymbol{A}_{\sigma_N}]\!] \\ &= \sum_{j=1}^{r} \lambda_j \cdot (\boldsymbol{A}_{\sigma_1})_{:,j} \circ \cdots \circ (\boldsymbol{A}_{\sigma_N})_{:,j},\end{aligned} \tag{11}$$

其中， $\boldsymbol{\lambda} \in \mathbb{R}^r$ 是权重向量，每个 $\boldsymbol{A}_k \in \mathbb{R}^{I_k\times r}$ 是与单元 $\mathcal{I}_k$ 相关联的因子矩阵，而每个 σ_n 用于标识模态 n 所属的单元，即

$$\forall_{k\in[K]}\,\forall_{n\in\mathcal{I}_k} \quad \sigma_n = k. \tag{12}$$

注意，根据构造，任意对称 Kruskal 张量$[\![\boldsymbol{\lambda}; \mathcal{I}_1 \to \boldsymbol{A}_1,\ldots,\mathcal{I}_K \to \boldsymbol{A}_K]\!]$关于划分 $\mathcal{I}_1 \sqcup \cdots \sqcup \mathcal{I}_K$都是对称的。下面的命题将这一说法表述得更加精确。该命题可通过代入和简单改写直接得出；为方便读者，我们在附录 A 中给出一个初等证明。

命题 3 （对称 Kruskal 张量是对称的）。对于 N 个模态的任意划分 $\mathcal{I}_1 \sqcup \cdots \sqcup \mathcal{I}_K$、因子矩阵 $\boldsymbol{A}_1 \in \mathbb{R}^{I_1\times r}$、.. .、$\boldsymbol{A}_K \in \mathbb{R}^{I_K\times r}$ 以及权重向量 $\boldsymbol{\lambda} \in \mathbb{R}^r$，对称 ***Kruskal*** 张量 $[\![\boldsymbol{\lambda}; \mathcal{I}_1 \to \boldsymbol{A}_1$、.. .、$\mathcal{I}_K \to \boldsymbol{A}_K]\!]$关于 $\mathcal{I}_1 \sqcup \cdots \sqcup \mathcal{I}_K$ 是对称的。

IV. 问题表述：对称 GCP（SymGCP）

现在，我们可以陈述所提出的对称 GCP（SymGCP）分解问题了。总体思路是针对一般损失函数，将低秩模型张量拟合到数据上，同时约束模型呈现所需的对称性。具体而言，我们将模型张量约束为定义 2 中所述的对称 Kruskal 张量。由此得到以下 SymGCP 问题表述

$$\min_{\boldsymbol{\lambda},\boldsymbol{A}_1,\ldots,\boldsymbol{A}_K} \underbrace{\mathcal{L}(\boldsymbol{\mathcal{X}}, [\![\boldsymbol{\lambda}; \mathcal{I}_1 \to \boldsymbol{A}_1,\ldots,\mathcal{I}_K \to \boldsymbol{A}_K]\!])}_{=\mathcal{F}(\boldsymbol{\lambda},\boldsymbol{A}_1,\ldots,\boldsymbol{A}_K)}, \tag{13}$$

其中

- $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1\times\cdots\times I_N}$ 是一个 N 阶数据张量，
- $\mathcal{I}_1 \sqcup \cdots \sqcup \mathcal{I}_K = [N]$ 将 N 个模态划分为 _ _
- $\boldsymbol{A}_1 \in \mathbb{R}^{I_1\times r},\ldots,\boldsymbol{A}_K \in \mathbb{R}^{I_K\times r}$ 是相应的因子矩阵，其大小针对每个 $k \in [K]$ 定义如下

$$\widetilde{I}_k = \text{the unique element of } \{I_n : n \in \mathcal{I}_k\},$$

- $\boldsymbol{\lambda} \in \mathbb{R}^r$ 是所需的权重，并且
- $\mathcal{L}$: $\mathbb{R}^{I_1\times\cdots\times I_N} \times \mathbb{R}^{I_1\times\cdots\times I_N} \to \mathbb{R}$是一个一般损失函数，定义在数据张量$\boldsymbol{\mathcal{X}}$与模型张量 $\boldsymbol{\mathcal{M}} = [\![\boldsymbol{\lambda}$之间; $\mathcal{I}_1 \to \boldsymbol{A}_1$、.....、$\mathcal{I}_K \to \boldsymbol{A}_K]\!]$。

请注意，数据张量$\boldsymbol{\mathcal{X}}$的维度必须遵循划分 $\mathcal{I}_1 \sqcup \cdots \sqcup \mathcal{I}_K$ 所定义的对称性，这样 SymGCP 才有意义，即集合 $\{I_n : n \in \mathcal{I}_k\}$ 对于每个 $k \in [K]$都必须只有一个唯一元素。

对于损失函数 $\mathcal{L}$，我们将重点关注以逐元素损失 ℓ 定义的损失函数: $\mathbb{R}\times\mathbb{R}\to\mathbb{R}$ as

$$\mathcal{L}(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{M}}) = \sum_i \ell(x_i, m_i), \tag{14}$$

其中，求和通常遍及张量的所有元素。逐项权重可以直接纳入，如下所示:

$$\mathcal{L}(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{M}}) = \sum_i w_i \ell(x_i, m_i), \tag{15}$$

其中 $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ 是一个相关的权重张量。加权形式使得处理缺失数据变得十分容易，只需将未观测条目的权重设为零即可。在对称张量的背景下，权重还可用于构造损失函数，以避免对称性所产生的重复条目被“重复计数”；有关更多讨论，请参见例如 [18, 第 4 节2] 。

我们以指出（13）中的目标函数具有 Kruskal 张量所固有的缩放歧义来结束本节，因为

$$\begin{aligned}&[\![\boldsymbol{\lambda}; \mathcal{I}_1 \to \boldsymbol{A}_1, \ldots, \mathcal{I}_K \to \boldsymbol{A}_K]\!] \\ &\quad = [\![\rho^N \boldsymbol{\lambda}; \mathcal{I}_1 \to \boldsymbol{A}_1/\rho, \ldots, \mathcal{I}_K \to \boldsymbol{A}_K/\rho]\!],\end{aligned}$$

对于任意 $\rho > 0$均成立。如 [18, 第 4 节4], 所述，在最小二乘损失的背景下也会出现相同的歧义（事实上，任何损失函数都会出现这种歧义）。此外，我们发现，这种歧义在实际优化过程中造成了问题。在此，我们通过添加以下正则项，以类似于 [18]的方式处理这一歧义:

$$r_\gamma(\boldsymbol{A}_1, \ldots, \boldsymbol{A}_K) = \gamma \cdot \sum_{k=1}^{K} \sum_{j=1}^{r} (\|(\boldsymbol{A}_k)_{:,j}\|^2 - 1)^2, \tag{16}$$

其中 $\gamma > 0$ 是正则化参数。该正则项通过惩罚列不具有单位范数的因子矩阵，消除了缩放歧义。

V. 对称 GCP 的梯度

本节推导 SymGCP 问题 (13) 的梯度，说明如何利用现有的张量核计算这些梯度，从而通过基于梯度的优化高效地进行 SymGCP 分解。为简单起见，我们在此仅讨论按 (14) 中的方式逐元素定义的损失函数 $\mathcal{L}$ ；以类似方式即可纳入权重或适用一般损失函数。

下面的定理给出了 SymGCP 梯度的简洁公式。其证明见附录 B。

定理 4 (SymGCP 梯度)。 设 $\mathcal{L}(\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{M}})$ 为一个由逐元素损失 $\ell(x, m)$ 定义的损失函数，如 (14) 所示。则 (13) 中的 *SymGCP* 目标函数 $\mathcal{F}(\boldsymbol{\lambda}, \boldsymbol{A}_1, \ldots, \boldsymbol{A}_K)$ 的梯度为

$$\frac{\partial \mathcal{F}}{\partial \boldsymbol{\lambda}} = (\odot_{n=1}^{N} \boldsymbol{A}_{\sigma_n})^T \operatorname{vec}(\boldsymbol{\mathcal{Y}}), \tag{17}$$

$$\frac{\partial \mathcal{F}}{\partial \boldsymbol{A}_k} = \sum_{t \in \mathcal{I}_k} \boldsymbol{Y}_{(t)} (\odot_{n \neq t} \boldsymbol{A}_{\sigma_n}) \operatorname{diag}(\boldsymbol{\lambda}), \tag{18}$$

其中

$$\odot_{j=1}^{N} \boldsymbol{A}_{\sigma_j} = \boldsymbol{A}_{\sigma_N} \odot \cdots \odot \boldsymbol{A}_{\sigma_1},$$

$$\odot_{j \neq t} \boldsymbol{A}_{\sigma_j} = \boldsymbol{A}_{\sigma_N} \odot \cdots \odot \boldsymbol{A}_{\sigma_{t+1}} \odot \boldsymbol{A}_{\sigma_{t-1}} \odot \cdots \odot \boldsymbol{A}_{\sigma_1},$$

且$\boldsymbol{\mathcal{Y}}$ 是导数张量，其元素为

$$y_i = \frac{\partial \ell}{\partial m_i}(x_i, m_i).$$

该定理将 [18,Section 4.3] 中给出的（最小二乘）对称 CP 的梯度公式推广到一般损失函数。同样，该定理也将 [13, Theorem 3] 中给出的非对称 GCP 的梯度公式推广到一般损失函数。与（非对称）GCP 类似，该定理表明，可以通过一系列张量展开矩阵乘 Khatri-Rao 积（MTTKRP），再对每个单元格中的各个模态进行简单汇总，高效地计算梯度。这使得我们能够利用针对这些运算优化的张量核，高效计算梯度；例如参见 [21]。

请注意，无论$\boldsymbol{\mathcal{X}}$是否具有对称性，定理 4 均成立，即它不必具有任何对称性。因此，在我们预期潜在现象具有对称性，但数据张量因噪声等原因而不对称的情形下，也可以使用这些公式。

当数据张量$\boldsymbol{\mathcal{X}}$ 实际上关于划分$\boldsymbol{\mathcal{I}}_1 \sqcup \cdots \sqcup \boldsymbol{\mathcal{I}}_K$ 对称时（这种情况确实很常见），梯度还可以进一步简化。这依赖于对称 MTTKRP 的以下性质。

命题 5 （对称 MTTKRP）。 若$\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ 关于划分$\boldsymbol{\mathcal{I}}_1 \sqcup \cdots \sqcup \boldsymbol{\mathcal{I}}_K$对称，则对每个 $k \in [K]$，都有

$$\forall_{u,v \in \mathcal{I}_k} \quad \boldsymbol{Y}_{(u)} (\odot_{j \neq u} \boldsymbol{A}_{\sigma_j}) = \boldsymbol{Y}_{(v)} (\odot_{j \neq v} \boldsymbol{A}_{\sigma_j}),$$

其中 σ 与定义 *2* 中相同。

我们在附录 C 中给出一个简单的证明。将该命题应用于梯度公式 (18)，即可立即得到$\boldsymbol{\mathcal{X}}$ 对称时因子矩阵梯度的以下简化形式。

推论 6 （对称数据的 SymGCP 梯度）。 若$\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ 关于划分$\boldsymbol{\mathcal{I}}_1 \sqcup \cdots \sqcup \boldsymbol{\mathcal{I}}_K$对称，则 *SymGCP* 目标函数$\mathcal{F}(\boldsymbol{\lambda}, \boldsymbol{A}_1, \ldots, \boldsymbol{A}_K)$ 关于因子矩阵 $\boldsymbol{A}_k$ 的梯度 (18) 可简化为 *as*

$$\frac{\partial \mathcal{F}}{\partial \boldsymbol{A}_k} = |\mathcal{I}_k| \boldsymbol{Y}_{(k')} (\odot_{j \neq k'} \boldsymbol{A}_{\sigma_j}) \operatorname{diag}(\boldsymbol{\lambda}), \tag{19}$$

其中， $|\mathcal{I}_k|$ 表示 $\mathcal{I}_k$的大小， $k' \in \mathcal{I}_k$ 是 $\mathcal{I}_k$中的任意一个模态。

这种形式能进一步提高 SymGCP 梯度的计算效率，因为它将所需的 MTTKRP 数量从 N 减少到 K。此外，它还用简单的缩放替代了（非对称）梯度的求和。

VI. 对称 GCP 的随机梯度

对于非常大的张量，计算第五节推导出的完整梯度可能会耗费过多资源。本文将提出随机近似，从而通过随机优化实现可扩展的 SymGCP。

特别地，我们考虑与 [14] 针对非对称 GCP 提出的随机梯度类似的方法。该方法的动机来自以下观察：对于一个含有 n^N 个条目的数据张量 $\boldsymbol{\mathcal{X}}$，要计算关于任意因子矩阵或权重的完整梯度，都需要计算并存储导数张量 $\boldsymbol{\mathcal{Y}}$，其大小与 $\boldsymbol{\mathcal{X}}$ 相同。随后，$\boldsymbol{\mathcal{Y}}$ 与因子矩阵的 MTTKRP 各自需要 $\mathcal{O}(Rn^N)$

的计算成本。另一方面，如果$\mathcal{Y}$ 是稀疏的，且包含 nnz($\mathcal{Y}$) 个非零元素，那么计算并存储 $\mathcal{Y}$ 的成本可降至 $\mathcal{O}$ (nnz($\mathcal{Y}$))，而每次计算 MTTKRP 的成本可降至 $\mathcal{O}(r$ nnz($\mathcal{Y}$)) [22]。当 nnz($\mathcal{Y}$) $\ll n^N$ 时，成本将大幅降低。

因此，思路是通过随机子采样$\mathcal{Y}$ ，以较低成本获得完整梯度的随机近似，从而生成~无偏稀疏近似$\mathcal{Y}$。我们考虑两种采样策略（见第六节 A 小节），并推导计算相应随机梯度的高效公式（见第六节 B 小节）。

A. 采样策略

这里，我们简要回顾所考虑的两种采样策略（均匀采样和分层采样），见 [14]。

在均匀采样中，我们抽取一个大小为 s的批次，其中每个样本都是从所有可能的索引集合 $[I_1]\times\cdots\times[I_N]$中均匀随机抽取的一个索引 $i=(i_1,\ i_2,\ \ldots,\ i_N)$，且采用有放回抽样。由于采用有放回抽样，同一个索引可能在一个批次中被抽取多次；令 $\tilde{s}_i$ 表示索引 i 在一个批次中被抽取的次数。现在，为了使$\mathcal{Y}$ 成为$\mathcal{Y}$的无偏估计量，我们令

$$\tilde{y}_i=\tilde{s}_i\frac{n^N}{s}\frac{\partial\ell}{\partial m_i}(x_i,m_i), \tag{20}$$

对于每个抽样到的索引 i 。这种采样策略通常仅在$\mathcal{X}$ 为稠密时最为合适。

在分层采样中，我们改为抽取 p个非零元素和 q 个零元素，也就是说，我们控制所选非零元素~和零元素的数量。此时，为了使$\mathcal{Y}$ 成为$\mathcal{Y}$的无偏估计量，我们设

$$\tilde{y}_i=\tilde{p}_i\frac{\operatorname{nnz}(\mathcal{X})}{p}\frac{\partial\ell}{\partial m_i}(x_i,m_i), \tag{21}$$

对于每个抽样到的非零元素（其中 $\tilde{p}_i$ 是该元素被抽取的次数），并设

$$\tilde{y}_i=\tilde{q}_i\frac{1-\operatorname{nnz}(\mathcal{X})}{q}\frac{\partial\ell}{\partial m_i}(x_i,m_i), \tag{22}$$

对于每个抽样到的零元素（其中 $\tilde{q}_i$ 是该元素被抽取的次数）。当$\mathcal{X}$ 稀疏时，这种采样策略通常最为合适；在这种情况下，均匀采样很少能抽到非零元素，而这些元素可能包含最多信息。当$\mathcal{X}$ 以稀疏坐标格式存储时，可以通过从已存储元素列表中抽样，高效地抽取非零元素。在这种情况下，可以通过拒绝采样来抽取零元素。

B. 计算随机梯度

~有了$\mathcal{Y}$ 的稀疏随机近似$\mathcal{Y}$ 后，只需在 (17) 和 (18) 中用$\mathcal{Y}$ 替换$\mathcal{Y}$ ，即可计算随机梯度。以下~定理说明了如何利用$\mathcal{Y}$ 的稀疏性高效计算这些随机梯度。该方法与非对称 GCP 的方法基本相同；为方便读者，我们在附录 D 中给出证明。

定理 7 (随机对称广义 CP 梯度)。 对于在 s 个指标处具有非零元素的 $\mathcal{Y}$ 的稀疏~随机近似，指标为 $(i_1^1,\ldots,i_N^1),\ldots,(i_1^s,\ldots,i_N^s)$，随机梯度近似可按如下方式计算

$$\frac{\partial\mathcal{F}}{\partial\boldsymbol{\lambda}}\approx(*_{j=1}^N\widehat{\boldsymbol{A}}_{\sigma_j})^T\hat{\boldsymbol{y}}, \tag{23}$$

$$\frac{\partial\mathcal{F}}{\partial\boldsymbol{A}_k}\approx\widehat{\boldsymbol{Y}}_{(n)}(*_{j\neq n}\widehat{\boldsymbol{A}}_{\sigma_j})\operatorname{diag}(\boldsymbol{\lambda}), \tag{24}$$

其中

$$\widehat{\boldsymbol{Y}}_{(n)}=\begin{bmatrix}| & & |\\ \widetilde{\boldsymbol{Y}}_{i_1^1,\ldots,i_{n-1}^1,:,i_{n+1}^1,\ldots,i_N^1} & \cdots & \widetilde{\boldsymbol{Y}}_{i_1^s,\ldots,i_{n-1}^s,:,i_{n+1}^s,\ldots,i_N^s}\\ | & & |\end{bmatrix}$$

and

$$\widehat{\boldsymbol{A}}_{\sigma_j}=\begin{bmatrix}-(\boldsymbol{A}_{\sigma_j})_{i_j^1,:}-\\ \vdots\\ -(\boldsymbol{A}_{\sigma_j})_{i_j^s,:}-\end{bmatrix},\qquad \hat{\boldsymbol{y}}=\begin{bmatrix}\tilde{y}_{i_1^1,\ldots,i_N^1}\\ \vdots\\ \tilde{y}_{i_1^s,\ldots,i_N^s}\end{bmatrix}.$$

VII. 数值实验

本节通过在合成数据集上进行数值实验，评估所提出的 SymGCP 分解。对于模型的非随机拟合，我们使用第五节推导的梯度，并采用带边界约束的有限内存 BFGS（L-BFGS-B） [23]。对于随机算法，我们使用 Adam [24] ，并进行与 [14] 中类似的标准修改。具体而言，我们将迭代点划分为训练周期，每个周期包含预先设定数量的迭代，并在每个训练周期结束时估计目标函数的值。如果目标函数的值未能按预设的相对比例 κ（例如 0.99）下降，我们就将该训练周期判定为"坏训练周期"，并将所有参数（当前因子矩阵的值、一阶和二阶动量估计以及迭代计数器）重置为上一个训练周期保存的值。

A. 全对称二元张量

这里，我们测试 SymGCP 在恢复采用优势比链接的完全对称二元张量底层因子方面的有效性。我们采用与附录 D], 中相同的数据生成过程 [14, , 并对其进行修改以生成完全对称的数据张量。具体而言，为生成每个模态大小为 n、秩为 r m 的阶二元张量，我们首先选择一个稀疏度 δ、信号分量中取 1 的概率 ρ_{high}以及噪声分量中取 1 的概率 ρ_{low}。真实因子矩阵 $\boldsymbol{A}_*$的前 $r-1$ 列指定为信号分量，最后一列指定为噪声分量。对于每个信号分量，我们随机选择比例为 δ 的条目设为非零，并为每个非零条目从 $\mathcal{N}(\sqrt[m]{\rho_{\text{high}}}/(1-\rho_{\text{high}})\,0.5)$ 分布中生成一个值。噪声列中的每个条目都恰好设为$\sqrt[m]{\rho_{\text{low}}}/(1-\rho_{\text{low}})$。最后，利用由~~$\boldsymbol{A}_*$~~生成的真实模型张量 $\boldsymbol{\mathcal{M}}_*$，将数据张量 $\boldsymbol{\mathcal{X}}$ 中的每个条目 x_i 以概率 $m_i^*/(1+m_i^*)$ 设为 1。

为了获得 SymGCP 的初始化，我们首先对一个元素取值为 $\mathcal{N}(0,$ 1) 的随机高斯矩阵进行采样，以此生成初始因子矩阵 $\boldsymbol{A}$ ，然后对其重新缩放，使初始化后的模型张量$\boldsymbol{\mathcal{M}}$ 的范数与数据张量 $\boldsymbol{\mathcal{X}}$的范数相同。我们针对每种初始化，分别使用最小二乘损失和伯努利损失（采用优势比链接）运行 SymGCP,

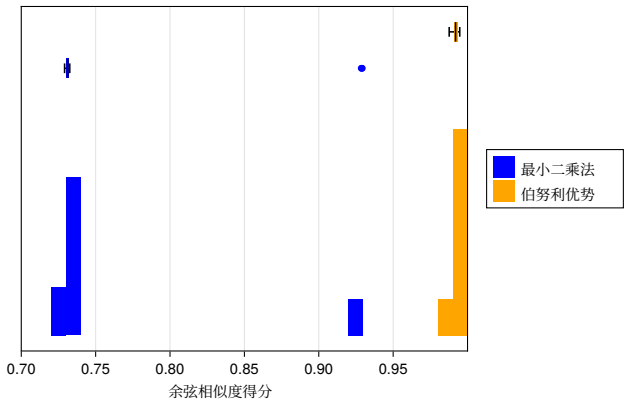

图 1. 在采用最小二乘法和伯努利优势损失函数的二元测试问题上，SymGCP 对每个实例得到的最佳余弦相似度得分直方图。每个实例的最佳相似度得分，是取得最低最终损失值的那次初始化所对应的相似度得分。直方图上方显示了相应的箱线图。

函数$\widehat{}$。为了比较恢复解 $\boldsymbol{A}$ 与真实因子矩阵 $\boldsymbol{A}_*$的接近程度，我们计算余弦相似度得分

$$\frac{1}{r}\sum_{j=1}^{r}\cos\left((\boldsymbol{A}_*)_{:,j},(\widehat{\boldsymbol{A}})_{:,\pi(j)}\right), \tag{25}$$

$\widehat{}$其中 π 是对 $\boldsymbol{A}$的列进行置换后得到的排列，该排列使得得分最高。

在我们的实验中，我们设置了 $m=4$、 $n=50$、 $r=5$、 $\rho_{\text{high}}=0.9$、 $\rho_{\text{low}}=0.002$和 $\delta=0.15$。我们创建了一个真实因子矩阵 $\boldsymbol{A}_*$，随后以此生成了数据张量$\boldsymbol{\mathcal{X}}$ 的 20 个不同实例。对于每个实例，我们针对两种损失函数分别运行了 25 次不同的 SymGCP 初始化。

图 1 比较了最小二乘损失和伯努利优势损失在各个实例上的最佳余弦相似度得分。对于 $\boldsymbol{A}_*$，采用伯努利优势损失的 SymGCP 能更好地恢复底层因子：在 20 个不同实例中，最佳得分的中位数为 0.998，而最小二乘损失的中位数为 0.835。此外，伯努利损失在各实例中的最差得分（0.997）也优于最小二乘损失在各实例中的最佳得分（0.982）。这说明了使用一般损失函数对噪声统计特性进行建模的好处。

B. 随机实验

接下来，我们使用第六节中针对 SymGCP 的随机梯度，测试随机方法在模拟的全对称二元张量数据上的表现；这些数据以与第七节 A 部分相同的方式生成。保持所有实验参数的取值不变，我们创建了一个新的真实因子矩阵 $\boldsymbol{A}_*$，随后再次生成了张量数据 $\boldsymbol{\mathcal{X}}$ 的 20 个不同实例，来自 $\boldsymbol{A}_*$。数据张量的稀疏度介于 0.43% 和 0.44% 之间，非零元素总数介于 26874 和 27272 之间。

对于每个实例，我们使用 L-BFGS-B 和 Adam，在均匀采样和分层采样下分别采用 25 种不同的初始化运行带有伯努利（优势比链接）损失的 SymGCP。对于 Adam，我们在两种采样策略下均使用总样本量 1000；对于分层采样，其中包含 500 个零值和 500 个非零值。

图 2 展示了对于一个具有代表性的数据实例（其他实例的情况类似），SymGCP 使用 Adam（均匀采样和分层采样）以及 L-BFGS-B 时，目标函数值随运行时间的变化情况，以及每种方法在所有数据实例中采用最佳初始化时的余弦相似度得分。对于这些稀疏对称二元张量，使用带分层采样的 Adam 可实现约 50 倍的加速：大多数 L-BFGS-B 运行约需 50 秒才能达到接近由 $\boldsymbol{\mathcal{M}}_*$ 给出的“真实损失”的损失值，而大多数分层 Adam 运行约 1 秒即可达到相同水平。使用均匀采样的 Adam 也比 L-BFGS-B 快得多，但速度慢于分层采样。总体而言，随机方法的迭代成本低得多，因此能够更快地沿目标函数下降。另一方面，由于随机方法无法收敛到同样优质的解，其最终解的准确度往往也较低。尽管如此，随机解仍然与真实的底层因子相当接近：分层采样的余弦相似度得分中位数为 0.990，均匀采样为 0.987，而 L-BFGS-B 的中位数得分为 0.998。

VIII. 真实数据实验

本节在两个真实数据集上评估所提出的 SymGCP 分解。我们使用与第七节中相同的算法。

A. 猴脑神经数据

我们首先将 SymGCP 应用于猴子尝试通过脑机接口（BMI）将光标移向四个不同目标之一时记录的神经数据。这些数据最初收集于一项运动学习任务中，用于研究神经动态 [25], [26]。我们使用为 Tensor Toolbox 软件包演示而整理的数据文件 [27]。该数据集包含一个大小为 $43\times200\times88$ 的张量，其中记录了 43 个不同神经元在连续 200 个时间步内、88 次不同试验中的活动；每次试验中，猴子都尝试将光标移向四个目标之一。此外，还提供了每次试验的目标标签。

我们不分解 $43\times200\times88$ 神经活动张量（记作$\boldsymbol{\mathcal{X}}$），而是考虑由神经元共激活构成的张量。共激活张量$\boldsymbol{\mathcal{C}}$ 的 $(i,\ j,\ k)$ 元素，是$\boldsymbol{\mathcal{X}}$中对应于试验 k 期间神经元 i 和 j 时间序列的模式 2 纤维的内积：

$$c_{i,j,k}=(x_{i,:,k})^T x_{j,:,k}, \tag{26}$$

对所有 $i\in[43]$, $j\in[43]$, $k\in[88]$。注意，$\boldsymbol{\mathcal{C}}$沿其前两个模态对称。

我们使用非负最小二乘损失，对共激活张量 $\boldsymbol{\mathcal{C}}$ 进行不同秩（从 $[2,4,6,8,10,15,20]$起）的 SymGCP 分解。我们使用试次的目标标签来评估分解恢复底层试验结构的效果。为了评估秩为 r 的分解，我们将恢复的试次因子矩阵中的每一行视为

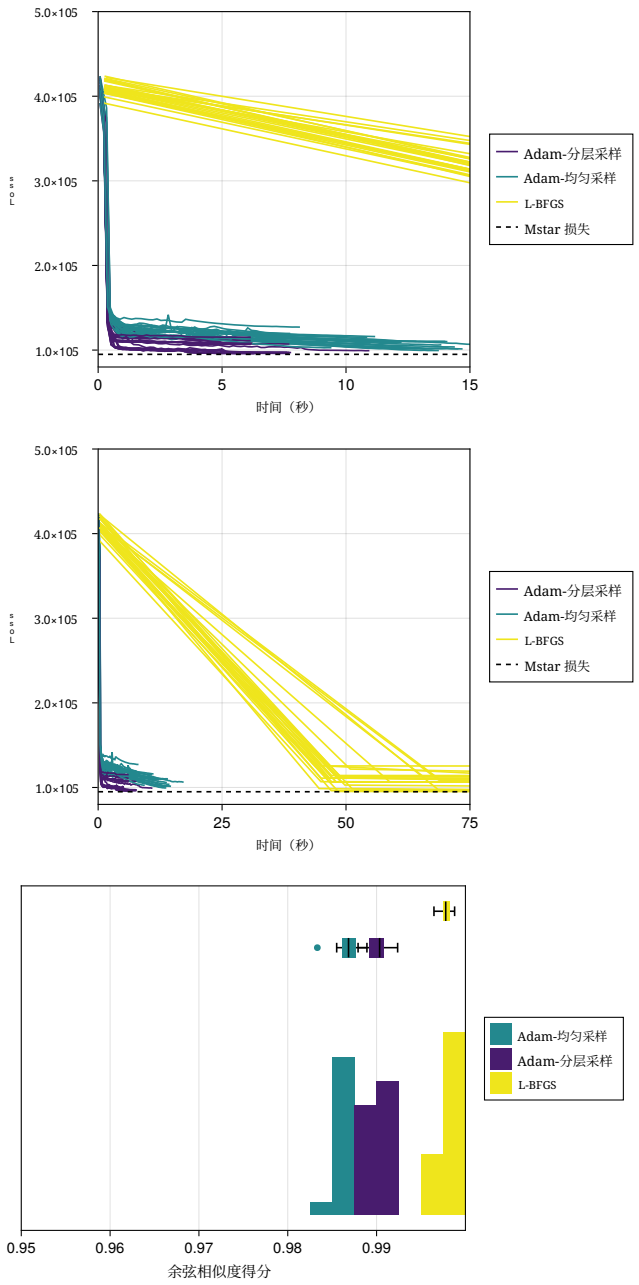

图2。上：在一个具有代表性的数据实例上，使用 Adam（分层采样和均匀采样）及 L-BFGS-B 对二元测试问题进行 SymGCP 时，目标函数值随时间的变化。中：缩小时间轴范围后，目标函数值随时间的变化。下：直方图和箱线图，展示了使用 Adam 和 L-BFGS-B 对二元测试问题进行 SymGCP 时，各数据实例的最佳余弦相似度得分。每个数据实例的最佳相似度得分，是使最终损失值最低的初始化所对应的相似度得分。

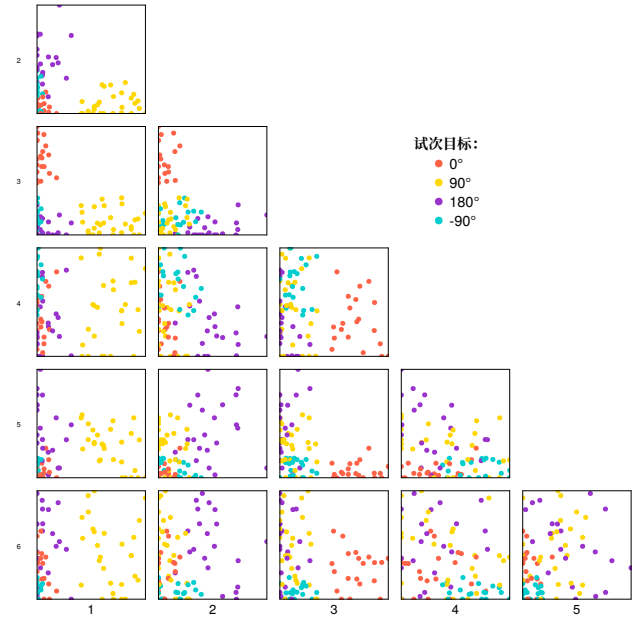

图3。猴脑神经数据的共激活张量经过秩为6的对称非负最小二乘分解后，恢复的试验因子矩阵各行之间的两两关系。试次按目标条件着色。

$\mathbb{R}^r$ 中的一个数据点，并使用具有 4 个质心的 K均值对这些行进行聚类。聚类后，我们记录被正确聚类的试次百分比（即试验因子矩阵中正确聚类的行数占比；在调整质心的任意标签顺序、使准确度达到最高后计算）。

我们发现，低秩 SymGCP 能够恢复数据的试验结构；当秩为6时，对恢复的因子矩阵应用 K均值后，可实现完美聚类。在图3中，我们绘制了秩为3的分解所恢复的因子矩阵中88行（对应88个试次）之间的两两关系，并根据这些行对应的4种目标条件为点着色。尽管共激活张量并未明确包含每个试次的目标条件信息，但对称分解得到的试验因子矩阵清晰地区分了不同的目标条件。

B. 加州大学欧文分校社交网络

接下来，我们在加州大学尔湾分校社交网络数据集 [28],上评估 SymGCP。该数据集记录了社交网络中用户对之间每天的消息数量，共有 1899 位用户，消息发送时间跨度为 193 天。我们从原始数据集出发，其中记录了每天每对用户之间的消息数量，并据此构造一个计数张量：元素 $(x,\ y,\ z)$ 表示用户 x 与 y 在第 z 天交换的消息数量，因此该张量沿前两个模态对称。我们仅纳入最活跃的 200 位用户。尽管我们不了解数据中的潜在结构，但可以观察到，实验前期（大致从第 0 天到第 50 天）的总体通信量明显更高，

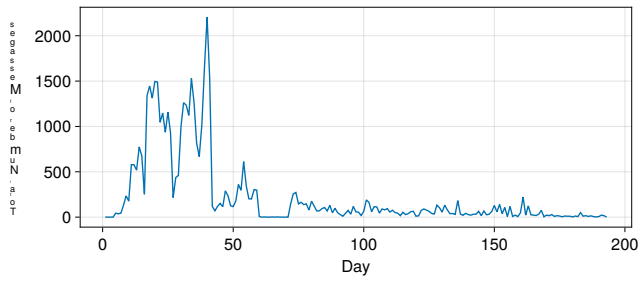

图4。实验期间，加州大学欧文分校社交网络中互动最频繁的200名用户之间每天交换的消息总数。 [28]

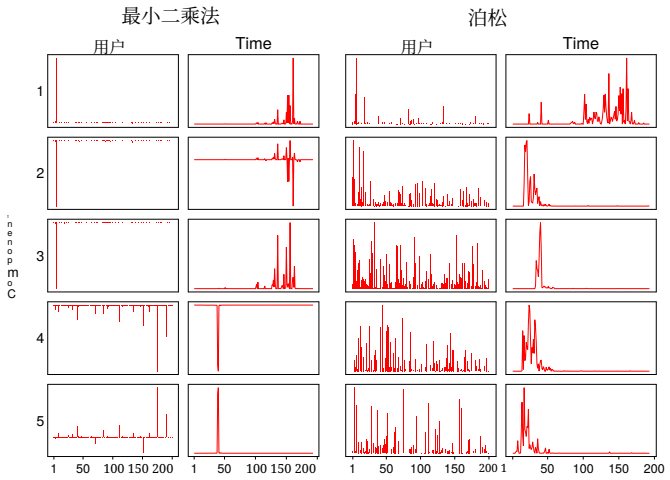

图5。对于加州大学欧文分校社交网络数据集中的200名最活跃用户，采用最小二乘损失的 SymGCP 分解（左，对应于对称 CP）和泊松损失的 SymGCP 分解（右）。对于每种秩为 $r = 10$ 的分解，展示了 λ 中权重最大的前5个成分。

如图4所示，该图展示了每天所有排名前 200 位的用户之间的消息总数。

对于构造的计数张量，我们计算了秩为 r =10 的对称 CP 分解（最小二乘损失），以及采用与泊松分布相对应的损失函数的 SymGCP 分解。图5展示了两种分解按权重降序排列后的前5个成分。与对称CP分解不同，后者得到的成分只关注少数用户和天数（前3个成分似乎都识别出了同一个用户，而成分4和5则分别识别出单独的一天），采用泊松损失的 SymGCP 发现的成分包含更多用户和更长的时间区间，这可能表明它们对应着更有意义的社交群体。具体而言，成分1、2、4和5识别出了实验早期活动度较高的用户群体，与图4中早期总体活动度较高的情况相吻合。

IX. 结论

本文提出了一种新的 CP 分解，可用于具有任意形式对称性且采用一般损失函数的张量。我们定义了张量对称性的一般概念，其中包括以往对称 CP 工作 [18]–[20], 主要关注的完全对称张量，以及部分对称性。我们推导了 SymGCP优化问题的梯度，并采用了一次性基于梯度的方法。我们还提出了高效的梯度随机近似方法，使其能够扩展到超大型张量。数值实验表明，采用适当损失函数的 SymGCP 能够有效恢复对称二元张量中的底层低秩因子；对于大型张量，随机方法还能实现加速。最后，我们将 SymGCP 应用于猴脑神经数据 [25]–[27] 和社交网络数据 [28] ，以发现数据中具有实际意义的底层成分。

未来的一个研究方向是开发更高效的优化算法，用于计算 SymGCP 分解，可能通过进一步利用数据张量和/或模型中的对称结构来实现。例如，对于一个因子矩阵的 MTTKRP 序列所产生的中间结果，或许可以用于减少其他因子矩阵的 MTTKRP 所需的计算量。另一个未来方向是开发用于 SymGCP 的流式/在线算法，以便在数据生成的同时计算其对称广义CP分解，例如应用于大规模模拟等场景。

附录 A 命题 3 的证明

令 $i \in [I_1] \times \cdots \times [I_N]$ 和 $\pi \in \Pi(\mathcal{I}_1) \times \cdots \times \Pi(\mathcal{I}_K)$ 为任意对象。注意到

$$\begin{aligned}
&[\![\boldsymbol{\lambda}; \mathcal{I}_1 \to \boldsymbol{A}_1, \ldots, \mathcal{I}_K \to \boldsymbol{A}_K]\!]_i \\
&\quad = [\![\boldsymbol{\lambda}; \boldsymbol{A}_{\sigma_1}, \ldots, \boldsymbol{A}_{\sigma_N}]\!]_i \\
&\quad = \left(\sum_{j=1}^{r} \lambda_j \cdot (\boldsymbol{A}_{\sigma_1})_{:,j} \circ \cdots \circ (\boldsymbol{A}_{\sigma_N})_{:,j} \right)_i \\
&\quad = \sum_{j=1}^{r} \lambda_j \cdot (\boldsymbol{A}_{\sigma_1})_{i_1,j} \circ \cdots \circ (\boldsymbol{A}_{\sigma_N})_{i_N,j} \\
&\quad = \sum_{j=1}^{r} \lambda_j \prod_{n=1}^{N} (\boldsymbol{A}_{\sigma_N})_{i_n,j} \\
&\quad = \sum_{j=1}^{r} \lambda_j \prod_{\ell=1}^{K} \prod_{k \in \mathcal{I}_\ell} (\boldsymbol{A}_\ell)_{i_k,j} .
\end{aligned} \tag{27}$$

现在，由于 π 仅对划分的单元格内的指标进行置换，因此对于任意 $\ell \in [K]$, $\{\pi(k) \colon k \in \mathcal{I}_\ell\} = \mathcal{I}_\ell$ ，从而

$$\prod_{k \in \mathcal{I}_\ell} (\boldsymbol{A}_\ell)_{i_{\pi(k)},j} = \prod_{k \in \mathcal{I}_\ell} (\boldsymbol{A}_\ell)_{i_k,j} \tag{28}$$

对于任意的 $j \in [r]$ 和 $\ell \in [K]$。因此

$$\begin{aligned}
&[\![\boldsymbol{\lambda}; \mathcal{I}_1 \to \boldsymbol{A}_1, \ldots, \mathcal{I}_K \to \boldsymbol{A}_K]\!]_{\pi(i)} \\
&\quad = \sum_{j=1}^{r} \lambda_j \prod_{\ell=1}^{K} \prod_{k \in \mathcal{I}_\ell} (\boldsymbol{A}_\ell)_{i_{\pi(k)},j} \\
&\quad = \sum_{j=1}^{r} \lambda_j \prod_{\ell=1}^{K} \prod_{k \in \mathcal{I}_\ell} (\boldsymbol{A}_\ell)_{i_k,j} \\
&\quad = [\![\boldsymbol{\lambda}; \mathcal{I}_1 \to \boldsymbol{A}_1, \ldots, \mathcal{I}_K \to \boldsymbol{A}_K]\!]_i,
\end{aligned} \tag{29}$$

由此完成证明。 □

附录 B 定理 4 的证明

回顾 [13, 推论 4]，非对称 GCP 目标函数

$$\mathcal{G}(\boldsymbol{\lambda}, \widetilde{\boldsymbol{A}}_1, \ldots, \widetilde{\boldsymbol{A}}_N) = \mathcal{L}(\boldsymbol{\mathcal{X}}, [\![\boldsymbol{\lambda}; \widetilde{\boldsymbol{A}}_1, \ldots, \widetilde{\boldsymbol{A}}_N]\!]) \tag{30}$$

具有梯度

$$\frac{\partial \mathcal{G}}{\partial \boldsymbol{\lambda}} = (\odot_{j=1}^{N} \widetilde{\boldsymbol{A}}_j)^T \operatorname{vec}(\boldsymbol{\mathcal{Y}}), \tag{31}$$

$$\frac{\partial \mathcal{G}}{\partial \widetilde{\boldsymbol{A}}_k} = \boldsymbol{Y}_{(k)} (\odot_{j \neq k} \widetilde{\boldsymbol{A}}_j) \operatorname{diag}(\boldsymbol{\lambda}), \tag{32}$$

其中

$$\odot_{j=1}^{N} \widetilde{\boldsymbol{A}}_j = \widetilde{\boldsymbol{A}}_N \odot \cdots \odot \widetilde{\boldsymbol{A}}_1,$$
$$\odot_{j \neq k} \widetilde{\boldsymbol{A}}_j = \widetilde{\boldsymbol{A}}_N \odot \cdots \odot \widetilde{\boldsymbol{A}}_{k+1} \odot \widetilde{\boldsymbol{A}}_{k-1} \odot \cdots \odot \widetilde{\boldsymbol{A}}_1,$$

且 $\boldsymbol{\mathcal{Y}}$ 是导数张量，其元素为

$$y_i = \frac{\partial \ell}{\partial m_i}(x_i, m_i).$$

对于 SymGCP，我们现在为对称模态的每个单元格而不是每个模态设置一个因子矩阵。注意，

$$\begin{aligned} &\mathcal{F}(\boldsymbol{\lambda}, \boldsymbol{A}_1, \ldots, \boldsymbol{A}_K) \qquad (33)\\ &\quad = \mathcal{L}(\boldsymbol{\mathcal{X}}, [\![\boldsymbol{\lambda}; \mathcal{I}_1 \to \boldsymbol{A}_1, \ldots, \mathcal{I}_K \to \boldsymbol{A}_K]\!])\\ &\quad = \mathcal{L}(\boldsymbol{\mathcal{X}}, [\![\boldsymbol{\lambda}; \boldsymbol{A}_{\sigma_1}, \ldots, \boldsymbol{A}_{\sigma_N}]\!])\\ &\quad = \mathcal{G}(\boldsymbol{\lambda}, \boldsymbol{A}_{\sigma_1}, \ldots, \boldsymbol{A}_{\sigma_N}), \qquad (34) \end{aligned}$$

因此可以立即得到

$$\frac{\partial \mathcal{F}}{\partial \boldsymbol{\lambda}} = \left(\odot_{j=1}^{N} \boldsymbol{A}_{\sigma_j}\right)^T \operatorname{vec}(\mathcal{Y}). \tag{35}$$

对于因子矩阵的梯度，应用链式法则可得

$$\frac{\partial \mathcal{F}}{\partial \operatorname{vec}(\boldsymbol{A}_k)} = \sum_{t=1}^{N} \left[\frac{\partial \operatorname{vec}(\boldsymbol{A}_{\sigma_t})}{\partial \operatorname{vec}(\boldsymbol{A}_k)}\right]^T \frac{\partial \mathcal{G}}{\partial \operatorname{vec}(\boldsymbol{A}_{\sigma_t})}. \tag{36}$$

现在，对于每个 $t \in [N]$，我们有

$$\frac{\partial \operatorname{vec}(\boldsymbol{A}_{\sigma_t})}{\partial \operatorname{vec}(\boldsymbol{A}_k)} = \begin{cases} \boldsymbol{I}, & \text{if } \sigma_t = k, \\ \boldsymbol{0}, & \text{otherwise}, \end{cases}$$

因此最终可得

$$\frac{\partial \mathcal{F}}{\partial \boldsymbol{A}_k} = \sum_{t \in \mathcal{I}_k} \frac{\partial \mathcal{G}}{\partial \boldsymbol{A}_{\sigma_t}} = \sum_{t \in \mathcal{I}_k} \boldsymbol{Y}_{(t)} \left(\odot_{j \neq t} \boldsymbol{A}_{\sigma_j}\right) \operatorname{diag}(\boldsymbol{\lambda}), \tag{37}$$

由此完成证明。 □

附录 C 命题 5 的证明

不失一般性，假设 $u < v$。利用 Kruskal 张量的众所周知的性质，我们可以将 $\boldsymbol{Y}_{(u)} \left(\odot_{j \neq u} \boldsymbol{A}_{\sigma_j}\right)$ 与任意矩阵 $\boldsymbol{B} \in \mathbb{R}^{s \times r}$ 的内积写为，其中 $s = I_u = I_v$:

$$\begin{aligned} &\left\langle \boldsymbol{B}, \boldsymbol{Y}_{(u)} \left(\odot_{j \neq u} \boldsymbol{A}_{\sigma_j}\right) \right\rangle\\ &\quad = \left\langle \boldsymbol{Y}_{(u)}, \boldsymbol{B} \left(\odot_{j \neq u} \boldsymbol{A}_{\sigma_j}\right)^T \right\rangle\\ &\quad = \left\langle \mathcal{Y}, [\![\boldsymbol{A}_{\sigma_1}, \ldots, \boldsymbol{A}_{\sigma_{u-1}}, \boldsymbol{B}, \boldsymbol{A}_{\sigma_{u+1}}, \ldots, \boldsymbol{A}_{\sigma_N}]\!] \right\rangle. \end{aligned} \tag{38}$$

现在，由于 $\boldsymbol{\mathcal{Y}}$ 沿模态 u 和 v 对称，且 $\boldsymbol{A}_{\sigma_u} = \boldsymbol{A}_{\sigma_v}$，因此有:

$$\begin{aligned} &\left\langle \mathcal{Y}, [\![\boldsymbol{A}_{\sigma_1}, \ldots, \boldsymbol{A}_{\sigma_{u-1}}, \boldsymbol{B}, \boldsymbol{A}_{\sigma_{u+1}}, \ldots, \boldsymbol{A}_{\sigma_N}]\!] \right\rangle\\ &\quad = \left\langle \mathcal{Y}, [\![\boldsymbol{A}_{\sigma_1}, \ldots, \boldsymbol{A}_{\sigma_{u-1}}, \boldsymbol{A}_{\sigma_v}, \boldsymbol{A}_{\sigma_{u+1}}, \right. \qquad (39)\\ &\qquad \left. \ldots, \boldsymbol{A}_{\sigma_{v-1}}, \boldsymbol{B}, \boldsymbol{A}_{\sigma_{v+1}}, \ldots, \boldsymbol{A}_{\sigma_N}]\!] \right\rangle\\ &\quad = \left\langle \mathcal{Y}, [\![\boldsymbol{A}_{\sigma_1}, \ldots, \boldsymbol{A}_{\sigma_{u-1}}, \boldsymbol{A}_{\sigma_u}, \boldsymbol{A}_{\sigma_{u+1}}, \right. \qquad (40)\\ &\qquad \left. \ldots, \boldsymbol{A}_{\sigma_{v-1}}, \boldsymbol{B}, \boldsymbol{A}_{\sigma_{v+1}}, \ldots, \boldsymbol{A}_{\sigma_N}]\!] \right\rangle\\ &\quad = \left\langle \boldsymbol{Y}_{(v)}, \boldsymbol{B} \left(\odot_{j \neq v} \boldsymbol{A}_{\sigma_j}\right)^T \right\rangle\\ &\quad = \left\langle \boldsymbol{B}, \boldsymbol{Y}_{(v)} \left(\odot_{j \neq v} \boldsymbol{A}_{\sigma_j}\right) \right\rangle. \qquad (41) \end{aligned}$$

因此，我们已经证明，对于任意 $\boldsymbol{B} \in \mathbb{R}^{s \times r}$

$$\left\langle \boldsymbol{B}, \boldsymbol{Y}_{(u)} \left(\odot_{j \neq u} \boldsymbol{A}_{\sigma_j}\right) \right\rangle = \left\langle \boldsymbol{B}, \boldsymbol{Y}_{(v)} \left(\odot_{j \neq v} \boldsymbol{A}_{\sigma_j}\right) \right\rangle,$$

这意味着 $\boldsymbol{Y}_{(u)} \left(\odot_{j \neq u} \boldsymbol{A}_{\sigma_j}\right) = \boldsymbol{Y}_{(v)} \left(\odot_{j \neq v} \boldsymbol{A}_{\sigma_j}\right)$，从而完成了证明。 □

附录 D 定理 7 的证明

我们首先证明 $\widetilde{\boldsymbol{Y}}_{(n)} \left(\odot_{j \neq n} \boldsymbol{A}_{\sigma_j}\right) = \widehat{\boldsymbol{Y}}_{(n)} \left(*_{j \neq n} \widehat{\boldsymbol{A}}_{\sigma_j}\right)$。我们先将 Khatri-Rao 积展开为按列的克罗内克积，然后应用矩阵乘法的定义。令 $\left(\otimes_{j \neq n} \left(\boldsymbol{A}_{\sigma_j}\right)_{:,\ell}\right)$ 作为以下内容的简写

$$\left(\boldsymbol{A}_{\sigma_N}\right)_{:,\ell} \otimes \cdots \otimes \left(\boldsymbol{A}_{\sigma_{n+1}}\right)_{:,\ell} \otimes \left(\boldsymbol{A}_{\sigma_{n-1}}\right)_{:,\ell} \otimes \cdots \otimes \left(\boldsymbol{A}_{\sigma_1}\right)_{:,\ell},$$

以及 $\sum_{\substack{i_k=1 \\ k \neq n}}^{I_k}$ 表示.....的简写

$$\sum_{i_1=1}^{I_1} \cdots \sum_{i_{n-1}=1}^{I_{n-1}} \sum_{i_{n+1}=1}^{I_{n+1}} \cdots \sum_{i_N=1}^{I_N},$$

那么，模式-n MTTKRP $\widetilde{\boldsymbol{Y}}_{(n)} \left(\odot_{j \neq n} \boldsymbol{A}_{\sigma_j}\right)$ 的 (u, v) 元素为

$$\begin{aligned} &\left[\widetilde{\boldsymbol{Y}}_{(n)} \left(\odot_{j \neq n} \boldsymbol{A}_{\sigma_j}\right)\right]_{u,v}\\ &= \left[\widetilde{\boldsymbol{Y}}_{(n)} \left[\left(\otimes_{j \neq n} \left(\boldsymbol{A}_{\sigma_j}\right)_{:,1}\right) \quad \cdots \quad \left(\otimes_{j \neq n} \left(\boldsymbol{A}_{\sigma_j}\right)_{:,r}\right)\right]\right]_{u,v}\\ &= \sum_{\substack{i_k=1 \\ k \neq n}}^{I_k} \left(\tilde{y}_{i_1,\ldots,i_{n-1},u,i_{n+1},\ldots,i_N} \left(\prod_{j \neq n} \left(\boldsymbol{A}_{\sigma_j}\right)_{i_j,v}\right)\right) \end{aligned} \tag{42}$$

~由于Y </style}是稀疏的，仅包含 s 个非零元素，因此上述求和可化简为

$$\begin{aligned} &\left[\widetilde{\boldsymbol{Y}}_{(n)} \left(\odot_{j \neq n} \boldsymbol{A}_{\sigma_j}\right)\right]_{u,v}\\ &= \sum_{w=1}^{s} \left(\tilde{y}_{i_1^w,\ldots,i_{n-1}^w,u,i_{n+1}^w,\ldots,i_N^w} \left(\prod_{j \neq n} \left(\boldsymbol{A}_{\sigma_j}\right)_{i_j^w,v}\right)\right) \end{aligned} \tag{43}$$

^现在，若我们将 $\boldsymbol{Y}_{(n)}$ 定义为

$$\widehat{\boldsymbol{Y}}_{(n)} = \begin{bmatrix} \tilde{y}_{i_1^1,\ldots,i_{n-1}^1,1,i_{n+1}^1,\ldots,i_N^1} & \cdots & \tilde{y}_{i_1^s,\ldots,i_{n-1}^s,1,i_{n+1}^s,\ldots,i_N^s} \\ \vdots & \vdots & \vdots \\ \tilde{y}_{i_1^1,\ldots,i_{n-1}^1,I_n,i_{n+1}^1,\ldots,i_N^1} & \cdots & \tilde{y}_{i_1^s,\ldots,i_{n-1}^s,I_n,i_{n+1}^s,\ldots,i_N^s} \end{bmatrix}, \tag{44}$$

^和 $\boldsymbol{A}_{\sigma_j}$ 为

$$\widehat{\boldsymbol{A}}_{\sigma_j} = \begin{bmatrix} -(\boldsymbol{A}_{\sigma_j})_{i_j^1,:}- \\ \vdots \\ -(\boldsymbol{A}_{\sigma_j})_{i_j^s,:}- \end{bmatrix} = \begin{bmatrix} (\boldsymbol{A}_{\sigma_j})_{i_j^1,1} & \cdots & (\boldsymbol{A}_{\sigma_j})_{i_j^1,r} \\ \vdots & \vdots & \vdots \\ (\boldsymbol{A}_{\sigma_j})_{i_j^s,1} & \cdots & (\boldsymbol{A}_{\sigma_j})_{i_j^s,r} \end{bmatrix}, \tag{45}$$

那么我们有

$$\begin{aligned} &\left[\widehat{\boldsymbol{Y}}_{(n)}\left(*_{j\neq n}\widehat{\boldsymbol{A}}_{\sigma_j}\right)\right]_{u,v} \\ &= \left[\widehat{\boldsymbol{Y}}_{(n)}\begin{bmatrix} \prod_{j\neq n}(\boldsymbol{A}_{\sigma_j})_{i_j^1,1} & \cdots & \prod_{j\neq n}(\boldsymbol{A}_{\sigma_j})_{i_j^1,r} \\ \vdots & \vdots & \vdots \\ \prod_{j\neq n}(\boldsymbol{A}_{\sigma_j})_{i_j^s,1} & \cdots & \prod_{j\neq n}(\boldsymbol{A}_{\sigma_j})_{i_j^s,r} \end{bmatrix}\right]_{u,v} \\ &= \sum_{w=1}^{s}\left(\tilde{y}_{i_1^w,\ldots,i_{n-1}^w,u,i_{n+1}^w,\ldots,i_N^w}\left(\prod_{j\neq n}(\boldsymbol{A}_{\sigma_j})_{i_j^w,v}\right)\right) \\ &= \left[\widetilde{\boldsymbol{Y}}_{(n)}\left(\odot_{j\neq n}\boldsymbol{A}_{\sigma_j}\right)\right]_{u,v}. \end{aligned} \tag{46}$$

接下来，我们将证明 $\left(\odot_{j=1}^{N}\boldsymbol{A}_{\sigma_j}\right)^T\mathrm{vec}(\widetilde{\boldsymbol{\mathcal{Y}}}) = \hat{\boldsymbol{y}}\left(*_{j=1}^{N}\widehat{\boldsymbol{A}}_{\sigma_j}\right)$，采用类似的方法。若令

$$\left(\otimes_{j=1}^{n}\left(\boldsymbol{A}_{\sigma_j}\right)_{:,\ell}\right) = (\boldsymbol{A}_N)_{:,\ell}\otimes\cdots\otimes(\boldsymbol{A}_1)_{:,\ell}, \tag{47}$$

则 $(\odot_{j=1}^{N}\boldsymbol{A}_{\sigma_j})^T\mathrm{vec}(\widetilde{\boldsymbol{\mathcal{Y}}})$ 的第 v 个元素为

$$\begin{aligned} &\left[\left(\odot_{j=1}^{N}\boldsymbol{A}_{\sigma_j}\right)^T\mathrm{vec}\left(\widetilde{\boldsymbol{\mathcal{Y}}}\right)\right]_v \\ &= \left[\left[\left(\otimes_{j=1}^{n}\left(\boldsymbol{A}_{\sigma_j}\right)_{:,1}\right)\ \cdots\ \left(\otimes_{j=1}^{n}\left(\boldsymbol{A}_{\sigma_j}\right)_{:,r}\right)\right]\mathrm{vec}\left(\widetilde{\boldsymbol{\mathcal{Y}}}\right)\right]_v \\ &= \sum_{i_1=1}^{I_1}\cdots\sum_{i_N=1}^{I_N}\left(\tilde{y}_{i_1,\ldots,i_N}\left(\prod_{j=1}^{N}(\boldsymbol{A}_{\sigma_j})_{i_j,v}\right)\right) \end{aligned} \tag{48}$$

~由于 $\boldsymbol{\mathcal{Y}}$ 是稀疏的，且仅有 s个非零元素，上述求和可简化为

$$\begin{aligned} &\left[\left(\odot_{j=1}^{N}\boldsymbol{A}_{\sigma_j}\right)^T\mathrm{vec}\left(\widetilde{\boldsymbol{\mathcal{Y}}}\right)\right]_v \\ &\qquad = \sum_{w=1}^{s}\left(\tilde{y}_{i_1^w,\ldots,i_N^w}\left(\prod_{j=1}^{N}(\boldsymbol{A}_{\sigma_j})_{i_j^w,v}\right)\right) \end{aligned} \tag{49}$$

现在，令

$$\hat{\boldsymbol{y}} = \begin{bmatrix} \tilde{y}_{i_1^1,\ldots,i_N^1} \\ \vdots \\ \tilde{y}_{i_1^s,\ldots,i_N^s} \end{bmatrix}^T, \tag{50}$$

^并像之前一样定义 $\boldsymbol{A}_{\sigma_j}$，我们得到

$$\begin{aligned} &\left[\hat{\boldsymbol{y}}\left(*_{j=1}^{N}\widehat{\boldsymbol{A}}_{\sigma_j}\right)\right]_v \\ &= \left[\hat{\boldsymbol{y}}\begin{bmatrix} \prod_{j=1}^{N}(\boldsymbol{A}_{\sigma_j})_{i_j^1,1} & \cdots & \prod_{j=1}^{N}(\boldsymbol{A}_{\sigma_j})_{i_j^1,r} \\ \vdots & \vdots & \vdots \\ \prod_{j=1}^{N}(\boldsymbol{A}_{\sigma_j})_{i_j^s,1} & \cdots & \prod_{j=1}^{N}(\boldsymbol{A}_{\sigma_j})_{i_j^s,r} \end{bmatrix}\right]_v \\ &= \sum_{w=1}^{s}\left(\tilde{y}_{i_1^w,\ldots,i_N^w}\left(\prod_{j=1}^{N}(\boldsymbol{A}_{\sigma_j})_{i_j^w,v}\right)\right) \\ &= \left[\left(\odot_{j=1}^{N}\boldsymbol{A}_{\sigma_j}\right)^T\mathrm{vec}\left(\widetilde{\boldsymbol{\mathcal{Y}}}\right)\right]_v. \end{aligned} \tag{51}$$

□

参考文献

[1] F. L. Hitchcock，“张量或多项式表示为乘积之和”，《数学与物理学杂志》，第 6 卷，第 1–4 期，第 164–189 页，1927 年。[2] J. D. Carroll 和 J.-J. Chang，“通过‘Eckart–Young’分解的 n 路推广分析多维尺度分析中的个体差异”，《心理测量学》，第 35 卷，第 3 期，第 283–319 页，1970 年。[3] R. A. Harshman 等，“PARAFAC过程的基础：‘解释性’多模态因子分析的模型与条件”，加州大学洛杉矶分校语音学工作论文，第 16 卷，第 1 期，第 84 页，1970 年。[4] T. G. Kolda 和 B. W. Bader，“张量分解及其应用”，《*SIAM*评论》，第 51 卷，第 3 期，第 455–500 页，2009 年。[5] A. Cichocki 和 A.-H. Phan，“大规模非负矩阵与张量分解的快速局部算法”，《*IEICE*学报：电子、通信与计算机科学基础》，第 92 卷，第 3 期，第 708–721 页，2009 年。[6] A. Shashua 和 T. Hazan，“非负张量分解及其在统计学和计算机视觉中的应用”，载于第 *22* 届国际机器学习会议论文集，2005 年，第 792–799 页。[7] M. Welling 和 M. Weber，“正张量分解”，《模式识别快报》，第 22 卷，第 12 期，第 1255–1261 页，2001 年。[8] E. C. Chi 和 T. G. Kolda，“关于张量、稀疏性与非负分解”，*SIAM* 矩阵分析与应用期刊，第 33 卷，第 4 期，第 1272–1299 页，2012 年。[9] S. Hansen、T. Plantenga 和 T. G. Kolda，“用于 Kullback–Leibler非负张量分解的基于牛顿法的优化”，《优化方法与软件》，第 30 卷，第 5 期，第 1002–1029 页，2015 年。[10] T. M. Ranadive 和 M. M. Baskaran，“一种用于计数张量的一步式 CP 分解方法”，载于*2021* 年 *IEEE* 高性能极限计算会议（*HPEC*）。IEEE，2021 年，第 1–8 页。[11] P. Miettinen，“布尔张量分解”，载于*2011* 年 *IEEE* 第 *11* 届国际数据挖掘会议。IEEE，2011 年，第 447–456 页。[12] M. Wang 和 L. Li，“从二值多路数据中学习：概率张量分解及其统计最优性”，《机器学习研究杂志》，第 21 卷，第 154 期，第 1–38 页，2020 年。[13] D. Hong、T. G. Kolda 和 J. A. Duersch，“广义典型多项式张量分解”，《*SIAM*评论》，第 62 卷，第 1 期，第 133–163 页，2020 年 1 月。 [在线]。可查阅：http://dx.doi.org/113.7/ 18M1203626[14]T. G. Kolda 和 D. Hong，“大规模张量分解的随机梯度”，《*SIAM*数据科学数学杂志》，第 2 卷，第 4 期，第 1066–1095 页，2020 年 1 月。 [在线]。可查阅：http://dx.doi.org/ 10 1137 19 1266265 。/ M W. [15] Pu、S. Ibrahim、X. Fu 和 M. Hong，“非欧氏损失下低秩张量分解的随机镜像下降”，*IEEE* 信号处理汇刊，第 70 卷，第 1803–1818 页，2022 年。[16] Z. Liu、Q. Wang、C. Cui 和 Y. Xia，“用于大规模广义张量 CP 分解的惯性加速随机镜像下降”，《计算优化与应用》，第 91 卷，第 1 期，第 201–233 页，2025 年。[17] M. Vandecappelle、N. Vervliet 和 L. De Lathauwer，“一种用于拟合非最小二乘代价典型多项式分解的二阶方法”，*IEEE* 信号处理汇刊，第 68 卷，第 4454–4465 页，2020 年。

[18] T. G. Kolda，“对称张量分解的数值优化”，*Mathematical Programming*，第 151 卷，第 1 期，第 225–248 页，4 月 2015 [ ] 10 1007 10107 015 0895 0 。在线。[19] 可查阅：http://dx.doi.org/ 。/s - - - S. Sherman 和 T. G. Kolda，“利用对称张量分解估计高阶矩”，*SIAM* 矩阵分析与应用期刊，第 41 卷，第 3 期，第 1369–1387 页，2020 年。[20] J. Kileel 和 J. M. Pereira，“用于对称张量分解和广义主成分分析的子空间幂方法”，*arXiv*预印本 *arXiv:1912.04007*，第 3 卷，2019 年。[21] A.-H. Phan、P. Tichavsk`y 和 A. Cichocki，“用于高阶 CANDECOMP/PARAFAC 张量分解的快速交替最小二乘算法”，*IEEE* 信号处理汇刊，第 61 卷，第 19 期，第 4834–4846 页，2013 年。[22] B. W. Bader 和 T. G. Kolda，“使用稀疏和因子化张量进行高效 MATLAB 计算”，*SIAM* 科学计算期刊，第 30 卷，第 1 期，第 205–231 页，2008 年。[23] R. H. Byrd、P. Lu、J. Nocedal 和 C. Zhu，“一种用于边界约束优化的有限记忆算法”，*SIAM* 科学计算期刊，第 16 卷，第 5 期，第 1190–1208 页，1995 年 9 月。 [在线]。可查阅：http://dx.doi.org/ 10 1137 0916069 。/ D. P. Kingma 和 J. Ba，[24] “Adam：一种随机优化方法”，*arXiv*预印本 *arXiv:1412.6980*，2014 年。[25] S. Vyas、N. Even-Chen、S. D. Stavisky、S. I. Ryu、P. Nuyujukian 和 K. V. Shenoy，“运动学习迁移背后的神经群体动力学”，*Neuron*，第 97 卷，第 5 期，第 1177–1186 页，2018 年。[26] S. Vyas、D. J. O’Shea、S. I. Ryu 和 K. V. Shenoy，“运动准备在误差驱动学习中的因果作用”，*Neuron*，第 106 卷，第 2 期，第 329–339 页，2020 年。[27] T. G. Kolda，“猴子 BMI 张量数据集”，https://gitlab.com/tensors/ tensor data monkey bmi，2021 年，访问日期：2025-07-21。[28] T. Opsahl 和 P. Panzarasa，“加权网络中的聚类”，*Social Networks*，第 31 卷，第 2 期，第 155–163 页，2009 年。