张量笔记

Notes of Tensor

Zebediah

2024 年 5 月 13 日

目录

Chapter 1

张量计算

1.1 记号

标量、向量、矩阵和张量分别用 $a, \mathbf{a}, \mathbf{A}, \mathcal{A}$ 表示, 后三者的元素分别写作 $\mathbf{a}(i), \mathbf{A}(i,j), \mathcal{A}(i_1, i_2, \cdots, i_N)$ 或 $a_i, a_{i,j}, a_{i_1,i_2,\cdots,i_N}$.

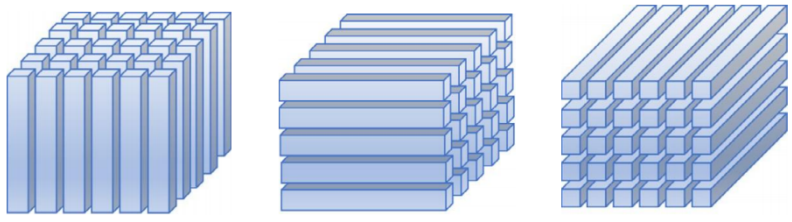
固定张量 $\mathcal{A}$ 的某些模数可以产生一些子张量, 例如纤维或切片. 具体地, 对于一个三阶张量 $\mathcal{A} \in \mathbb{R}^{3\times3\times3}$, 它的 frontal slices, horizontal slices, lateral slices 分别为

$$\mathcal{A}(:,:,1) = \begin{bmatrix} 1 & 14 & 15 \\ 23 & 6 & 20 \\ 24 & 18 & 8 \end{bmatrix}, \mathcal{A}(:,:,2) = \begin{bmatrix} 15 & 8 & 7 \\ 28 & 12 & 17 \\ 21 & 29 & 23 \end{bmatrix}, \mathcal{A}(:,:,3) = \begin{bmatrix} 9 & 5 & 13 \\ 7 & 22 & 26 \\ 21 & 1 & 19 \end{bmatrix}, \tag{1.1.1}$$

$$\mathcal{A}(1,:,:) = \begin{bmatrix} 1 & 14 & 15 \\ 15 & 8 & 7 \\ 9 & 5 & 13 \end{bmatrix}, \mathcal{A}(2,:,:) = \begin{bmatrix} 23 & 6 & 20 \\ 28 & 12 & 17 \\ 7 & 22 & 26 \end{bmatrix}, \mathcal{A}(3,:,:) = \begin{bmatrix} 24 & 18 & 8 \\ 21 & 29 & 23 \\ 21 & 1 & 19 \end{bmatrix}, \tag{1.1.2}$$

$$\mathcal{A}(:,1,:) = \begin{bmatrix} 1 & 23 & 24 \\ 15 & 28 & 21 \\ 9 & 7 & 21 \end{bmatrix}, \mathcal{A}(:,2,:) = \begin{bmatrix} 14 & 6 & 18 \\ 8 & 12 & 29 \\ 5 & 22 & 1 \end{bmatrix}, \mathcal{A}(:,3,:) = \begin{bmatrix} 15 & 20 & 8 \\ 7 & 17 & 23 \\ 13 & 26 & 19 \end{bmatrix}, \tag{1.1.3}$$

而其中一个 mode-1 纤维 $\mathcal{A}(:,2,2) = [8,12,29]^{\mathrm{T}}$, mode-2 纤维 $\mathcal{A}(2,:,2) = [28,12,17]^{\mathrm{T}}$, mode-3 纤维 $\mathcal{A}(2,2,:) = [6,12,22]^{\mathrm{T}}$. 三阶张量的纤维和切片示意图如图 1.1 和图 1.2 所示.



(a) Mode-1 (column) fibers **(b)** Mode-2 (row) fibers **(c)** Mode-3 (tube) fibers

图 1.1: 张量的纤维

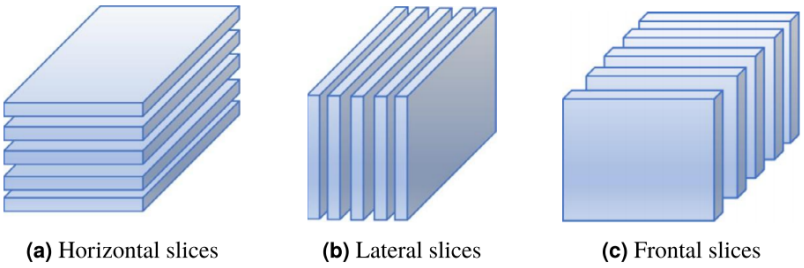
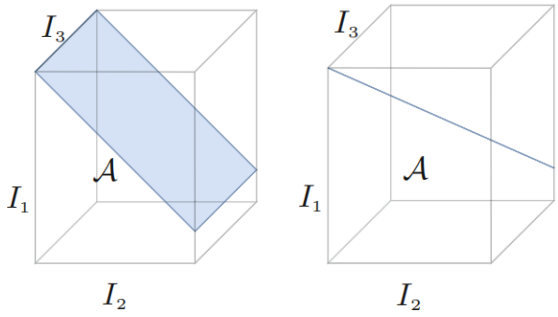


图 1.2: 张量的切片

定义 1.1 (*f*-对角张量和对角张量) 它们的元素如图 1.3 所示:



(a) f-diagonal tensor (b) diagonal tensor

图 1.3: f-对角张量和对角张量

1.2 基本矩阵计算

首先对于矩阵 A, 其转置、共轭转置、逆和 Moore-Penrose 广义逆分别记为 $\mathbf{A}^{\mathrm{T}}, \mathbf{A}^{\mathrm{H}}, \mathbf{A}^{-1}, \mathbf{A}^{\dagger}$.

定义 1.2 (矩阵的迹) 矩阵 $\mathbf{A} \in \mathbb{R}^{I \times I}$ 的迹定义为 $\mathrm{tr}(\mathbf{A}) = \sum_{i=1}^{I} \mathbf{A}(i,i)$.

定义 1.3 (矩阵内积) 相同大小的矩阵 $\mathbf{A}$ 和 $\mathbf{B}$ 的内积定义为

$$c = \langle \mathbf{A}, \mathbf{B} \rangle = \mathrm{vec}(\mathbf{A})^{\mathrm{T}} \mathrm{vec}(\mathbf{B}) \in \mathbb{R}, \tag{1.2.1}$$

其中 $\mathrm{vec}(\cdot)$ 表示沿列向量化.

定义 1.4 (Frobenius 范数) 矩阵 $\mathbf{A}$ 的 Frobenius 范数表示为 $\|\mathbf{A}\|_{\mathrm{F}} = \sqrt{\langle \mathbf{A}, \mathbf{A} \rangle} \in \mathbb{R}$, 亦称为 F 范数.

定义 1.5 (向量外积) 对于 $\mathbf{a} \in \mathbb{R}^{I}$ 和 $\mathbf{b} \in \mathbb{R}^{J}$, 外积定义为

$$\mathbf{C} = \mathbf{a} \circ \mathbf{b} \in \mathbb{R}^{I \times J}. \tag{1.2.2}$$

类似地, 对于向量 $\mathbf{a}_n \in \mathbb{R}^{I_n}, n = 1, \cdots, N$, 向量的外积将得到一个多维张量:

$$\mathcal{C} = \mathbf{a}_1 \circ \mathbf{a}_2 \circ \cdots \circ \mathbf{a}_N \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}. \tag{1.2.3}$$

定义 1.6 (Hadamard 积) 对于具有相同大小 $I \times J$ 的矩阵 $\mathbf{A}$ 和 $\mathbf{B}$, 其 Hadamard 积定义为

$$\begin{aligned}\mathbf{C} &= \mathbf{A} \circledast \mathbf{B} \in \mathbb{R}^{I\times J} \\ &= \begin{bmatrix} a_{1,1}b_{1,1} & a_{1,2}b_{1,2} & \cdots & a_{1,J}b_{1,J} \\ a_{2,1}b_{2,1} & a_{2,2}b_{2,2} & \cdots & a_{2,J}b_{2,J} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I,1}b_{I,1} & a_{I,2}b_{I,2} & \cdots & a_{I,J}b_{I,J} \end{bmatrix}.\end{aligned} \tag{1.2.4}$$

定义 1.7 (矩阵串联) $[\mathbf{A}, \mathbf{B}] \in \mathbb{R}^{I\times(J+K)}$ 表示矩阵 $\mathbf{A} \in \mathbb{R}^{I\times J}$ 和 $\mathbf{B} \in \mathbb{R}^{I\times K}$ 的按列串联, 而 $[\mathbf{A}; \mathbf{B}] \in \mathbb{R}^{(I+J)\times K}$ 表示矩阵 $\mathbf{A} \in \mathbb{R}^{I\times K}$ 和 $\mathbf{B} \in \mathbb{R}^{J\times K}$ 的按行串联.

定义 1.8 (Kronecker 积) 矩阵 $\mathbf{A} \in \mathbb{R}^{I\times J}$ 和 $\mathbf{B} \in \mathbb{R}^{K\times L}$ 的 Kronecker 积定义为

$$\begin{aligned}\mathbf{C} &= \mathbf{A} \otimes \mathbf{B} \in \mathbb{R}^{IK\times JL} \\ &= \begin{bmatrix} a_{1,1}\mathbf{B} & a_{1,2}\mathbf{B} & \cdots & a_{1,J}\mathbf{B} \\ a_{2,1}\mathbf{B} & a_{2,2}\mathbf{B} & \cdots & a_{2,J}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I,1}\mathbf{B} & a_{I,2}\mathbf{B} & \cdots & a_{I,J}\mathbf{B} \end{bmatrix}.\end{aligned} \tag{1.2.5}$$

定理 1.1 对于任意矩阵 $\mathbf{A}$、$\mathbf{B}$、$\mathbf{C}$ 和 $\mathbf{D}$, Kronecker 积具有以下一些有用的性质:

$$(\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C} = \mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C}), \tag{1.2.6}$$

$$\mathbf{AC} \otimes \mathbf{BD} = (\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}), \tag{1.2.7}$$

$$(\mathbf{A} \otimes \mathbf{C})^{\mathrm{T}} = \mathbf{A}^{\mathrm{T}} \otimes \mathbf{C}^{\mathrm{T}}, \tag{1.2.8}$$

$$(\mathbf{A} \otimes \mathbf{C})^{\dagger} = \mathbf{A}^{\dagger} \otimes \mathbf{C}^{\dagger}, \tag{1.2.9}$$

$$\mathrm{vec}(\mathbf{ABC}) = \left(\mathbf{C}^{\mathrm{T}} \otimes \mathbf{A}\right) \mathrm{vec}(\mathbf{B}), \tag{1.2.10}$$

其中, (1.2.10) 经常用于处理形式为 $\min_{\mathbf{X}} \|\mathbf{Y} - \mathbf{AXB}\|_{\mathbf{F}}^2$ 的线性最小二乘问题.

定义 1.9 (Khatri-Rao 积) 矩阵 $\mathbf{A} \in \mathbb{R}^{I\times K}$ 和 $\mathbf{B} \in \mathbb{R}^{J\times K}$ 的 Khatri-Rao 积定义为

$$\mathbf{A} \odot \mathbf{B} = [\mathbf{a}_1 \otimes \mathbf{b}_1, \mathbf{a}_2 \otimes \mathbf{b}_2, \cdots, \mathbf{a}_K \otimes \mathbf{b}_K] \in \mathbb{R}^{IJ\times K}, \tag{1.2.11}$$

其中 $\mathbf{a}_k$ 和 $\mathbf{b}_k$ 是矩阵 $\mathbf{A}$ 和 $\mathbf{B}$ 的第 k 列.

例如, 根据定义, 我们可以首先得到矩阵 $\mathbf{A}$ 和 $\mathbf{B}$ 的列为

$$\mathbf{a}_1 = \begin{bmatrix} 2 \\ 2 \end{bmatrix}; \mathbf{a}_2 = \begin{bmatrix} 6 \\ 1 \end{bmatrix}; \mathbf{b}_1 = \begin{bmatrix} 1 \\ 3 \end{bmatrix}; \mathbf{b}_2 = \begin{bmatrix} 5 \\ 8 \end{bmatrix}.$$

则矩阵 $\mathbf{A}$ 和 $\mathbf{B}$ 的 Khatri-Rao 乘积可以通过以下方式计算:

$$\mathbf{A}\odot\mathbf{B}=[\mathbf{a}_1\otimes\mathbf{b}_1,\mathbf{a}_2\otimes\mathbf{b}_2]=\begin{bmatrix}2\mathbf{b}_1 & 6\mathbf{b}_2\\ 2\mathbf{b}_1 & \mathbf{b}_2\end{bmatrix}=\begin{bmatrix}2 & 30\\ 6 & 48\\ 2 & 5\\ 6 & 8\end{bmatrix}.$$

定理 1.2 对于任意矩阵 $\mathbf{A},\mathbf{B},\mathbf{C}$ 和 $\mathbf{D}$, 我们有

$$(\mathbf{A}\odot\mathbf{B})\odot\mathbf{C}=\mathbf{A}\odot(\mathbf{B}\odot\mathbf{C}), \tag{1.2.12}$$

$$(\mathbf{A}\odot\mathbf{C})^{\mathrm{T}}(\mathbf{A}\odot\mathbf{C})=\mathbf{A}^{\mathrm{T}}\mathbf{A}\circledast\mathbf{B}^{\mathrm{T}}\mathbf{B}, \tag{1.2.13}$$

$$(\mathbf{A}\otimes\mathbf{B})(\mathbf{C}\odot\mathbf{D})=\mathbf{A}C\odot\mathbf{B}\mathbf{D}, \tag{1.2.14}$$

并且当 $\mathbf{D}=\mathrm{diag}(\mathbf{d}),\mathbf{d}\in\mathbb{R}^I$ 时, 即 $\mathbf{D}(i,i)=\mathbf{d}(i),i=1,\cdots,I$, 成立

$$\mathrm{vec}(\mathbf{ADC})=\left(\mathbf{C}^{\mathrm{T}}\odot\mathbf{A}\right)\mathbf{d}. \tag{1.2.15}$$

定义 1.10 (卷积) 矩阵 $\mathbf{A}\in\mathbb{R}^{I_1\times I_2}$ 和 $\mathbf{B}\in\mathbb{R}^{J_1\times J_2}$ 的卷积定义为

$$\mathbf{C}=\mathbf{A}\boxtimes\mathbf{B}\in\mathbb{R}^{K_1\times K_2}, \tag{1.2.16}$$

其中 $K_n=I_n+J_n-1,n=1,2$. $\mathbf{C}$ 的元素满足

$$\mathbf{C}\left(k_1,k_2\right)=\sum_{j_1,j_2}\mathbf{B}\left(j_1,j_2\right)\mathbf{A}\left(k_1-j_1+1,k_2-j_2+1\right), \tag{1.2.17}$$

其中 $k_n=1,\cdots,K_n,n=1,2$.

定义 1.11 (ℓ_p 范数) 对于矩阵 $\mathbf{A}\in\mathbb{R}^{I\times J}$, 其 ℓ_p 定义为

$$\|\mathbf{A}\|_p=\left(\sum_{i,j}a_{i,j}^p\right)^{1/p}. \tag{1.2.18}$$

定义 1.12 (最大范数) 矩阵 $\mathbf{A}\in\mathbb{R}^{I\times J}$ 的最大范数定义为

$$\|\mathbf{A}\|_\infty=\max\left\{\left|a_{i,j}\right|,1\le i\le I,1\le j\le J\right\}. \tag{1.2.19}$$

定义 1.13 (核范数) 矩阵 $\mathbf{A}\in\mathbb{R}^{I\times J}$ 的核范数定义为

$$\|\mathbf{A}\|_*=\sum_{k=1}^{\min(I,J)}\sigma_k, \tag{1.2.20}$$

其中 σ_k 是矩阵 $\mathbf{A}$ 的第 k 个奇异值.

1.3 张量展开

1.3.1 mode-n 展开

定义 1.14 (mode-n 展开) 对于张量 $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, 其 mode-$n$ 展开矩阵表示为 $\mathbf{A}_{(n)}$ 或 $\mathbf{A}_{[n]}$. mode-n 展开是将张量的第 n 个模数作为行, 而其余作为列来进行构造:

$$\begin{aligned}\mathbf{A}_{(n)}\left(i_n, j_1\right) &= \mathcal{A}\left(i_1, \cdots, i_n, \cdots, i_N\right)\\ \mathbf{A}_{[n]}\left(i_n, j_2\right) &= \mathcal{A}\left(i_1, \cdots, i_n, \cdots, i_N\right),\end{aligned}$$

其中 $j_1 = \overline{i_1, \cdots, i_{n-1}, i_{n+1}, \cdots, i_N}, j_2 = \overline{i_{n+1}, \cdots, i_N, i_1, \cdots, i_{n-1}}$ 是多重索引.

注 多重索引 $i = \overline{i_1, \cdots, i_N}$ 可以是:

- 反向字典序 (小端序) 的形式:

$$\overline{i_1, \cdots, i_N} = i_1 + \left(i_2 - 1\right) I_1 + \left(i_3 - 1\right) I_1 I_2 + \cdots + \left(i_N - 1\right) I_1 \cdots I_{N-1}.$$

- 或正向字典序 (大端序) 的形式:

$$\overline{i_1, \cdots, i_N} = i_N + \left(i_{N-1} - 1\right) I_N + \cdots + \left(i_2 - 1\right) I_3 \cdots I_N + \left(i_1 - 1\right) I_2 \cdots I_N.$$

在接下来的内容里, 除非另有说明, 否则我们将使用小端序表示法.

$\mathbf{A}_{(n)}$ 和 $\mathbf{A}_{[n]}$ 之间唯一的区别在于列中剩余维度的顺序. 我们以之前举例的张量进行说明:

$$\mathcal{A}(:,:,1) = \begin{bmatrix} 1 & 14 & 15 \\ 23 & 6 & 20 \\ 24 & 18 & 8 \end{bmatrix}, \mathcal{A}(:,:,2) = \begin{bmatrix} 15 & 8 & 7 \\ 28 & 12 & 17 \\ 21 & 29 & 23 \end{bmatrix}, \mathcal{A}(:,:,3) = \begin{bmatrix} 9 & 5 & 13 \\ 7 & 22 & 26 \\ 21 & 1 & 19 \end{bmatrix}.$$

我们有

$$\mathbf{A}_{(1)} = \mathbf{A}_{[1]} = \left[\begin{array}{ccc|ccc|ccc} 1 & 14 & 15 & 15 & 8 & 7 & 9 & 5 & 13 \\ 23 & 6 & 20 & 28 & 12 & 17 & 7 & 22 & 26 \\ 24 & 18 & 8 & 21 & 29 & 23 & 21 & 1 & 19 \end{array}\right],$$

$$\mathbf{A}_{(2)} = \left[\begin{array}{ccc|ccc|ccc} 1 & 23 & 24 & 15 & 28 & 21 & 9 & 7 & 21 \\ 14 & 6 & 18 & 8 & 12 & 29 & 5 & 22 & 1 \\ 15 & 20 & 8 & 7 & 17 & 23 & 13 & 26 & 19 \end{array}\right],$$

$$\mathbf{A}_{[2]} = \left[\begin{array}{ccc|ccc|ccc} 1 & 15 & 9 & 23 & 28 & 7 & 24 & 21 & 21 \\ 14 & 8 & 5 & 6 & 12 & 22 & 18 & 29 & 1 \\ 15 & 7 & 13 & 20 & 17 & 26 & 8 & 23 & 19 \end{array}\right],$$

$$\mathbf{A}_{(3)} = \mathbf{A}_{[3]} = \left[\begin{array}{ccc|ccc|ccc} 1 & 23 & 24 & 14 & 6 & 18 & 15 & 20 & 8 \\ 15 & 28 & 21 & 8 & 12 & 29 & 7 & 17 & 23 \\ 9 & 7 & 21 & 5 & 22 & 1 & 13 & 26 & 19 \end{array}\right].$$

1.3.2 mode-n_1n_2 展开

定义 1.15 (mode-n_1n_2 展开) 对于张量 $\mathcal{A}\in\mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$, 其 mode-$n_1n_2$ 展开张量表示为 $\mathcal{A}_{(n_1n_2)}\in\mathbb{R}^{I_{n_1}\times I_{n_2}\times\prod_{s\neq n_1,n_2}I_s}$, 其前向切片是 $\mathcal{A}$ 的 mode-n_1n_2 切片的字典序排列. 我们有

$$\mathcal{A}_{(n_1n_2)}\left(i_{n_1},i_{n_2},j\right)=\mathcal{A}\left(i_1,i_2,\cdots,i_N\right),$$

它们之间的元素映射关系为

$$j=1+\sum_{\substack{s=1\\ s\neq n_1,s\neq n_2}}^{N}\left(i_s-1\right)J_s \text{ with } J_s=\prod_{\substack{m=1\\ m\neq n_1,m\neq n_2}}^{s-1}I_m.$$

1.3.3 n-展开

定义 1.16 (n-展开) n-展开将原始张量直接重塑为一个矩阵, 使得前 n 个模数作为行, 而其余模数作为列. 对于一个 N 阶张量 $\mathcal{A}\in\mathbb{R}^{I_1\times\cdots\times I_N}$, 其 n-展开矩阵可以表示为 $\mathbf{A}_{\langle n\rangle}$, 其元素满足:

$$\mathbf{A}_{\langle n\rangle}\left(j_1,j_2\right)=\mathcal{A}\left(i_1,\cdots,i_n,\cdots,i_N\right),$$

其中 $j_1=\overline{i_1,\cdots,i_n}, j_2=\overline{i_{n+1},\cdots,i_N}$.

例如之前的例子中,

$$\mathbf{A}_{\langle 1\rangle}=\mathbf{A}_{[1]}=\left[\begin{array}{ccc|ccc|ccc}1&14&15&15&8&7&9&5&13\\23&6&20&28&12&17&7&22&26\\24&18&8&21&29&23&21&1&19\end{array}\right],$$

$$\mathbf{A}_{\langle 2\rangle}=\mathbf{A}_{[3]}{}^{\mathrm{T}}=\left[\begin{array}{ccc}1&15&9\\23&28&7\\24&21&21\\14&8&5\\6&12&22\\18&29&1\\15&7&13\\20&17&26\\8&23&19\end{array}\right],$$

$$\mathbf{A}_{\langle 3\rangle}=\operatorname{vec}(\mathcal{A})=\operatorname{vec}\left(\mathbf{A}_{[1]}\right).$$

1.4 张量积

定义 1.17 (张量内积) 设张量 $\mathcal{A}$ 和 $\mathcal{B}$ 具有相同的大小, 其内积定义为

$$c=\langle\mathcal{A},\mathcal{B}\rangle=\langle\operatorname{vec}(\mathcal{A}),\operatorname{vec}(\mathcal{B})\rangle\in\mathbb{R},\tag{1.4.1}$$

其中 $\operatorname{vec}(\mathcal{A})=\operatorname{vec}\left(\mathbf{A}_{(1)}\right)$.

定义 1.18 (张量外积) 对于两个张量 $\mathcal{A} \in \mathbb{R}^{i_1 \times \cdots \times i_N}$ 和 $\mathcal{B} \in \mathbb{R}^{j_1 \times \cdots \times j_M}$, 张量外积的定义为

$$\mathcal{C} = \mathcal{A} \circ \mathcal{B} \in \mathbb{R}^{i_1 \times \cdots \times i_N \times j_1 \times \cdots \times j_M}, \tag{1.4.2}$$

其元素表示为

$$\mathcal{C}(i_1, \cdots, i_N, j_1, \cdots, j_M) = \mathcal{A}(i_1, \cdots, i_N)\mathcal{B}(j_1, \cdots, j_M). \tag{1.4.3}$$

定义 1.19 (mode-n 乘积) 张量 $\mathcal{A} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ 和矩阵 $\mathbf{B} \in \mathbb{R}^{J \times I_n}$ 的 mode-n 乘积定义为

$$\mathcal{C} = \mathcal{A} \times_n \mathbf{B} \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N}, \tag{1.4.4}$$

其元素表示为

$$\mathcal{C}\left(i_1, \cdots, j, \cdots, i_N\right) = \sum_{i_n=1}^{I_n} \mathcal{A}\left(i_1, \cdots, i_n, \cdots, i_N\right) \mathbf{B}\left(j, i_n\right). \tag{1.4.5}$$

mode-n 具有如下性质:
(a) $(\mathcal{A} \times_n \mathbf{B}) \times_m \mathbf{C} = (\mathcal{A} \times_m \mathbf{C}) \times_n \mathbf{B} = \mathcal{A} \times_n \mathbf{B} \times_m \mathbf{C}$;
(b) $(\mathcal{A} \times_n \mathbf{B}) \times_n \mathbf{C} = \mathcal{A} \times_n (\mathbf{C} \cdot \mathbf{B})$;
(c) $\mathcal{C} = \mathcal{A} \times_n \mathbf{B} \Leftrightarrow \mathbf{C}_{(n)} = \mathbf{B}\mathbf{A}_{(n)}$;
(d) $\mathcal{X} = \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_N \mathbf{U}^{(N)} \Leftrightarrow \mathbf{X}_{(n)} = \mathbf{U}^{(n)} \mathbf{G}_{(n)} \left(\mathbf{U}^{(N)} \otimes \cdots \otimes \mathbf{U}^{(n+1)} \otimes \mathbf{U}^{(n-1)} \otimes \cdots \otimes \mathbf{U}^{(1)}\right)^{\mathrm{T}}$.

定义 1.20 (t-乘积) 张量 $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ 和 $\mathcal{B} \in \mathbb{R}^{I_2 \times I_4 \times I_3}$ 的 t-乘积定义为

$$\mathcal{C} = \mathcal{A} * \mathcal{B} \in \mathbb{R}^{I_1 \times I_4 \times I_3}, \tag{1.4.6}$$

其元素满足

$$\mathcal{C}\left(i_1, i_4, :\right) = \sum_{i_2=1}^{I_2} \mathcal{A}\left(i_1, i_2, :\right) \boxtimes_{I_3} \mathcal{B}\left(i_2, i_4, :\right). \tag{1.4.7}$$

Chapter 2

张量分解

2.1 CP 分解

定义 2.1 (Canonical Polyadic(CP) 分解) 给定一个 N 阶张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, CP 分解定义为

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{u}_r^{(1)} \circ \mathbf{u}_r^{(2)} \circ \cdots \circ \mathbf{u}_r^{(N)} = [\![\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \cdots, \mathbf{U}^{(N)}]\!], \tag{2.1.1}$$

或者逐元素地表示为

$$x_{i_1,i_2,\cdots,i_N} = \sum_{r=1}^{R} u_{i_1,r}^{(1)} u_{i_2,r}^{(2)} \cdots u_{i_N,r}^{(N)}, \tag{2.1.2}$$

其中 $\mathbf{U}^{(n)} = \left[\mathbf{u}_1^{(n)}, \mathbf{u}_2^{(n)}, \cdots \mathbf{u}_R^{(n)}\right] \in \mathbb{R}^{I_n \times R}, n = 1, \cdots, N$ 是因子矩阵.

例如, 一个三阶张量的 CP 分解如下:

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}_r \circ \mathbf{b}_r \circ \mathbf{c}_r = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]. \tag{2.1.3}$$

在这种情况下, mode-n 矩阵形式为

$$\mathbf{X}_{(1)} = \mathbf{A}(\mathbf{C} \odot \mathbf{B})^{\mathrm{T}}, \mathbf{X}_{(2)} = \mathbf{B}(\mathbf{A} \odot \mathbf{C})^{\mathrm{T}}, \mathbf{X}_{(3)} = \mathbf{C}(\mathbf{B} \odot \mathbf{A})^{\mathrm{T}}. \tag{2.1.4}$$

图 2.1 给出了一个三阶张量的 CP 分解的图形表示.

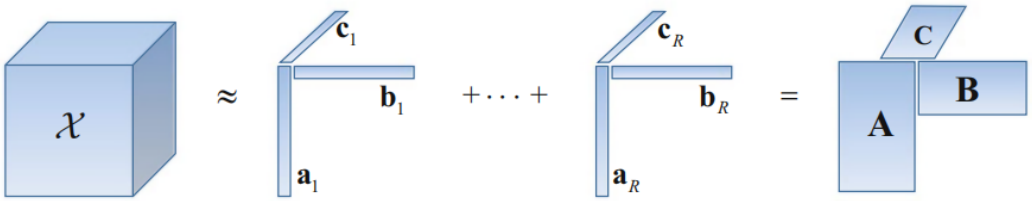


图 2.1: 三阶张量的 CP 分解

2.1.1 张量的秩

张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ 的秩记为 rank($\mathcal{X}$), 其定义为 CP 分解中的秩-1 张量的最小数量, 如下所示:

$$\mathcal{X} = \sum_{r=1}^{R} \mathcal{U}_r,$$

其中 $\mathcal{U}_r$ 是一个秩-1 张量, 可以表示为 $\mathcal{U}_r = \mathbf{u}_r^{(1)} \circ \mathbf{u}_r^{(2)} \circ \cdots \circ \mathbf{u}_r^{(N)}$, R 是使等式成立的最小值. 不同于矩阵的秩, 一个实值张量的秩在 $\mathbb{R}$ 上和 $\mathbb{C}$ 上可能是不同的.

2.1.2 CPD 分解

对于给定的张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, CP 分解的目标是找到最优的因子矩阵, 以预定义的张量秩来近似 $\mathcal{X}$, 其形式如下:

$$\min_{\mathbf{A},\mathbf{B},\mathbf{C}} \|\mathcal{X} - [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]\|_{\mathrm{F}}^2. \tag{2.1.5}$$

直接解决这个问题是困难的, 交替最小二乘 (ALS) 方法是解决这个问题的常用方法. ALS 方法固定 $\mathbf{B}$ 和 $\mathbf{C}$ 来求解 $\mathbf{A}$, 然后固定 $\mathbf{A}$ 和 $\mathbf{C}$ 来求解 $\mathbf{B}$, 接着固定 $\mathbf{A}$ 和 $\mathbf{B}$ 来求解 $\mathbf{C}$, 并持续重复整个过程, 直到满足某个收敛标准. 因此, 问题 (2.1.5) 可以分解为三个子问题, 如下所示:

$$\begin{aligned}
&\min_{\mathbf{A}} \left\|\mathbf{X}_{(1)} - \mathbf{A}(\mathbf{C} \odot \mathbf{B})^{\mathrm{T}}\right\|_{\mathrm{F}}^2, \\
&\min_{\mathbf{B}} \left\|\mathbf{X}_{(2)} - \mathbf{B}(\mathbf{A} \odot \mathbf{C})^{\mathrm{T}}\right\|_{\mathrm{F}}^2, \\
&\min_{\mathbf{C}} \left\|\mathbf{X}_{(3)} - \mathbf{C}(\mathbf{B} \odot \mathbf{A})^{\mathrm{T}}\right\|_{\mathrm{F}}^2.
\end{aligned} \tag{2.1.6}$$

问题 (2.1.6) 的解如下所示:

$$\begin{aligned}
\mathbf{A} &= \mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B}) \left(\mathbf{C}^{\mathrm{T}}\mathbf{C} \circledast \mathbf{B}^{\mathrm{T}}\mathbf{B}\right)^{\dagger}, \\
\mathbf{B} &= \mathbf{X}_{(2)}(\mathbf{A} \odot \mathbf{C}) \left(\mathbf{A}^{\mathrm{T}}\mathbf{A} \circledast \mathbf{C}^{\mathrm{T}}\mathbf{C}\right)^{\dagger}, \\
\mathbf{C} &= \mathbf{X}_{(3)}(\mathbf{B} \odot \mathbf{A}) \left(\mathbf{B}^{\mathrm{T}}\mathbf{B} \circledast \mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{\dagger}.
\end{aligned} \tag{2.1.7}$$

算法的细节如下算法 1 所示.

Algorithm 1 ALS for CP decomposition of a third-order tensor

1: **Input:** $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ and CP-rank R
2: **Initialize** $\mathbf{A}, \mathbf{B}, \mathbf{C}$
3: **repeat**
4: $\quad \mathbf{A} = \mathbf{X}_{(1)}(\mathbf{C} \odot \mathbf{B}) \left(\mathbf{C}^{\mathrm{T}}\mathbf{C} \circledast \mathbf{B}^{\mathrm{T}}\mathbf{B}\right)^{\dagger}$
5: $\quad \mathbf{B} = \mathbf{X}_{(2)}(\mathbf{A} \odot \mathbf{C}) \left(\mathbf{A}^{\mathrm{T}}\mathbf{A} \circledast \mathbf{C}^{\mathrm{T}}\mathbf{C}\right)^{\dagger}$
6: $\quad \mathbf{C} = \mathbf{X}_{(3)}(\mathbf{B} \odot \mathbf{A}) \left(\mathbf{B}^{\mathrm{T}}\mathbf{B} \circledast \mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{\dagger}$
7: **until** fit ceases to improve or maximum iterations exhausted
8: **Output:** $\mathbf{A}, \mathbf{B}, \mathbf{C}$

ALS 方法易于理解和实现, 但可能需要许多迭代才能收敛, 此外也不能保证收敛到全局最小值, 只能保证目标函数 (2.1.5) 减小. 为了解决这个问题, 可以使用一种用于 CP 分解的正则化交替最小二乘方法

(CP-RALS). 此方法中, 第 $(k+1)$ 次迭代中的子问题可以写成如下形式:

$$\begin{aligned}
\mathbf{A}^{k+1} &= \underset{\mathbf{A}}{\operatorname{argmin}} \left\| \mathbf{X}_{(1)} - \mathbf{A}\left(\mathbf{C}^k \odot \mathbf{B}^k\right)^{\mathrm{T}} \right\|_{\mathrm{F}}^2 + \tau \left\| \mathbf{A} - \mathbf{A}^k \right\|_{\mathrm{F}}^2, \\
\mathbf{B}^{k+1} &= \underset{\mathbf{B}}{\operatorname{argmin}} \left\| \mathbf{X}_{(2)} - \mathbf{B}\left(\mathbf{A}^{k+1} \odot \mathbf{C}^k\right)^{\mathrm{T}} \right\|_{\mathrm{F}}^2 + \tau \left\| \mathbf{B} - \mathbf{B}^k \right\|_{\mathrm{F}}^2, \\
\mathbf{C}^{k+1} &= \underset{\mathbf{C}}{\operatorname{argmin}} \left\| \mathbf{X}_{(3)} - \mathbf{C}\left(\mathbf{B}^{k+1} \odot \mathbf{A}^{k+1}\right)^{\mathrm{T}} \right\|_{\mathrm{F}}^2 + \tau \left\| \mathbf{C} - \mathbf{C}^k \right\|_{\mathrm{F}}^2,
\end{aligned} \tag{2.1.8}$$

其中 τ 为参数. 解可以更新为:

$$\begin{aligned}
\mathbf{A}^{k+1} &= \left(\mathbf{X}_{(1)}\left(\mathbf{C}^k \odot \mathbf{B}^k\right) + \tau \mathbf{A}^k\right)\left(\left(\mathbf{C}^k\right)^{\mathrm{T}}\mathbf{C}^k \circledast \left(\mathbf{B}^k\right)^{\mathrm{T}}\mathbf{B}^k + \tau \mathbf{I}\right)^{-1}, \\
\mathbf{B}^{k+1} &= \left(\mathbf{X}_{(2)}\left(\mathbf{A}^{k+1} \odot \mathbf{C}^k\right) + \tau \mathbf{B}^k\right)\left(\left(\mathbf{A}^{k+1}\right)^{\mathrm{T}}\mathbf{A}^{k+1} \circledast \left(\mathbf{C}^k\right)^{\mathrm{T}}\mathbf{C}^k + \tau \mathbf{I}\right)^{-1}, \\
\mathbf{C}^{k+1} &= \left(\mathbf{X}_{(3)}\left(\mathbf{B}^{k+1} \odot \mathbf{A}^{k+1}\right) + \tau \mathbf{C}^k\right)\left(\left(\mathbf{B}^{k+1}\right)^{\mathrm{T}}\mathbf{B}^{k+1} \circledast \left(\mathbf{A}^{k+1}\right)^{\mathrm{T}}\mathbf{A}^{k+1} + \tau \mathbf{I}\right)^{-1}.
\end{aligned} \tag{2.1.9}$$

算法的步骤如算法 2 所示.

Algorithm 2 RALS for CP decomposition of a third-order tensor

Input: $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ and CP-rank R
Initialize $\mathbf{A}^0, \mathbf{B}^0, \mathbf{C}^0, K, \tau$
repeat
  for $k = 1, \cdots K$ **do**
    $\mathbf{A}^{k+1} = \left(\mathbf{X}_{(1)}\left(\mathbf{C}^k \odot \mathbf{B}^k\right) + \tau \mathbf{A}^k\right)\left(\left(\mathbf{C}^k\right)^{\mathrm{T}}\mathbf{C}^k \circledast \left(\mathbf{B}^k\right)^{\mathrm{T}}\mathbf{B}^k + \tau \mathbf{I}\right)^{-1}$
    $\mathbf{B}^{k+1} = \left(\mathbf{X}_{(2)}\left(\mathbf{A}^{k+1} \odot \mathbf{C}^k\right) + \tau \mathbf{B}^k\right)\left(\left(\mathbf{A}^{k+1}\right)^{\mathrm{T}}\mathbf{A}^{k+1} \circledast \left(\mathbf{C}^k\right)^{\mathrm{T}}\mathbf{C}^k + \tau \mathbf{I}\right)^{-1}$
    $\mathbf{C}^{k+1} = \left(\mathbf{X}_{(3)}\left(\mathbf{B}^{k+1} \odot \mathbf{A}^{k+1}\right) + \tau \mathbf{C}^k\right)\left(\left(\mathbf{B}^{k+1}\right)^{\mathrm{T}}\mathbf{B}^{k+1} \circledast \left(\mathbf{A}^{k+1}\right)^{\mathrm{T}}\mathbf{A}^{k+1} + \tau \mathbf{I}\right)^{-1}$
  end for
until fit ceases to improve or maximum iterations exhausted
Output: $\mathbf{A}, \mathbf{B}, \mathbf{C}$

与 ALS 相比, RALS 的优势有两个. 一是解的稳定性更好, 二是为了防止结果的突变.

2.2 Tucker 分解

Tucker 分解是矩阵奇异值分解 (SVD) 的另一种高阶推广.

定义 2.2 (Tucker 分解) 对于张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, Tucker 分解为

$$\mathcal{X} = \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_N \mathbf{U}^{(N)} = [\![\mathcal{G}; \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \cdots, \mathbf{U}^{(N)}]\!], \tag{2.2.1}$$

或者逐元素地表示为

$$x_{i_1, i_2, \cdots, i_N} = \sum_{r_1}^{R_1} \sum_{r_2}^{R_2} \cdots \sum_{r_N}^{R_N} g_{r_1, r_2, \cdots, r_N} u_{i_1, r_1}^{(1)} u_{i_2, r_2}^{(2)} \cdots u_{i_N, r_N}^{(N)}, \tag{2.2.2}$$

其中 $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times R_n}, n = 1, \cdots, N$ 是因子矩阵, $R_n \leq I_n$ 且 $\mathbf{U}^{(n)\mathrm{T}}\mathbf{U}^{(n)} = \mathbf{I}_{R_n}$, 张量 $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times \cdots R_N}$ 称为核心张量.

图 2.2 是三阶张量的 Tucker 分解示意图, 需要注意的是, Tucker 分解不具有唯一性. 事实上, 令 $\mathbf{W} \in \mathbb{R}^{R_1 \times R_1}, \mathbf{U} \in \mathbb{R}^{R_2 \times R_2}, \mathbf{V} \in \mathbb{R}^{R_3 \times R_3}$ 为非奇异矩阵, 我们有

$$[\![\mathcal{G}; \mathbf{A}, \mathbf{B}, \mathbf{C}]\!] = [\![\mathcal{G} \times_1 \mathbf{W} \times_2 \mathbf{U} \times_3 \mathbf{V}; \mathbf{A}\mathbf{W}^{-1}, \mathbf{B}\mathbf{U}^{-1}, \mathbf{C}\mathbf{V}^{-1}]\!].$$

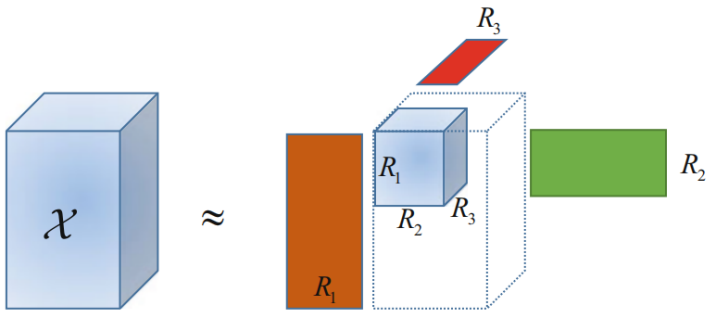


图 2.2: 三阶张量的 Tucker 分解

2.2.1 n 阶秩

设 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, $\mathcal{X}$ 的 n-阶秩 (或多线性秩) 记为 $\mathrm{rank}_n(\mathcal{X})$, 也是 $\mathcal{X}_{(n)}$ 的列秩. 换句话说, n-阶秩是由 mode-n 纤维张成的向量空间的维数.

例如, 三阶张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ 的 n-阶秩可以表示为

$$\begin{aligned} R_1 &= \dim\left(\mathrm{span}_{\mathbb{R}}\left\{\mathbf{x}_{:,i_2,i_3} \mid i_2 = 1, \cdots, I_2, i_3 = 1, \cdots, I_3\right\}\right), \\ R_2 &= \dim\left(\mathrm{span}_{\mathbb{R}}\left\{\mathbf{x}_{i_1,:,i_3} \mid i_1 = 1, \cdots, I_1, i_3 = 1, \cdots, I_3\right\}\right), \\ R_3 &= \dim\left(\mathrm{span}_{\mathbb{R}}\left\{\mathbf{x}_{i_1,i_2,:} \mid i_1 = 1, \cdots, I_1, i_2 = 1, \cdots, I_2\right\}\right), \end{aligned} \tag{2.2.3}$$

其中 $\mathbf{x}_{:,i_2,i_3}, \mathbf{x}_{i_1,:,i_3}, \mathbf{x}_{i_1,i_2,:}$ 分别是张量 $\mathcal{X}$ 的 mode-1, mode-2 和 mode-3 的纤维. 在这种情况下, 我们可以说 $\mathcal{X}$ 是一个秩为 (R_1, R_2, R_3) 的张量. 更一般地, 如果 $R_n = \mathrm{rank}_n(\mathcal{X}), n = 1, \cdots, N$, 我们可以说 $\mathcal{X}$ 是一个秩为 $(R_1, R_2, \cdots, R_N)$ 的张量.

2.2.2 HOSVD

高阶奇异值分解 (HOSVD) 是矩阵奇异值分解 (SVD) 的推广, 它是 Tucker 分解常见的优化方法.

定义 2.3 (HOSVD) 对于张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, Tucker 分解为

$$\mathcal{X} = \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_N \mathbf{U}^{(N)}, \tag{2.2.4}$$

其中 $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times I_n}, n = 1, \cdots, N$ 是因子矩阵, 核心张量 $\mathcal{G} \in \mathbb{R}^{I_1 \times I_2 \times \cdots I_N}$. 其等价的矩阵展开形式为

$$\mathbf{X}_{(n)} = \mathbf{U}^{(n)}\mathbf{G}_{(n)}\left(\mathbf{U}^{(N)} \otimes \cdots \otimes \mathbf{U}^{(n+1)} \otimes \mathbf{U}^{(n-1)} \otimes \mathbf{U}^{(1)}\right)^{\mathrm{T}}, \tag{2.2.5}$$

其中 $\mathbf{G}_{(n)}$ 为核心张量 $\mathcal{G}$ 的 mode-n 展开.

算法步骤如算法 3 所示.

Algorithm 3 Higher-order singular value decomposition(HOSVD)

1: **Input:** $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and multilinear rank $(R_1, R_2, \cdots, R_N)$
2: **Output:** $\mathcal{G}, \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \cdots, \mathbf{U}^{(N)}$
3: **for** $n = 1, \cdots, N$ **do**
4: $\quad \mathbf{U}^{(n)} \leftarrow R_n$ leading left singular vectors $\mathbf{X}_{(n)}$
5: **end for**
6: $\mathcal{G} \leftarrow \mathcal{X} \times_1 \mathbf{U}^{(1)\mathrm{T}} \times_2 \mathbf{U}^{(2)\mathrm{T}} \cdots \times_N \mathbf{U}^{(N)\mathrm{T}}$

2.2.3 HOOI

高阶正交迭代 (HOOI) 可以更高效地计算因子矩阵. 假设 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, 其优化问题为

$$\begin{aligned} &\min_{\mathcal{G}, \mathbf{U}^{(1)}, \cdots, \mathbf{U}^{(N)}} \left\| \mathcal{X} - [\![\mathcal{G}; \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \cdots, \mathbf{U}^{(N)}]\!] \right\|_{\mathrm{F}}^2 \\ &\text{s.t. } \mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \cdots \times R_N}, \mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times R_n}, \mathbf{U}^{(n)\mathrm{T}} \mathbf{U}^{(n)} = \mathbf{I}_{R_n \times R_n}. \end{aligned} \tag{2.2.6}$$

在上述优化问题中, 我们有

$$\begin{aligned} &\left\| \mathcal{X} - [\![\mathcal{G}; \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \cdots, \mathbf{U}^{(N)}]\!] \right\|_{\mathrm{F}}^2 \\ &= \|\mathcal{X}\|_{\mathrm{F}}^2 - 2 \left\langle \mathcal{X}, [\![\mathcal{G}; \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \cdots, \mathbf{U}^{(N)}]\!] \right\rangle + \left\| [\![\mathcal{G}; \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \cdots, \mathbf{U}^{(N)}]\!] \right\|_{\mathrm{F}}^2 \\ &= \|\mathcal{X}\|_{\mathrm{F}}^2 - 2 \left\langle \mathcal{X} \times_1 \mathbf{U}^{(1)\mathrm{T}} \cdots \times_N \mathbf{U}^{(N)\mathrm{T}}, \mathcal{G} \right\rangle + \left\| [\![\mathcal{G}; \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \cdots, \mathbf{U}^{(N)}]\!] \right\|_{\mathrm{F}}^2 \\ &= \|\mathcal{X}\|_{\mathrm{F}}^2 - 2\langle \mathcal{G}, \mathcal{G} \rangle + \|\mathcal{G}\|_{\mathrm{F}}^2 = \|\mathcal{X}\|_{\mathrm{F}}^2 - \|\mathcal{G}\|_{\mathrm{F}}^2 \\ &= \|\mathcal{X}\|_{\mathrm{F}}^2 - \left\| \mathcal{X} \times_1 \mathbf{U}^{(1)\mathrm{T}} \cdots \times_N \mathbf{U}^{(N)\mathrm{T}} \right\|_{\mathrm{F}}^2 \end{aligned} \tag{2.2.7}$$

从而问题 (2.2.6) 可化为

$$\begin{aligned} &\max_{\mathbf{U}^{(n)}} \left\| \mathcal{X} \times_1 \mathbf{U}^{(1)\mathrm{T}} \cdots \times_N \mathbf{U}^{(N)\mathrm{T}} \right\|_{\mathrm{F}}^2 \\ &\text{s.t. } \mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times R_n}, \mathbf{U}^{(n)\mathrm{T}} \mathbf{U}^{(n)} = \mathbf{I}_{R_n \times R_n}. \end{aligned} \tag{2.2.8}$$

写成矩阵形式就有

$$\begin{aligned} &\max_{\mathbf{U}^{(n)}} \left\| \mathbf{U}^{(n)\mathrm{T}} \mathbf{W} \right\|_{\mathrm{F}}^2 \\ &\text{s.t. } \mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times R_n}, \mathbf{U}^{(n)\mathrm{T}} \mathbf{U}^{(n)} = \mathbf{I}_{R_n \times R_n}, \end{aligned} \tag{2.2.9}$$

其中 $\mathbf{W} = \mathbf{X}_{(n)} \left(\mathbf{U}^{(N)} \otimes \cdots \otimes \mathbf{U}^{(n+1)} \otimes \mathbf{U}^{(n-1)} \otimes \cdots \otimes \mathbf{U}^{(1)} \right)$. 该问题的解可以通过奇异值分解 (SVD) 确定, $\mathbf{U}^{(n)}$ 由 $\mathbf{W}$ 的 R_n 个主左奇异向量构成. HOOI 的细节如算法 4 所示.

Algorithm 4 Higher-order orthogonal iteration(HOOI)

Input: $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and multilinear rank $(R_1, R_2, \cdots, R_N)$
Output: $\mathcal{G}, \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \cdots, \mathbf{U}^{(N)}$
Intialize $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times R_n}$ for $n = 1, \cdots, N$ using HOSVD
repeat
  for $n = 1, \cdots, N$ **do**
    $\mathcal{Y} \leftarrow \mathcal{X} \times_1 \mathbf{U}^{(1)\mathrm{T}} \cdots \times_{n-1} \mathbf{U}^{(n-1)\mathrm{T}} \times_{n+1} \mathbf{U}^{(n+1)\mathrm{T}} \cdots \times_N \mathbf{U}^{(N)\mathrm{T}}$
    $\mathbf{U}^{(n)} \leftarrow R_n$ leading left singular vectors $\mathbf{Y}_{(n)}$
  end for
until fit ceases to improve or maximum iterations exhausted
$\mathcal{G} \leftarrow \mathcal{X} \times_1 \mathbf{U}^{(1)\mathrm{T}} \times_2 \mathbf{U}^{(2)\mathrm{T}} \cdots \times_N \mathbf{U}^{(N)\mathrm{T}}$

2.2.4 代码总结

```python
import numpy as np
import torch
import tensorly as tl
import tensorly.tenalg as tg
from tensorly import random
from tensorly.decomposition import tucker, parafac

tl.set_backend('pytorch') # 选取后端为 PyTorch

## 生成张量
tensor = random.random_tensor((4, 2, 3)) # 随机生成一个 4*2*3 的张量
X = tl.tensor(np.arange(24).reshape((3, 4, 2))) # 生成一个 3*4*2 的张量，元素为 0-23

## 张量基本运算
X_fiber = X[:, 0, 0] # 纤维
X_slice = X[:, :, 1] # 切片
X_unfold = tl.unfold(X, 0) # mode-1 展开，结果为 3*8 的矩阵
X_fold = tl.fold(X_unfold, 0, X.shape) # 按照 mode-1 重塑与 X 形状相同的张量
transpose = tl.transpose(X) # 转置，结果为 2*4*3 的张量
vec = tl.tensor_to_vec(X) # 向量化
inner_product = tl.dot(vec, tl.tensor_to_vec(tensor)) # 内积，参数为两个向量，返回的是一维张量
kron = tg.kronecker([X_unfold, X_unfold]) # kronecker 积，结果为 9*64 的矩阵

## 张量分解
weight_CP, factors_CP = parafac(X, rank=5) # CP 分解
core_tucker, factors_tucker = tucker(X, [2, 2, 2]) # tucker 分解
X1 = tl.tucker_to_tensor((core_tucker, factors_tucker)) # 返回核张量和因子矩阵对应的张量
X2 = tl.tucker_to_unfolded((core_tucker, factors_tucker), 2) # 将返回的张量进行 mode-3 展开
```

2.3 BTD

BTD(Block Term Decomposition) 是 Tucker 分解和 CP 分解的一个扩展.

定义 2.4 (BTD) 对于一个张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, 它的 BTD 为

$$\mathcal{X} = \sum_{r=1}^{R} \mathcal{G}_r \times_1 \mathbf{U}_r^{(1)} \times_2 \mathbf{U}_r^{(2)} \cdots \times_N \mathbf{U}_r^{(N)}, \tag{2.3.1}$$

或按照元素可写为

$$\mathcal{X}\left(i_1, i_2, \cdots, i_N\right) = \sum_{r=1}^{R} \sum_{j_1}^{J_1^r} \sum_{j_2}^{J_2^r} \cdots \sum_{j_N}^{J_N^r} \mathcal{G}_r\left(j_1, j_2, \cdots, j_N\right) \mathbf{U}_r^{(1)}\left(i_1, j_1\right) \cdots \mathbf{U}_r^{(N)}\left(i_N, j_N\right), \tag{2.3.2}$$

其中 $\mathbf{U}_r^{(n)} \in \mathbb{R}^{I_n \times J_n^r}$ 表示第 r 项的第 n 个因子矩阵, $\mathcal{G}_r \in \mathbb{R}^{J_1^r \times J_2^r \times \cdots \times J_N^r}$ 为第 r 项的核心张量.

例如, 一个三阶张量 $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ 可以表示为 R 个秩为 (L_r, M_r, N_r) 项的和:

$$\mathcal{X} = \sum_{r=1}^{R} \mathcal{S}_r \times_1 \mathbf{U}_r \times_2 \mathbf{V}_r \times_3 \mathbf{W}_r$$

其中 $\mathcal{S}_r \in \mathbb{R}^{L_r \times M_r \times N_r}$ 是满秩的张, 且 $\mathbf{U}_r \in \mathbb{R}^{I \times L_r}$, $(I \geq L_r)$, $\mathbf{V}_r \in \mathbb{R}^{J \times M_r}$, $(J \geq M_r)$, 以及 $\mathbf{W}_r \in \mathbb{R}^{K \times N_r}$, $(K \geq N_r)$ 是列满秩的, $1 \leq r \leq R$. 为了简化起见, 我们假设 $L_r = L, M_r = M, N_r = N$, $r = 1, \cdots, R$.

在这种分解下, $\mathcal{X}$ 的矩阵表示可以表示为

$$\begin{aligned}
\mathbf{X}_{(1)} &= \mathbf{U}\,\text{blockdiag}\left(\left(\mathbf{S}_1\right)_{(1)}, \cdots, \left(\mathbf{S}_R\right)_{(1)}\right)\left(\mathbf{V} \odot_{\mathrm{b}} \mathbf{W}\right)^{\mathrm{T}}, \\
\mathbf{X}_{(2)} &= \mathbf{V}\,\text{blockdiag}\left(\left(\mathbf{S}_1\right)_{(2)}, \cdots, \left(\mathbf{S}_R\right)_{(2)}\right)\left(\mathbf{W} \odot_{\mathrm{b}} \mathbf{U}\right)^{\mathrm{T}}, \\
\mathbf{X}_{(3)} &= \mathbf{W}\,\text{blockdiag}\left(\left(\mathbf{S}_1\right)_{(3)}, \cdots, \left(\mathbf{S}_R\right)_{(3)}\right)\left(\mathbf{U} \odot_{\mathrm{b}} \mathbf{V}\right)^{\mathrm{T}},
\end{aligned}$$

其中分块矩阵 $\mathbf{U} = [\mathbf{U}_1, \cdots, \mathbf{U}_R], \mathbf{V} = [\mathbf{V}_1, \cdots, \mathbf{V}_R]$, $\mathbf{W} = [\mathbf{W}_1, \cdots, \mathbf{W}_R], \left(\mathbf{S}_r\right)_{(n)}$ 是 $\mathcal{S}_r$ 的 mode-n 展开矩阵, 这里 $\odot_{\mathrm{b}}$ 表示分块的 Khatri-Rao 乘积:

$$\mathbf{A} \odot_{\mathrm{b}} \mathbf{B} = \left(\mathbf{A}_1 \otimes \mathbf{B}_1, \cdots, \mathbf{A}_R \otimes \mathbf{B}_R\right).$$

将 $\mathcal{X}$ 按照向量表示, 分解可以写成

$$\text{vec}(\mathcal{X}) = \left(\mathbf{U} \odot_{\mathrm{b}} \mathbf{V} \odot_{\mathrm{b}} \mathbf{W}\right)\left[\text{vec}\left(\mathcal{S}_1\right); \cdots; \text{vec}\left(\mathcal{S}_R\right)\right].$$

图 2.3 展示了三阶张量的 BTD 的可视化表示.

这样, 对于三阶张量的 BTD, 有两种特殊情况: $(L, L, 1)$ 分解和 $(L, M, \cdot)$ 分解.

定义 2.5 ($(L, L, 1)$ 分解) 三阶张量 $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ 的 $(L, L, 1)$ 分解为

$$\mathcal{X} = \sum_{r}^{R} \mathbf{S}_r \circ \mathbf{c}_r, \tag{2.3.3}$$

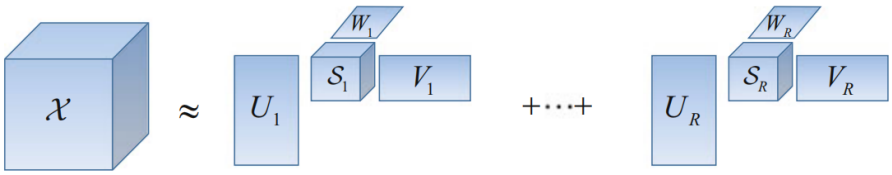


图 2.3: 三阶张量的 BTD

其中矩阵 $\mathbf{S}_r, r = 1, \cdots, R$ 的秩为 L.

在大多数情况下, 不同的矩阵 $\mathbf{S}_r, r = 1, \cdots, R$ 并不一定都具有相同的秩, 我们假设它们的秩为 $L_r, r = 1, \cdots, R$, 从而就有了 $(L_r, L_r, 1)$ 分解.

定义 2.6 ($(L_r, L_r, 1)$ 分解) 三阶张量 $\mathcal{X} \in \mathbb{R}^{I \times J \times K}$ 的 $(L_r, L_r, 1)$ 分解为

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{S}_r \circ \mathbf{c}_r, \tag{2.3.4}$$

其中矩阵 $\mathbf{S}_r, r = 1, \cdots, R$ 的秩为 $L_r, r = 1 \cdots, R$. 如果我们将 $\mathbf{S}_r$ 分解为 $\mathbf{A}_r \mathbf{B}_r^{\mathrm{T}}$, 其中矩阵 $\mathbf{A}_r \in \mathbb{R}^{I \times L_r}$, $\mathbf{B}_r \in \mathbb{R}^{J \times L_r}, L_r \leq I$ 且 $L_r \leq J, r = 1, \cdots, R$, 那么 (2.3.4) 可以写成

$$\mathcal{X} = \sum_{r=1}^{R} \left(\mathbf{A}_r \mathbf{B}_r^{\mathrm{T}}\right) \circ \mathbf{c}_r,$$

$\mathcal{X}$ 的标准矩阵表示为

$$\begin{aligned}
\mathbf{X}_{(1)} &= \mathbf{A} \left(\mathbf{B} \odot_{\mathrm{b}} \mathbf{C}\right)^{\mathrm{T}}, \\
\mathbf{X}_{(2)} &= \mathbf{B} \left(\mathbf{C} \odot_{\mathrm{b}} \mathbf{A}\right)^{\mathrm{T}}, \\
\mathbf{X}_{(3)} &= \mathbf{C} \left[\left(\mathbf{A}_1 \odot \mathbf{B}_1\right) \mathbf{1}_{L_1}, \cdots, \left(\mathbf{A}_R \odot \mathbf{B}_R\right) \mathbf{1}_{L_R}\right]^{\mathrm{T}}.
\end{aligned}$$

其中 $\mathbf{A} = [\mathbf{A}_1, \cdots, \mathbf{A}_R]$, $\mathbf{B} = [\mathbf{B}_1, \cdots, \mathbf{B}_R]$, $\mathbf{C} = [\mathbf{c}_1, \cdots, \mathbf{c}_R]$.

图 2.4 展示了三阶张量的 $(L_r, L_r, 1)$ 分解的可视化表示.

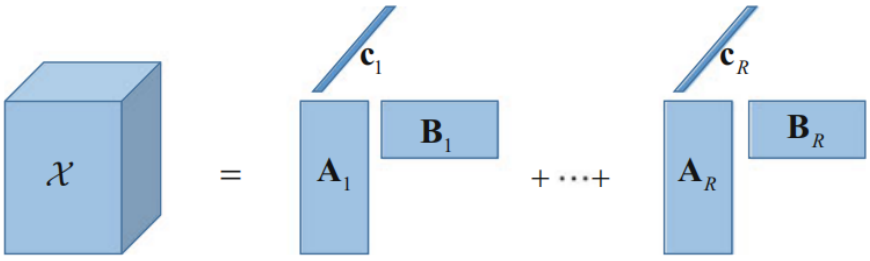


图 2.4: 三阶张量的 $(L_r, L_r, 1)$ 分解

定义 2.7 ($(L,M,\cdot)$ 分解) 三阶张量 $\mathcal{X}\in\mathbb{R}^{I\times J\times K}$ 的 $(L,M,\cdot)$ 分解为

$$\mathcal{X}=\sum_{r=1}^{R}\mathcal{S}_r\times_1\mathbf{A}_r\times_2\mathbf{B}_r, \tag{2.3.5}$$

其中 $\mathcal{S}_r\in\mathbb{R}^{L\times M\times K}$, $\mathbf{A}_r\in\mathbb{R}^{I\times L}$, $\mathbf{B}_r\in\mathbb{R}^{J\times M}, r=1,\cdots,R$ 是列满秩的.

$\mathcal{X}$ 的标准矩阵表示为

$$\begin{aligned}
\mathbf{X}_{(1)}&=\mathbf{A}\left[(\mathcal{S}_1\times_2\mathbf{B}_1)_{(1)};\cdots;(\mathcal{S}_R\times_2\mathbf{B}_R)_{(1)}\right],\\
\mathbf{X}_{(2)}&=\mathbf{B}\left[(\mathcal{S}_1\times_1\mathbf{A}_1)_{(2)};\cdots;(\mathcal{S}_R\times_1\mathbf{A}_R)_{(2)}\right],\\
\mathbf{X}_{(3)}&=\left[(\mathbf{S}_1)_{(3)},\cdots,(\mathbf{S}_R)_{(3)}\right](\mathbf{A}\odot_{\mathrm{b}}\mathbf{B})^{\mathrm{T}},
\end{aligned}$$

其中 $\mathbf{A}=[\mathbf{A}_1,\cdots,\mathbf{A}_R],\mathbf{B}=[\mathbf{B}_1,\cdots,\mathbf{B}_R]$.

图 2.5 展示了三阶张量的 $(L,M,\cdot)$ 分解的可视化表示.

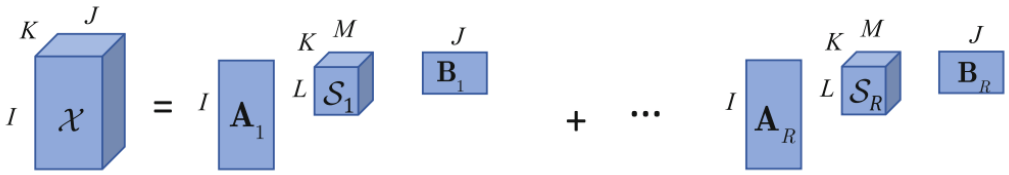


图 2.5: 三阶张量的 $(L,M,\cdot)$ 分解

2.3.1 BTD 的计算

三阶张量 $\mathcal{X}$ 的 (L,M,N) 分解可考虑如下问题:

$$\min_{\mathbf{U},\mathbf{V},\mathbf{W},\mathcal{S}}\left\|\mathcal{X}-\sum_{r=1}^{R}\mathcal{S}_r\times_1\mathbf{U}_r\times_2\mathbf{V}_r\times_3\mathbf{W}_r\right\|_{\mathrm{F}}^2. \tag{2.3.6}$$

类似于在 ALS 框架下的 CP 分解, (2.3.6) 可以分解为以下四个子问题:

$$\begin{aligned}
&\min_{\mathbf{U}}\|\mathbf{X}_{(1)}-\mathbf{U}\,\mathrm{blockdiag}\left((\mathbf{S}_1)_{(1)},\cdots,(\mathbf{S}_R)_{(1)}\right)(\mathbf{V}\odot_{\mathrm{b}}\mathbf{W})^{\mathrm{T}}\|_{\mathrm{F}}^2,\\
&\min_{\mathbf{V}}\|\mathbf{X}_{(2)}-\mathbf{V}\,\mathrm{blockdiag}\left((\mathbf{S}_1)_{(2)},\cdots,(\mathbf{S}_R)_{(2)}\right)(\mathbf{W}\odot_{\mathrm{b}}\mathbf{U})^{\mathrm{T}}\|_{\mathrm{F}}^2,\\
&\min_{\mathbf{W}}\|\mathbf{X}_{(3)}-\mathbf{W}\,\mathrm{blockdiag}\left((\mathbf{S}_1)_{(3)},\cdots,(\mathbf{S}_R)_{(3)}\right)(\mathbf{U}\odot_{\mathrm{b}}\mathbf{V})^{\mathrm{T}}\|_{\mathrm{F}}^2,\\
&\min_{\mathcal{S}}\|\mathrm{vec}(\mathcal{X})-(\mathbf{U}\odot_{\mathrm{b}}\mathbf{V}\odot_{\mathrm{b}}\mathbf{W})\left[\mathrm{vec}\left(\mathcal{S}_1\right);\cdots;\mathrm{vec}\left(\mathcal{S}_R\right)\right]\|_{\mathrm{F}}^2,
\end{aligned}$$

其中 $\mathbf{U}=[\mathbf{U}_1,\cdots,\mathbf{U}_R],\mathbf{V}=[\mathbf{V}_1,\cdots,\mathbf{V}_R]$, $\mathbf{W}=[\mathbf{W}_1,\cdots,\mathbf{W}_R]$. 因此也可以设计出 ALS 框架下的 (L,M,N) 分解算法, 这里不再赘述.

2.3.2 唯一性

对于三阶张量 $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$, 设其 (L, M, N) 分解为

$$\mathcal{X} = \sum_{r=1}^{R} \mathcal{S}_r \times_1 \mathbf{U}_r \times_2 \mathbf{V}_r \times_3 \mathbf{W}_r,$$

当满足以下条件时, 该分解具有本质唯一性:

(1) $L = M$, $I \geq LR$, $J \geq MR$, $N \geq 3$ 且 $\mathbf{W}_r$ 是列满秩的, $1 \leq r \leq R$;

(2) $I \geq LR$, $N > L + M - 2$ 且 $\min\left(\left\lfloor \frac{J}{M} \right\rfloor, R\right) + \min\left(\left\lfloor \frac{K}{N} \right\rfloor, R\right) \geq R + 2$ 或者 $J \geq MR$, $N > L + M - 2$ 且 $\min\left(\left\lfloor \frac{I}{L} \right\rfloor, R\right) + \min\left(\left\lfloor \frac{K}{N} \right\rfloor, R\right) \geq R + 2$;

(3) $N > L + M - 2$ 且 $\min\left(\left\lfloor \frac{I}{L} \right\rfloor, R\right) + \min\left(\left\lfloor \frac{K}{N} \right\rfloor, R\right) + \min\left(\left\lfloor \frac{J}{M} \right\rfloor, R\right) \geq R + 2$.

2.4 张量奇异值分解

不同于 CP 分解和 Tucker 分解, 张量奇异值分解 (t-SVD) 框架下张量的表示是三个张量的 t-乘积. 在介绍 t-SVD 之前, 我们将给出几个重要的定义:

定义 2.8 (t-乘积) 对于三阶张量 $\mathcal{A} \in \mathbb{R}^{I_1\times I_2\times I_3}$ 和 $\mathcal{B} \in \mathbb{R}^{I_2\times J\times I_3}$, t-乘积 $\mathcal{A} * \mathcal{B}$ 是大小为 $I_1 \times J \times I_3$ 的张量:

$$\mathcal{A} * \mathcal{B} = \text{fold}(\text{circ}(\mathcal{A})\,\text{MatVec}(\mathcal{B})),$$

其中

$$\text{circ}(\mathcal{A}) = \begin{bmatrix} \mathcal{A}^{(1)} & \mathcal{A}^{(I_3)} & \cdots & \mathcal{A}^{(2)} \\ \mathcal{A}^{(2)} & \mathcal{A}^{(1)} & \cdots & \mathcal{A}^{(3)} \\ \vdots & \vdots & \vdots & \vdots \\ \mathcal{A}^{(I_3)} & \mathcal{A}^{(I_3-1)} & \cdots & \mathcal{A}^{(1)} \end{bmatrix}, \text{MatVec}(\mathcal{B}) = \begin{bmatrix} \mathcal{B}^{(1)} \\ \mathcal{B}^{(2)} \\ \vdots \\ \mathcal{B}^{(I_3)} \end{bmatrix},$$

且 $\mathcal{A}^{(i_3)}$ 和 $\mathcal{B}^{(i_3)}, i_3 = 1, \cdots, I_3$ 是 $\mathcal{A}$ 和 $\mathcal{B}$ 的前切片.

块循环矩阵可以通过傅里叶变换被块对角化, 从而有以下等式:

$$\overline{\mathbf{A}} = (\mathbf{F} \otimes \mathbf{I}_1)\,\text{circ}(\mathcal{A})\,(\mathbf{F}^* \otimes \mathbf{I}_2) = \begin{bmatrix} \hat{\mathcal{A}}^{(1)} & 0 & \cdots & 0 \\ 0 & \hat{\mathcal{A}}^{(2)} & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & \hat{\mathcal{A}}^{(I_3)} \end{bmatrix},$$

其中 $\mathbf{F} \in \mathbb{R}^{I_3\times I_3}$ 是离散傅里叶变换 (DFT) 矩阵, $\mathbf{F}^*$ 是 $\mathbf{F}$ 的共轭转置, $\mathbf{I}_1 \in \mathbb{R}^{I_1\times I_1}$, $\mathbf{I}_2 \in \mathbb{R}^{I_2\times I_2}$ 是单位矩阵, $\hat{\mathcal{A}}$ 是沿着 $\mathcal{A}$ 的 mode-3 快速傅里叶变换 (FFT). 同时

$$\text{circ}(\mathcal{A})\,\text{Mat Vec}(\mathcal{B}) = (\mathbf{F}^* \otimes \mathbf{I}_1)\,((\mathbf{F} \otimes \mathbf{I}_1)\,\text{circ}(\mathcal{A})\,(\mathbf{F}^* \otimes \mathbf{I}_2))\,(\mathbf{F} \otimes \mathbf{I}_2)\,\text{Mat Vec}(\mathcal{B}),$$

其中 $(\mathbf{F} \otimes \mathbf{I}_2)\,\text{MatVec}(\mathcal{B})$ 可以通过沿着 $\mathcal{B}$ 的 mode-3 应用 FFT 来计算. 记 FFT 的结果为 $\hat{\mathcal{B}}$, 则 t-乘积可以通过将 $\hat{\mathcal{A}}$ 的每个前切片与 $\hat{\mathcal{B}}$ 的每个前切片相乘来计算, 然后沿着管道取逆 FFT 得到.

定义 2.9 (t-SVD) 设 $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, 则 $\mathcal{A}$ 可以分解为

$$\mathcal{A} = \mathcal{U} * \mathcal{S} * \mathcal{V}^{\mathrm{T}},$$

其中 $\mathcal{U} \in \mathbb{R}^{I_1 \times I_1 \times I_3}$ 和 $\mathcal{V} \in \mathbb{R}^{I_2 \times I_2 \times I_3}$ 是正交张量, i.e., $\mathcal{U}^{\mathrm{T}} * \mathcal{U} = \mathcal{V}^{\mathrm{T}} * \mathcal{V} = \mathcal{I}$. 在傅里叶域中, $\hat{\mathcal{A}}^{(i_3)} = \hat{\mathcal{U}}^{(i_3)} \hat{\mathcal{S}}^{(i_3)} \hat{\mathcal{V}}^{(i_3)}, i_3 = 1, \cdots, I_3$.

t-SVD 的可视化如图 2.6 所示, t-SVD 的计算如算法 5 所示, Lu 等人提出了一种新的方法来高效地计算 t-SVD, 如算法 6 所示.

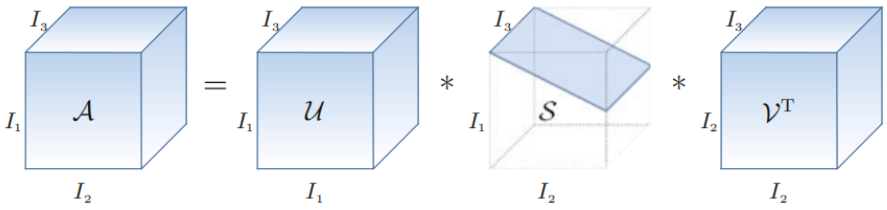


图 2.6: 三阶张量的 t-SVD

Algorithm 5 t-SVD for 3-way tensor

Input: $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$.
$\mathcal{D} \leftarrow \mathrm{fft}(\mathcal{A}, [], 3)$,
for $i_3 = 1$ to I_3 **do**
  $[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \mathrm{SVD}\left(\mathcal{D}\left(:, :, i_3\right)\right)$,
  $\hat{\mathcal{U}}\left(:, :, i_3\right) = \mathbf{U}, \hat{\mathcal{S}}\left(:, :, i_3\right) = \mathbf{S}, \hat{\mathcal{V}}\left(:, :, i_3\right) = \mathbf{V}$,
end for
$\mathcal{U} \leftarrow \mathrm{ifft}(\hat{\mathcal{U}}, [], 3), \mathcal{S} \leftarrow \mathrm{ifft}(\hat{\mathcal{S}}, [], 3), \mathcal{V} \leftarrow \mathrm{ifft}(\hat{\mathcal{V}}, [], 3)$,
Output: $\mathcal{U}, \mathcal{S}, \mathcal{V}$

Algorithm 6 New: t-SVD for 3-way tensor

Input: $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$.
$\hat{\mathcal{A}} = \mathrm{fft}(\mathcal{A}, [], 3)$,
for $i_3 = 1, \cdots, \left\lceil \frac{I_3+1}{2} \right\rceil$ **do**
  $\left[\hat{\mathcal{U}}^{(i_3)}, \hat{\mathcal{S}}^{(i_3)}, \hat{\mathcal{V}}^{2}\left(i_3\right)\right] = \mathrm{SVD}\left(\hat{\mathcal{A}}^{(i_3)}\right)$;
end for
for $i_3 = \left\lceil \frac{I_3+1}{2} \right\rceil + 1, \cdots, I_3$ **do**
  $\hat{\mathcal{U}}^{(i_3)} = \mathrm{conj}\left(\hat{\mathcal{U}}^{(I_3 - i_3 + 2)}\right)$;
  $\hat{\mathcal{S}}^{(i_3)} = \hat{\mathcal{S}}^{(I_3 - i_3 + 2)}$;
  $\hat{\mathcal{V}}^{(i_3)} = \mathrm{conj}\left(\hat{\mathcal{V}}^{(I_3 - i_3 + 2)}\right)$;
end for
$\mathcal{U} = \mathrm{ifft}(\hat{\mathcal{U}}, [], 3), \mathcal{S} = \mathrm{ifft}(\hat{\mathcal{S}}, [], 3), \mathcal{V} = \mathrm{ifft}(\hat{\mathcal{V}}, [], 3)$.
Output: $\mathcal{U}, \mathcal{S}, \mathcal{V}$

定义 2.10 (张量管秩) 张量管秩 (记为 $\text{rank}_\text{t}(\mathcal{A})$) 定义为 $\mathcal{S}$ 的非零奇异管的数量, 其中 $\mathcal{S}$ 是从 t-SVD $\mathcal{A} = \mathcal{U} * \mathcal{S} * \mathcal{V}^\text{T}$ 得到的, i.e.,

$$\text{rank}_\text{t}(\mathcal{A}) = \#\{i : \mathcal{S}(i,i,:) \neq 0\}$$

其中 $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$.

2.5 张量网络

多维数据在科学和工程学科无处不在, 一个数据集可能包含数十亿条目并具有很高的维度, 因此研究适用于高维数据集的基于张量的算法是有必要的. 张量网络 (TNs) 将高阶张量分解为稀疏互连的低阶核心张量, 这为大数据提供了一种自然的稀疏和分布式表示.

本节我们主要介绍两个重要的网络: HT(hierarchical Tucker) 和 TT(tensor train) 网络.

2.5.1 HT 分解

我们首先引入几个重要的定义.

定义 2.11 (维度树) 阶数为 D 的维度树 $\mathbb{T}$ 是以 $\mathbb{D}, \mathbb{D} := \{1, \cdots, D\}$ 为根的二叉树, 对每个节点 $\mathbb{C}_q \subset \mathbb{T}, q = 1, \cdots, Q$ 满足以下性质:

(1) 只有一个条目的节点称为叶子节点, i.e., $\mathbb{C}_p = \{d\}$. 所有叶子节点的集合记为

$$\mathbb{F}(\mathbb{T}) = \{\mathbb{C}_p \subset \mathbb{T} \mid \mathbb{C}_p \text{ is a leaf node of } \mathbb{T}, p = 1, \cdots, P\}$$

其中 P 是叶子的数量.

(2) 一个内部节点是两个不相交的后继节点的并集. 所有内部节点的集合可以表示为

$$\mathbb{E}(\mathbb{T}) = \mathbb{T} \backslash \mathbb{F}(\mathbb{T})$$

其中 $\mathbb{F}(\mathbb{T})$ 表示叶子节点的集合, 内部节点的数量为 $Q - P$.

定义 2.12 (维度树的矩阵化) 对于一个张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$, 一个维度索引的节点 $\mathbb{C}_q \subset \mathbb{D}$ 以及它的补集 $\overline{\mathbb{C}}_q := \mathbb{D} \backslash \mathbb{C}_q$, 该维度树的给定张量 $\mathcal{X}$ 的矩阵化定义为

$$\mathbf{X}^{(q)} \in \mathbb{R}^{I_{\mathbb{C}} \times I_{\overline{\mathbb{C}}_q}},$$

其中

$$I_{\mathbb{C}_q} := \prod_{c \in \mathbb{C}_q} I_c, \quad I_{\overline{\mathbb{C}}_q} := \prod_{\bar{c} \in \overline{\mathbb{C}}_q} I_{\bar{c}}.$$

定义 2.13 (分层秩) 令 $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_D}$, 张量树的秩 $\mathbf{k}$ 定义为

$$\mathbf{k} = \left\{ k_q \mid k_q = \text{rank}\left(\mathbf{X}^{(q)}\right), \forall \mathbb{C}_q \subset \mathbb{T} \right\}.$$

所有分层秩不超过 $\mathbf{k}$ 的张量集合定义为

$$\mathbb{X}_\mathbf{k} = \left\{ \mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_D} \mid \text{rank}\left(\mathbf{X}^{(q)}\right) \leq k_q, \forall \mathbb{C}_q \subset \mathbb{T} \right\}.$$

引理 2.1 (嵌套性性质) 令 $\mathcal{X} \in \mathbb{X}_{\mathbf{k}}$, 对于每个节点 $\mathbb{C}_q$ 及其补集 $\overline{\mathbb{C}}_q$, 我们可以得到一个子空间

$$\mathbb{U}_q := \operatorname{span}\left\{\mathbf{x} \in \mathbb{R}^{I_{\mathbb{C}_q}} \mid \mathbf{x} \text{ is the left singular vector of } \mathbf{X}^{(q)}\right\},$$

其中 $\mathbf{X}^{(q)} \in \mathbb{R}^{I_{\mathbb{C}_q} \times I_{\overline{\mathbb{C}}_q}}$. 对于每个 $\mathbb{C}_q \subset \mathbb{E}(\mathbb{T})$, 若其有两个后继节点 $\mathbb{C}_{q_1}, \mathbb{C}_{q_2}$, 则空间 $\mathbb{U}_q$ 自然分解为

$$\mathbb{U}_q = \mathbb{U}_{q_1} \otimes \mathbb{U}_{q_2},$$

其中 $\otimes$ 表示 Kronecker 积.

基于以上定义, HT 分解定义如下:

定义 2.14 (HT 分解) 令 $\mathcal{X} \in \mathbb{X}_{\mathbf{k}}$, 对于每个节点 $\mathbb{C}_q \subset \mathbb{T}$, 我们可以将 $\mathbf{X}^{(q)}$ 表示为

$$\mathbf{X}^{(q)} = \mathbf{U}_{\mathbb{C}_q} \mathbf{V}_{\mathbb{C}_q}^{\mathrm{T}}, \quad \mathbf{U}_{\mathbb{C}_q} \in \mathbb{R}^{I_{\mathbb{C}_q} \times k_{\mathbb{C}_q}},$$

其中 $k_{\mathbb{C}_q}$ 是 $\mathbf{X}^{(q)}$ 的秩. 对于 $\mathbb{C}_q = \{\mathbb{C}_{q_1}, \mathbb{C}_{q_2}\}$, $\mathbf{U}_{\mathbb{C}_q}$ 的列向量 $\mathbf{U}_{\mathbb{C}_q}(:, l)$ 满足嵌套性性质, 即每个向量 $\mathbf{U}_{\mathbb{C}_q}(:, l)$ 对于 $1 \le l \le k_{\mathbb{C}_q}$, 都有

$$\mathbf{U}_{\mathbb{C}_q}(:, l) = \sum_{l_1=1}^{k_{\mathbb{C}_{q_1}}} \sum_{l_2=1}^{k_{\mathbb{C}_{q_2}}} \mathcal{B}_{\mathbb{C}_q}(l, l_1, l_2)\, \mathbf{U}_{\mathbb{C}_{q_1}}(:, l_1) \otimes \mathbf{U}_{\mathbb{C}_{q_2}}(:, l_2), \tag{2.5.1}$$

其中 $\mathcal{B}_{\mathbb{C}_q}$ 是线性组合的系数.

例如, 对于一个以 HT 分解表示的四阶张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times I_3 \times I_4}$, 其根节点是 $\{1, 2, 3, 4\}$. 我们可以得到根节点的表示 $\mathbf{u}_{\{1,2,3,4\}} \in \mathbb{R}^{I_1 I_2 I_3 I_4 \times 1}$. 由于根节点 $\{1, 2, 3, 4\}$ 是一个内部节点, 它有两个不相交的后继节点, 例如 $\{1, 2\}, \{3, 4\}$ 或 $\{1, 3\}, \{2, 4\}$ 或 $\{1, 4\}, \{2, 3\}$. 我们选取 $\{1, 2\}, \{3, 4\}$ 进行解释. 由 (2.5.1), 根节点 $\mathbf{u}_{\{1,2,3,4\}}$ 可以分解为

$$\mathbf{u}_{\{1,2,3,4\}} = \sum_{l_1=1}^{k_{\{1,2\}}} \sum_{l_2=1}^{k_{\{3,4\}}} \mathcal{B}_{\{1,2,3,4\}}(l_1, l_2)\, \mathbf{U}_{\{1,2\}}(:, l_1) \otimes \mathbf{U}_{\{3,4\}}(:, l_2).$$

同样, 列向量 $\mathbf{U}_{\{1,2\}}(:, l), 1 \le l \le k_{\{1,2\}}$ 可以分解为

$$\mathbf{U}_{\{1,2\}}(:, l) = \sum_{l_1=1}^{k_{\{1\}}} \sum_{l_2=1}^{k_{\{2\}}} \mathcal{B}_{\{1,2\}}(l, l_1, l_2)\, \mathbf{U}_{\{1\}}(:, l_1) \otimes \mathbf{U}_{\{2\}}(:, l_2)$$

其中 $\mathbf{U}_{\{d\}} \in \mathbb{R}^{I_d \times k_{\{d\}}}, d = 1, \cdots, 4$. 通过这种方式, $\mathcal{X}$ 的叶节点表示为 $\mathbf{U}_{\{1\}}, \mathbf{U}_{\{2\}}, \mathbf{U}_{\{3\}}, \mathbf{U}_{\{4\}}$, 内部节点的表示为 $\mathcal{B}_{\{1,2\}}, \mathcal{B}_{\{3,4\}}$ 和 $\mathcal{B}_{\{1,2,3,4\}}$. 图 2.7 给出了图示说明.

如图 2.7 所示, 我们需要存储所有内部节点的转移张量 $\mathcal{B}$ 和所有叶节点的矩阵 $\mathbf{U}$. 因此, 对于所有节点 $q = 1, \cdots, Q$, 假设所有 $k_{\mathbb{C}_q} = k$, 则存储复杂度为 $\sum_{p=1}^{P} Ik + \sum_{q=1}^{Q-P} k^3$.

2.5.1.1 HT 分解的计算

设 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, HT 分解的目标是找到最优的 $\mathcal{B}$ 和 $\mathbf{U}$ 来逼近 $\mathcal{X}$. 这里介绍两种算法.

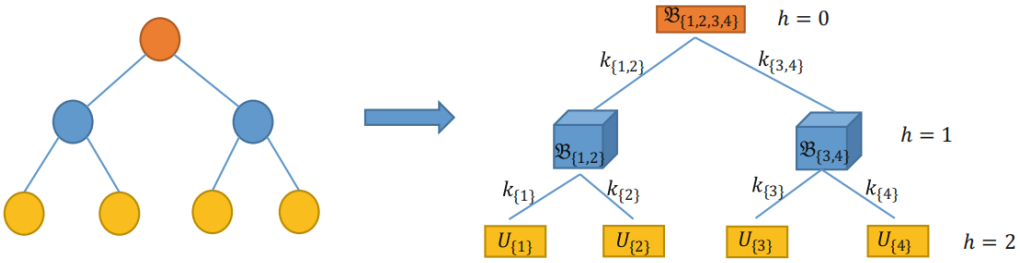


图 2.7: 四阶张量的 HT 分解

第一个是从根到叶的更新方式, 如算法 7 所示, 其中 $\mathrm{diag}(\mathbf{S})$ 表示矩阵 $\mathbf{S}$ 的对角元素, $\mathrm{Length}(\cdot)$ 表示特定向量的大小. 对于张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ 和深度为 H 的维度树 $\mathbb{T}$, 计算复杂度为 $\mathrm{O}\left(\left(\prod_{n=1}^{N} I_n\right)^{3/2}\right)$.

Algorithm 7 Root-to-leaves updating for hierarchical Tucker decomposition

Input: $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$
for $p = 1, \cdots, P$ **do**
  $[\mathbf{U}_p, \mathbf{S}_p, \mathbf{V}_p] = \mathrm{SVD}\left(\mathbf{X}^{(p)}\right)$
  $k_p = \mathrm{rank}\left(\mathbf{S}_p\right)$
end for
for $h = (H-1), \cdots, 0$ **do**
  for $q_h = 1, \cdots, Q_h$ **do**
    if the q_h-th node is an interior one **then**
      $[\mathbf{U}_{q_h}, \mathbf{S}_{q_h}, \mathbf{V}_{q_h}] = \mathrm{SVD}\left(\mathbf{X}^{(q_h)}\right)$
      $k_{q_h} = \mathrm{rank}\left(\mathbf{S}_{q_h}\right)$
      $\mathcal{B}_{q_h} = \mathrm{reshape}\left(\left\langle \mathbf{U}_{q_h}, \mathbf{U}_{q_{h,1}} \otimes \mathbf{U}_{q_{h,2}} \right\rangle, k_{q_h}, k_{q_{h,1}}, k_{q_{h,2}}\right)$
    end if
  end for
end for
$\hat{\mathcal{X}}$ can be constructed from $\mathcal{B}_{q_h}$ and $\mathbf{U}_p$
Output: $\hat{\mathcal{X}}$ in tensor tree format.

第二个是从叶到根的更新方式, 如算法 8 所示, 计算复杂度为 $\mathrm{O}\left(I^N\right)$.

2.5.1.2 HT 分解的推广

HT 分解的推广是树形张量网络状态 (tree tensor network state, TTNS), 其中所有节点都是三阶或更高阶的. 这是一种变分方法, 可以有效地以张量积形式逼近完全活性空间 (CAS) 配置相互作用 (CI) 波函数. TTNS 图示如图 2.8 所示.

Algorithm 8 Leaves-to-root updating for hierarchical Tucker decomposition

Input: $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$
for $p = 1, \cdots, P$ **do**
  $[\mathbf{U}_p, \mathbf{S}_p, \mathbf{V}_p] = \text{SVD}\left(\mathbf{X}^{(p)}\right)$
  $k_p = \text{rank}\left(\mathbf{S}_p\right)$
end for
$\mathcal{C}_{H-1} = \mathcal{X} \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \cdots \times_P \mathbf{U}_P$
for $h = (H-1), \cdots, 1$ **do**
  for $q_h = 1, \cdots, Q_h$ **do**
    if the q_h-th node is an interior one **then**
      $[\mathbf{U}_{q_h}, \mathbf{S}_{q_h}, \mathbf{V}_{q_h}] = \text{SVD}\left(\mathbf{X}^{(q_h)}\right)$
      $k_{q_h} = \text{rank}\left(\mathbf{S}_{q_h}\right)$
      $\mathcal{B}_{q_h} = \text{reshape}\left(\mathbf{U}_{q_h}, k_{q_h}, k_{q_{h,1}}, k_{q_{h,2}}\right)$
    end if
  end for
  $\mathcal{C}_{h-1} = \mathcal{C}_h \times_1 \mathbf{U}_1 \times_2 \mathbf{U}_2 \cdots \times_{(Q_h - P_h)} \mathbf{U}_{(Q_h - P_h)}$
end for
$\hat{\mathcal{X}}$ can be constructed from $\mathcal{B}_{q_h}$ and $\mathbf{U}_p$
Output: $\hat{\mathcal{X}}$ in tensor tree format

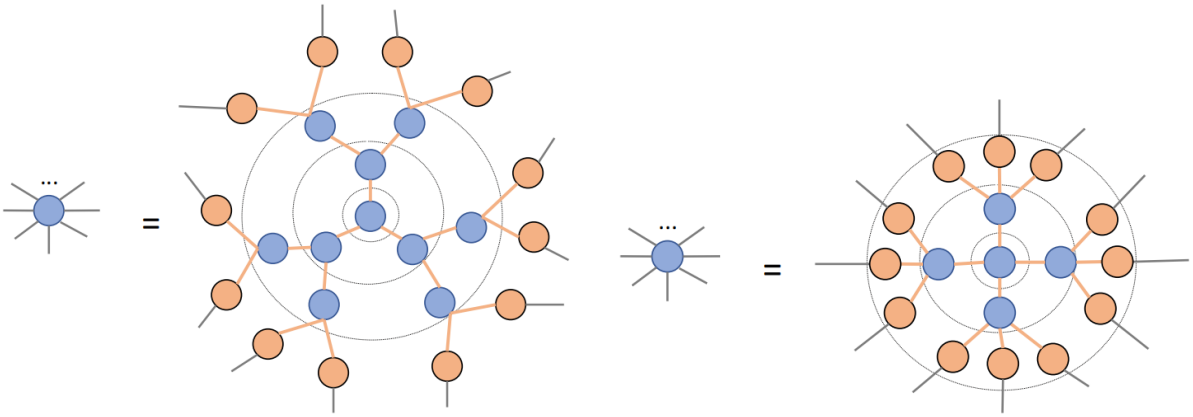


图 2.8: 使用三阶核心 (左) 和四阶核心 (右) 表示 12 阶张量的 TTNS

2.5.2 Tensor train(TT) 分解

定义 2.15 对于一个 N 阶张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, TT 分解定义为

$$\mathcal{X} = \sum_{r_1=1}^{R_1} \cdots \sum_{r_{N+1}=1}^{R_{N+1}} \mathcal{G}^{(1)}\left(r_1, :, r_2\right) \circ \mathcal{G}^{(2)}\left(r_2, :, r_3\right) \circ \cdots \circ \mathcal{G}^{(N)}\left(r_N, :, r_{N+1}\right), \tag{2.5.2}$$

或逐元素表示为

$$\mathcal{X}\left(i_1, i_2 \cdots, i_N\right) = \mathcal{G}^{(1)}\left(:, i_1, :\right) \mathcal{G}^{(2)}\left(:, i_2, :\right) \cdots \mathcal{G}^{(N)}\left(:, i_N, :\right), \tag{2.5.3}$$

其中 $\mathcal{G}_n \in \mathbb{R}^{R_n \times I_n \times R_{n+1}}, n = 1, \cdots, N\,(R_1 = R_{N+1} = 1)$ 是核心因子. 图 2.9 展示了一个 N 阶张量的 TT 分解.

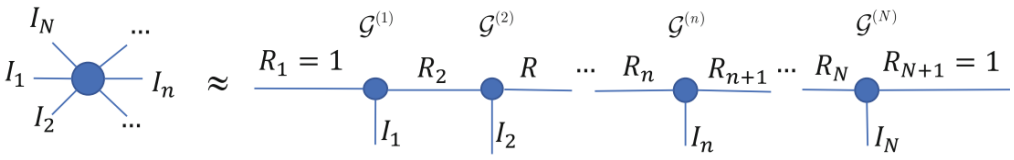


图 2.9: TT 分解

定义 2.16 (TT 秩) 设 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, $\mathcal{X}$ 的 TT 秩记为 $\operatorname{rank}_{\mathrm{TT}}(\mathcal{X})$, 它是 $\mathcal{X}_{\langle n \rangle}$ 的秩. 例如,

$$\operatorname{rank}_{\mathrm{TT}}(\mathcal{X}) = [R_1, \cdots, R_{N+1}],$$

其中 $R_n = \operatorname{rank}\left(\mathcal{X}_{\langle n-1 \rangle}\right), n = 2, \cdots, N, R_1 = R_{N+1} = 1$. 假设所有 $I_n = I$, $R_n = R$, 在 TT 模型中的存储复杂度为 $\mathrm{O}\left((N-2)IR^2 + 2IR\right)$, 其中我们只需要存储三阶核心因子.

2.5.2.1 TT 分解的计算

对于 N 张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, TT 分解的构造基于展开矩阵 $\mathbf{A}_{\langle n \rangle}$ 的顺序 SVD.
例如, 考虑展开矩阵 $\mathbf{X}_{\langle 1 \rangle}$, 如果 $\operatorname{rank}\left(\mathbf{X}_{\langle 1 \rangle}\right) = R_2$, 我们可以得到

$$\mathbf{X}_{\langle 1 \rangle} = \mathbf{U}\mathbf{V}^{\mathrm{T}},$$

其元素为

$$\mathbf{X}_{\langle 1 \rangle}(i_1; j) = \sum_{r_2=1}^{R_2} \mathbf{U}(i_1, r_2)\, \mathbf{V}(r_2, j),$$

其中 $j = \overline{i_2, \cdots, i_d}$. 我们令 $\mathcal{G}_1 = \mathbf{U}$, $\mathcal{G}_1 \in \mathbb{R}^{R_1 \times I_1 \times R_2}, R_1 = 1$. 之后, 矩阵 $\mathbf{V}$ 可以重塑为一个 $N-1$ 阶张量 $\hat{\mathcal{X}} \in \mathbb{R}^{R_2 I_2 \times I_3 \times \cdots \times I_N}$. 我们可以得到

$$\hat{\mathbf{X}}_{\langle 1 \rangle} = \mathbf{U}\mathbf{V}^{\mathrm{T}},$$

其中 $\operatorname{rank}\left(\hat{\mathbf{X}}_{\langle 1 \rangle}\right) = R_3$, $\mathcal{G}_2 = \mathbf{U} \in \mathbb{R}^{R_2 \times I_2 \times R_3}$. 按照这种顺序 SVD 方法, 我们可以得到 TT 分解的核心因子.

若给定 TT 秩, 我们也可以通过 ALS 获取 TT 分解的核心因子. 例如, 设 $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, 优化问题是找到 TT 分解的最佳核心因子来拟合 $\mathcal{X}$:

$$\min_{\mathcal{G}_1, \cdots, \mathcal{G}_N} \left\| \mathcal{X} - \sum_{r_1=1}^{R_1} \cdots \sum_{r_{N+1}=1}^{R_{N+1}} \mathcal{G}^{(1)}(r_1, :, r_2) \circ \mathcal{G}^{(2)}(r_2, :, r_3) \circ \cdots \circ \mathcal{G}^{(N)}(r_N, :, r_{N+1}) \right\|_{\mathrm{F}}^2.$$

算法 9 给出了用于 TT 分解的顺序 SVD 的细节, TT 分解的 ALS 方法如算法 10 所示.

2.5.2.2 TT 分解的推广

TT 分解的推广包括张量环 (TR) 分解、投影纠缠对态 (PEPS)、蜂窝晶格 (HCL)、多尺度纠缠重正化 (MERA) 等张量网络结构. 这里我们重点介绍较为简单的 TR 分解.

Algorithm 9 Sequential SVD for TT decomposition (SSVD)

Input: $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, TT ranks $R_n, n = 1, \cdots, N$
$\mathbf{M}_0 = \mathbf{X}_{(1)}, [\mathbf{U}, \mathbf{S}, \mathbf{V}] = \text{SVD}(\mathbf{M}_0), R_1 = R_{N+1}, \mathbf{G}^{(1)} = \mathbf{U}, \mathbf{M}_1 = \mathbf{SV}$.
repeat
  $[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \text{SVD}(\text{Reshape}(\mathbf{M}_{n-1}, R_n I_n, []))$.
  $R = \text{Length}(\text{diag}(\mathbf{S}))$.
  $R_{n+1} = R, \mathcal{G}^{(n)} = \text{Reshape}(\mathbf{U}, [R_n, I_n, R_{n+1}]), \mathbf{M}_n = \mathbf{SV}$.
until $n = N - 1$
$\mathbf{G}^{(N)} = \mathbf{M}_{N-1}$.
Output: $\mathbf{G}^{(1)}, \cdots, \mathcal{G}^{(n)}, \cdots, \mathbf{G}^{(N)}$

Algorithm 10 Basic ALS for TT decomposition of a tensor

Input: $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ and TT-rank threshold $R_{\max}$
Initialization: run SSVD for TT initialization
repeat
  for $n = 1, \cdots N$ **do**
    if $n == N$ **then**
      $\mathcal{B}_n = \mathcal{G}^{(1)}$.
    else
      $\mathcal{B}_n = \mathcal{G}^{(n+1)}$.
    end if
    for $m = [n+2, \cdots N, 1, \cdots, n-1]$ **do**
      $\mathcal{B}_n = \left\langle \mathcal{B}_n, \mathcal{G}^{(m+1)} \right\rangle_1$.
    end for
    $\mathcal{B}_n = \text{Reshape}\left(\mathcal{B}_n, \left[R_{n+1}, \prod_{m=1, m\neq n}^{N} I_m, R_n\right]\right)$.
    $\mathbf{B}_n = \text{Reshape}\,\text{Permute}(\mathcal{B}_n, [3, 1, 2]), R_n R_{n+1}, [])$.
    update $\mathcal{G}^{(n)}$ by solving least square $\min_{\mathcal{G}^{(n)}} \left\| \left(\mathbf{G}_{(2)}^{(n)} \mathbf{B}_n \right) - \mathbf{X}_{(n)} \right\|_{\text{F}}^2$.
  end for
until fit ceases to improve or maximum iterations exhausted
Output: $\mathcal{G}^{(n)}, n = 1, \cdots, N$

定义 2.17 (张量环分解) 对于一个 N 阶张量 $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, TR 分解定义为

$$\mathcal{X} = \sum_{r_1=1}^{R_1} \cdots \sum_{r_N=1}^{R_N} \mathcal{G}^{(1)}(r_1, :, r_2) \circ \mathcal{G}^{(2)}(r_2, :, r_3) \circ \cdots \circ \mathcal{G}^{(N)}(r_N, :, r_1),$$

其元素为

$$\mathcal{X}(i_1, i_2, \cdots, i_N) = \text{tr}\left(\mathcal{G}^{(1)}(:, i_1, :) \mathcal{G}^{(2)}(:, i_2, :) \cdots \mathcal{G}^{(N)}(:, i_N, :) \right),$$

其中 $\mathcal{G}^{(n)} \in \mathbb{R}^{R_n \times I_n \times R_{n+1}}, n = 1, \cdots, N$ 是核心因子, TR 秩定义为 $[R_1, \cdots, R_N]$. 我们用 $\mathcal{F}\left(\mathcal{G}^{(1)}, \cdots, \mathcal{G}^{(N)}\right)$ 表示张量环分解. 与 TT 分解类似, 在 TR 分解中假设所有 $I_n = I$ 和 $R_n = R$, 存储复杂度为 NIR^2.

张量环分解的详细计算总结在算法 12 中, 其中 $\mathcal{B}n = \bar{\otimes}_{i=1, i\neq n}^{N} \mathcal{G}^{(i)}$ 定义为张量 $\mathcal{G}^{(i)}, i \neq n, i = 1, \cdots, N$ 的张量收缩积.

Algorithm 11 Sequential SVD for TR initialization

Input: $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, TR-rank threshold $R_{\max}$
Initialization: $\mathcal{G}^{(n)}, n = 1, \cdots, N$
$\mathbf{M}_0 = \mathbf{X}_{(1)}, [\mathbf{U}, \mathbf{S}, \mathbf{V}] = \text{SVD}(\mathbf{M}_0), R_1 = 1, R_2 = \min\{R, R_{\max}\}, \mathcal{G}^{(1)}(1 : R_1, :, 1 : R_2) = \mathbf{U}(:, 1 : R_2)$, $\mathbf{M}_1 = \mathbf{SV}$.
repeat
  $[\mathbf{U}, \mathbf{S}, \mathbf{V}] = \text{SVD}(\text{Reshape}(\mathbf{M}_{n-1}, R_n I_n, []))$.
  $R_{n+1} = \min\{R, R_{\max}\}, \mathcal{G}^{(n)} = \text{Reshape}(\mathbf{U}, [R_n, I_n, R_{n+1}])$.
  $\mathbf{M}_n = \mathbf{SV}$
until $n = N - 1$
$\mathcal{G}^{(N)}(1 : R_N, :, 1 : R_1) = \mathbf{M}_{N-1}$.
Output: $\mathcal{G}^{(n)}, n = 1, \cdots, N$

Algorithm 12 Basic ALS for TR decomposition of a tensor

Input: $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ and TR-rank threshold $R_{\max}$
Initialization: run SSVD for TR initialization in Algorithm 11
repeat
  for $n = 1, \cdots N$ **do**
    if $n == N$ **then**
      $\mathcal{B}_n = \mathcal{G}^{(1)}$.
    else
      $\mathcal{B}_n = \mathcal{G}^{(n+1)}$.
    end if
    for $m = [n + 2, \cdots N, 1, \cdots, n - 1]$ **do**
      $\mathcal{B}_n = \langle \mathcal{B}_n, \mathcal{G}^{(m+1)} \rangle_1$.
    end for
    $\mathcal{B}_n = \text{Reshape}\left(\mathcal{B}_n, \left[R_{n+1}, \prod_{m=1, m\neq n}^{N} I_m, R_n\right]\right)$.
    $\mathbf{B}_n = \text{Reshape}(\text{Permute}(\mathcal{B}_n, [3, 1, 2]), R_n R_{n+1}, [])$.
    update $\mathcal{G}^{(n)}$ by solving least square $\min_{\mathcal{G}^{(n)}} \left\| \left(\mathbf{G}_{(2)}^{(n)} \mathbf{B}_n \right) - \mathbf{X}_{(n)} \right\|_{\text{F}}^2$.
  end for
until until fit ceases to improve or maximum iterations exhausted
Output: $\mathcal{G}^{(n)}, n = 1, \cdots, N$

有时, 随着数据张量阶数的增加, TT 分解中的秩可能会迅速增加, 这可能不利于紧凑表示. 为了缓解这个问题, PEPS 提出分层二维 TT 模型, 图 2.10 给出了示例. 此时秩就可以保持相当小, 但代价是使用了四阶甚至五阶核心. 然而, 对于非常高阶的张量, 随着所需近似精度的提高, 秩可能会迅速增加. 为了进一步控制秩, 可以采用其他张量网络, 例如使用三阶核的 HCL(图 2.11) 和由三阶及四阶核张量组成的 MERA(图 2.12).

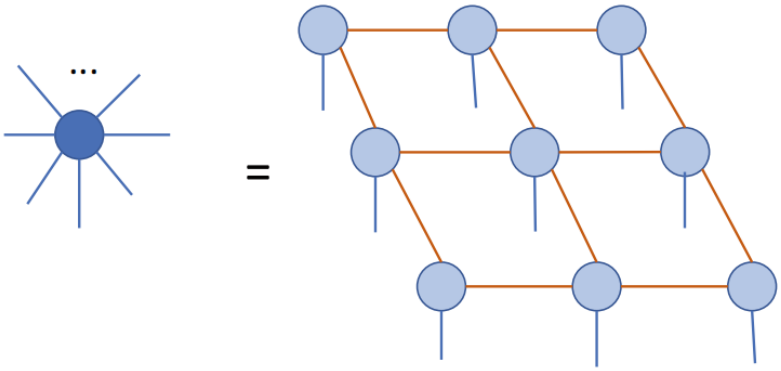


图 2.10: 九阶张量的 PEPS

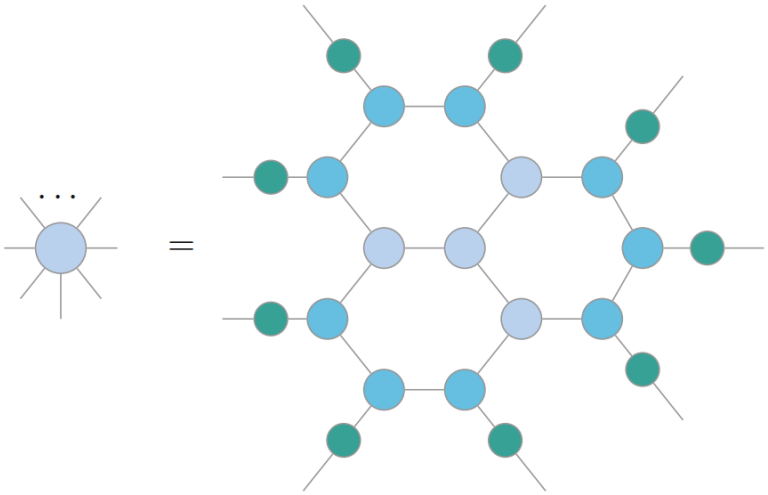


图 2.11: 九阶张量的 HCL

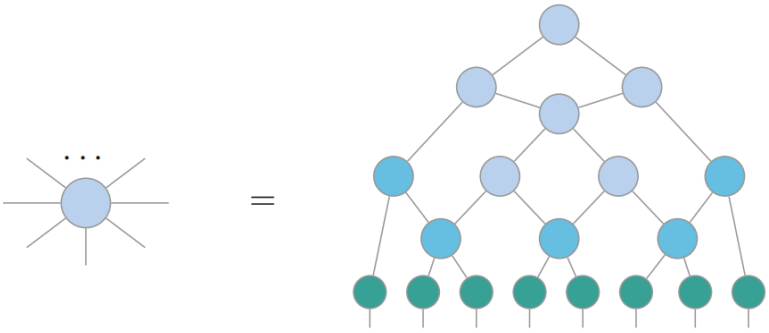


图 2.12: 八阶张量的 MERA

2.6 可伸缩张量分解

张量网络在处理高阶数据方面非常高效, 但在处理包含数十亿条目的大型数据时却显得力不从心. 因此, 开发可伸缩的张量算法来处理这些大数据是有必要的. 根据数据的特性, 我们可以将这些方法分为两类: 一类是利用张量的稀疏性; 另一类是基于低秩假设将大规模张量细分为较小的张量.

2.6.1 可伸缩的稀疏张量分解

处理大规模张量时, 经常会遇到中间数据爆炸问题. 例如, 使用 CP-ALS 算法处理大规模数据 $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$ 时, 计算复杂度主要来自更新因子, 如下所示:

$$\mathbf{A} = \mathbf{X}_{(1)}(\mathbf{C}\odot\mathbf{B})\left(\mathbf{C}^{\mathrm{T}}\mathbf{C}\circledast\mathbf{B}^{\mathrm{T}}\mathbf{B}\right)^{\dagger}, \tag{2.6.1}$$

$$\mathbf{B} = \mathbf{X}_{(2)}(\mathbf{A}\odot\mathbf{C})\left(\mathbf{A}^{\mathrm{T}}\mathbf{A}\circledast\mathbf{C}^{\mathrm{T}}\mathbf{C}\right)^{\dagger}, \tag{2.6.2}$$

$$\mathbf{C} = \mathbf{X}_{(3)}(\mathbf{B}\odot\mathbf{A})\left(\mathbf{B}^{\mathrm{T}}\mathbf{B}\circledast\mathbf{A}^{\mathrm{T}}\mathbf{A}\right)^{\dagger}. \tag{2.6.3}$$

为了简化问题, 我们主要关注方程 (2.6.1). 为了计算方程 (2.6.1), 我们首先计算 $\mathbf{C}\odot\mathbf{B}$ 和 $\mathbf{C}^{\mathrm{T}}\mathbf{C}\circledast\mathbf{B}^{\mathrm{T}}\mathbf{B}$. 对于大规模数据, 矩阵 $\mathbf{C}\odot\mathbf{B}\in\mathbb{R}^{JK\times R}$ 非常大且密集, 无法存储在多个磁盘中, 从而导致中间数据爆炸.

GigaTensor 是第一个在处理大规模稀疏张量时避免中间数据爆炸问题的工作. 其思想是解耦 Khatri-Rao 乘积中的 $\mathbf{C}\odot\mathbf{B}$, 并执行涉及 $\mathbf{X}_{(1)}$ 和 $\mathbf{C}$ 以及 $\mathbf{X}_{(1)}$ 和 $\mathbf{B}$ 的代数运算, 然后合并结果. 在这项工作中, 作者证明了计算 $\mathbf{X}_{(1)}(\mathbf{C}\odot\mathbf{B})$ 等价于计算 $(\mathbf{F}_1\circledast\mathbf{F}_2)\,\mathbf{1}_{JK}$, 其中 $\mathbf{F}_1=\mathbf{X}_{(1)}\circledast\left(\mathbf{1}_I\circ\left(\mathbf{C}(:,r)^{\mathrm{T}}\otimes\mathbf{1}_J^{\mathrm{T}}\right)\right),\mathbf{F}_2=\mathrm{bin}\left(\mathbf{X}_{(1)}\right)\circledast\left(\mathbf{1}_I\circ\left(\mathbf{1}_K^{\mathrm{T}}\otimes\mathbf{B}(:,r)^{\mathrm{T}}\right)\right)$, $\mathbf{1}_{JK}$ 是一个大小为 JK 的全一向量, bin() 是将任何非零值转换为 1 的运算符. 详细的计算如算法 13 所示. 通过这一过程, 计算 $\mathbf{X}_{(1)}(\mathbf{C}\odot\mathbf{B})$ 的浮点运算从 $JKR+2mR$ 减少到 $5mR$, 中间数据大小从 $JKR+m$ 减少到 $\max(J+m,K+m)$, 其中 m 是 $\mathbf{X}_{(1)}$ 的非零元素数量.

Algorithm 13 Calculating $\mathbf{X}_{(1)}(\mathbf{C}\odot\mathbf{B})$

1: **Input:** $\mathbf{X}_{(1)}\in\mathbb{R}^{I\times JK},\mathbf{C}\in\mathbb{R}^{K\times R},\mathbf{B}\in\mathbb{R}^{J\times R}$
2: **Initialization:** $\mathbf{M}_1=\mathbf{0}$;
3: **for** $r=1,\cdots,R$ **do**
4: $\quad\mathbf{F}_1=\mathbf{X}_{(1)}\circledast\left(\mathbf{1}_I\circ\left(\mathbf{C}(:,r)^{\mathrm{T}}\otimes\mathbf{1}_J^{\mathrm{T}}\right)\right)$
5: $\quad\mathbf{F}_2=\mathrm{bin}\left(\mathbf{X}_{(1)}\right)\circledast\left(\mathbf{1}_I\circ\left(\mathbf{1}_K^{\mathrm{T}}\otimes\mathbf{B}(:,r)^{\mathrm{T}}\right)\right)$
6: $\quad\mathbf{F}_3=\mathbf{F}_1\circledast\mathbf{F}_2$
7: $\quad\mathbf{M}_1(:,r)=\mathbf{F}_3\mathbf{1}_{JK}$
8: **end for**
9: **Output:** $\mathbf{M}_1=\mathbf{X}_{(1)}(\mathbf{C}\odot\mathbf{B})$

在这项工作的基础上, HaTen2 将 Tucker 和 CP 分解统一到一个通用框架中. Beutel 等人提出了 FlexiFaCT, 这是一种基于分布式随机梯度下降法的灵活张量分解方法. FlexiFaCT 支持多种分解类型, 如矩阵分解和耦合矩阵-张量因子分解. 这些论文的主要思想是在计算顺序张量-矩阵 (mode) 乘积时避免中间乘积爆炸. 然而, 这些方法在每次迭代步骤中对整个张量进行操作, 当张量非常大时, 这仍然是不可行的.

2.6.2 密集张量分解策略

为了处理大规模且非稀疏的数据, 我们需要将大规模数据分割成小规模数据进行处理. 基于对大数据的低秩假设, 现有的处理大规模低秩张量的算法主要分为两类: 一类是基于并行分布式技术; 另一类是基于投影技术.

2.6.2.1 可伸缩的分布式张量分解

在这一类方法中, 处理大规模数据的思路是开发“分而治之”策略. 该策略包含三个步骤: 首先将大规模张量分解为小规模张量, 其次找到小规模张量的因子, 最后将小规模张量的因子组合起来恢复原始大规模张量的因子. PARCUBE 是第一个使用“分而治之”策略处理大规模张量的方法, 其框架分为三部分: 首先从大规模张量中并行地子采样出多个小规模张量, 其次每个小规模张量独立地通过 CP 分解进行因子分解, 最后通过一个主线性方程将小规模张量的因子结合起来.

例如, 给定一个大规模数据 $\mathcal{X} \in \mathbb{R}^{I\times J\times K}$, 首先从大规模张量中并行子采样出多个小规模张量 (图 2.13), 其中子样本矩阵 $\mathbf{U}_n \in \mathbb{R}^{I\times I_n}, \mathbf{V}_n \in \mathbb{R}^{J\times J_n}, \mathbf{W}_n \in \mathbb{R}^{K\times K_n}$ 是从一个绝对连续分布中随机抽取的, 这些子样本矩阵的元素是具有零均值和单位方差的独立同分布高斯随机变量. 通过这样的子采样和分解策略, 可以有效处理大规模且非稀疏的张量数据, 避免中间数据爆炸的问题, 同时还能提升计算效率.

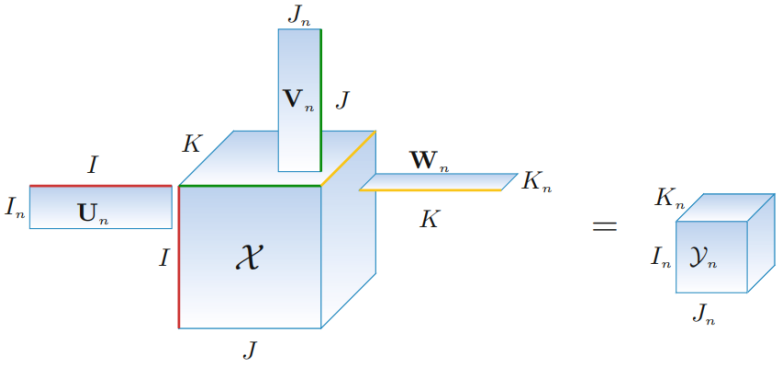


图 2.13: 子采样过程

经过 N 次子采样后, 我们得到 N 个小规模张量, 并在不同的机器上通过 CP 分解并行分解这些小规模张量. 接下来, 我们需要从并行分解步骤中获得的因子恢复原始张量的因子矩阵. 整个过程如图 2.14 所示.

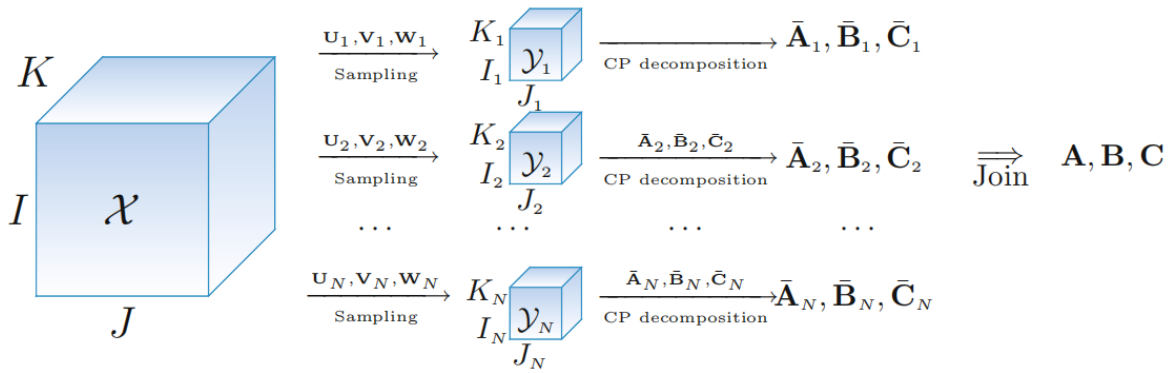


图 2.14: PARCUBE 框架示意图

2.6.2.2 随机张量分解

与可伸缩的分布式张量分解不同, 随机张量分解方法的思想首先是应用随机投影来获得压缩张量, 然后对压缩张量进行不同的张量分解, 最后通过将压缩张量的因子矩阵投影回去, 以获得原始张量的分解. 下面说明使用随机投影对大规模数据进行 CP 分解的思想.

假设 $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ 是一个 N 阶张量, 其中 $I_n (n = 1, \cdots, N)$ 非常大, 随机 CP 分解的第一步是获得一个小规模张量 $\mathcal{Y} \in \mathbb{R}^{J \times \cdots \times J}$, 该张量能够保留原始张量的几乎所有多维信息. 获得压缩张量的关键步骤是寻找一组正交矩阵 $\left\{\mathbf{U}_n \in \mathbb{R}^{I_n \times J}\right\}_{n=1}^{N}$ 作为自然基, 使得

$$\mathcal{X} \approx \mathcal{X} \times_1 \mathbf{U}_1 \mathbf{U}_1^{\mathrm{T}} \times_2 \cdots \times_N \mathbf{U}_N \mathbf{U}_N^{\mathrm{T}}.$$

给定固定的目标秩 J, 这些基矩阵 $\left\{\mathbf{U}_n \in \mathbb{R}^{I_n \times J}\right\}_{n=1}^{N}$ 可以通过随机算法高效地获得. 首先我们构造一个随机样本矩阵 $\mathbf{W} \in \mathbb{R}_{d \neq n} I_d \times J$, 其中每个列向量是从高斯分布中抽取的. 然后随机样本矩阵用于采样 $\mathbf{X}_{(n)} \in \mathbb{R}^{I_n \times \prod_{d \neq n} I_d}$ 的列空间, 如下所示:

$$\mathbf{Z} = \mathbf{X}_{(n)} \mathbf{W},$$

其中 $\mathbf{Z} \in \mathbb{R}^{I_n \times J}$ 表示 $\mathcal{X}_{(n)}$ 的行空间的近似基. 根据概率论, 如果样本矩阵 $\mathbf{W} \in \mathbb{R}^{\Pi_{d \neq n} I_d \times J}$ 的每个列向量都以高概率是线性独立的, 则随机投影 $\mathbf{Z}$ 将有效地采样 $\mathcal{X}_{(n)}$ 的行空间. 最终, 正交基可以通过 QR 分解得到:

$$\mathbf{U}_n = \mathrm{QR}(\mathbf{Z}),$$

之后我们可以将 $\mathcal{X}$ 投影到低维子空间, 如下所示:

$$\mathcal{Y} = \mathcal{X} \times_n \mathbf{U}_n^{\mathrm{T}}.$$

经过 N 次迭代, 我们可以得到压缩张量 $\mathcal{Y}$ 及一组正交矩阵 $\left\{\mathbf{U}_n \in \mathbb{R}^{I_n \times J}\right\}_{n=1}^{N}$. 下一步在 $\mathcal{Y}$ 上执行 CP 分解, 并获得压缩因子矩阵 $\overline{\mathbf{A}}_n \in \mathbb{R}^{J \times R}, n = 1 \ldots, N$. 原始张量的因子矩阵可以通过如下公式恢复:

$$\mathbf{A}_n \approx \mathbf{U}_n \overline{\mathbf{A}}_n, n = 1, \ldots, N$$

其中 $\mathbf{U}_n \in \mathbb{R}^{I_n \times J}, n = 1, \ldots, N$ 表示正交基矩阵. 此外, 过采样和幂迭代方法可用于提高随机算法的求解精度. 整个算法过程总结如算法 14 所示.

Algorithm 14 Randomized tensor compression algorithm

1: **Input:** $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ and a desired target rank J
2: **Initialization:** compressed tensor $\mathcal{Y} = \mathcal{X}$
3: **for** $n = 1, \ldots, N$ **do**
4: generate random sample matrix $\mathbf{W} \in \mathbb{R}^{\prod_{d \neq n} I_d \times J}$
5: $\mathbf{Z} = \mathbf{X}_{(n)} \mathbf{W}$
6: obtain n-mode orthogonal basis $\mathbf{U}_n = \mathrm{QR}(\mathbf{Z})$
7: update compressed tensor $\mathcal{Y} = \mathcal{Y} \times_n \mathbf{U}_n^{\mathrm{T}}$
8: **end for**
9: **Output:** compressed tensor $\mathcal{Y} \in \mathbb{R}^{J \times \cdots \times J}$, orthonormal matrices $\left\{\mathbf{U}_n \in \mathbb{R}^{I_n \times J}\right\}_{n=1}^{N}$.