Structured Matrix Problems from Tensors

Charles F. Van Loan

Abstract This chapter looks at the structured matrix computations that arise in the context of various "svd-like" tensor decompositions. Kronecker products and low-rank manipulations are central to the theme. Algorithmic details include the exploitation of partial symmetries, componentwise optimization, and how we might beat the "curse of dimensionality." Order-4 tensors figure heavily in the discussion.

1 Introduction

A tensor is a multi-dimensional array. Instead of just $A(i,j)$ as for matrices we have $A(i,j,k,\ell,\ldots)$. High-dimensional modeling, cheap storage, and sensor technology combine to explain why tensor computations are surging in importance. Here is an annotated timeline that helps to put things in perspective:

Scalar-Level Thinking

1960's $\Downarrow$ The factorization paradigm: LU, LDL^T, QR, $U\Sigma V^T$, etc.

Matrix-Level Thinking

1980's $\Downarrow$ Memory traffic awareness:, cache, parallel computing, LAPACK, etc.

Block Matrix-Level Thinking

2000's $\Downarrow$ Matrix-tensor connections: unfoldings, Kronecker product, multilinear optimization, etc.

Tensor-Level Thinking

C.F. Van Loan (✉)
Department of Computer Science, Cornell University, Ithaca, NY, USA
e-mail: cv@cs.cornell.edu

M. Benzi, V. Simoncini (eds.), *Exploiting Hidden Structure in Matrix Computations: Algorithms and Applications*, Lecture Notes in Mathematics 2173,
DOI 10.1007/978-3-319-49887-4_1

An important subtext is the changing definition of what we mean by a "big problem." In matrix computations, to say that $A \in \mathbb{R}^{n_1 \times n_2}$is "big" is to say that both n_1 and n_2 are big. In tensor computations, to say that $\mathcal{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$is "big" is to say that $n_1 n_2 \cdots n_d$ is big and this does NOT necessarily require big n_k. For example, no computer in the world (nowadays at least!) can store a tensor with modal dimensions $n_1 = n_2 = \cdots = n_{1000} = 2$. What this means is that a significant part of the tensor research community is preoccupied with the development of algorithms that scale with d. Algorithmic innovations must deal with the "curse of dimensionality." How the transition from *matrix-based* scientific computation to *tensor-based* scientific computation plays out is all about the fate of the "the curse."

This chapter is designed to give readers who are somewhat familiar with matrix computations an idea about the underlying challenges associated with tensor computations. These include mechanisms by which tensor computations are turned into matrix computations and how various matrix algorithms and decompositions (especially the SVD) turn up all along the way. An important theme throughout is the exploitation of Kronecker product structure.

To set the tone we use Sect. 2 to present an overview of some remarkable "hidden structures" that show up in matrix computations. Each of the chosen examples has a review component and a message about tensor-based matrix computations. The connection between block matrices and order-4 tensors is used in Sect. 3 to introduce the idea of a tensor unfolding and to connect Kronecker products and tensor products. A simple nearest rank-1 tensor problem is used in Sect. 4 to showcase the idea of componentwise optimization, a strategy that is widely used in tensor computations. In Sect. 5 we show how Rayleigh quotients can be used to extend the notion of singular values and vectors to tensors. Transposition and tensor symmetry are discussed in Sect. 6. Extending the singular value decomposition to tensors can be done in a number of ways. We present the Tucker decomposition in Sect. 7, the CP decomposition in Sect. 8, the Kronecker product SVD in Sect. 9, and the tensor train SVD in Sect. 10. Cholesky with column pivoting also has a role to play in tensor computations as we show in Sect. 11.

We want to stress that this chapter is a high-level, informal look at the kind of matrix problems that arise out of tensor computations. Implementation details and rigorous analysis are left to the references. To get started with the literature and for general background we recommend [6, 10, 15, 16, 18, 27].

2 The Exploitation of Structure in Matrix Computations

We survey five interesting matrix examples that showcase the idea of hidden structure. By "'hidden" we mean "not obvious". In each case the exploitation of the hidden structure has important ramifications from the computational point of view.

2.1 Exploiting Data Sparsity

The n-by-n discrete Fourier transform matrix F_n is defined by

$$[F_n]_{kq} \;=\; \omega_n^{kq} \qquad \omega_n \;=\; \cos\left(\frac{2\pi}{n}\right) - i\sin\left(\frac{2\pi}{n}\right)$$

where we are subscripting from zero. Thus,

$$F_4 \;=\; \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \omega_4 & \omega_4^2 & \omega_4^3 \\ 1 & \omega_4^2 & \omega_4^4 & \omega_4^6 \\ 1 & \omega_4^3 & \omega_4^6 & \omega_4^9 \end{bmatrix} \;=\; \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{bmatrix}.$$

If we carefully reorder the columns of F_{2m}, then copies of F_m magically appear, e.g.,

$$F_4 \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \;=\; \left[\begin{array}{cc|cc} 1 & 1 & 1 & 1 \\ 1 & -1 & -i & i \\ \hline 1 & 1 & -1 & -1 \\ 1 & -1 & i & -i \end{array}\right] \;=\; \left[\begin{array}{c|c} F_2 & \Omega_2 F_2 \\ \hline F_2 & -\Omega_2 F_2 \end{array}\right]$$

where

$$\Omega_2 \;=\; \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix}.$$

In general we have

$$F_{2m}\Pi_{2,m} \;=\; \begin{bmatrix} F_m & \Omega_m F_m \\ F_m & -\Omega_m F_m \end{bmatrix} \tag{1}$$

where $\Pi_{2,m}$ is a *perfect shuffle* permutation (to be described in Sect. 3.7) and Ω_m is the diagonal matrix

$$\Omega_m \;=\; \mathrm{diag}(1, \omega_n, \ldots, \omega_n^{m-1}).$$

The DFT matrix is dense, but by exploiting the recursion (1) it can be factored into a product of sparse matrices, e.g.,

$$F_{1024} = A_{10} \cdots A_2 A_1 P^T.$$

Here, P is the bit reversal permutation and each A_k has just two nonzeros per row. It is this structured factorization that makes it possible to have a *fast*, $O(n \log n)$ Fourier transform, e.g.,

$y = P^T x$
for $k = 1 : 10$
    $y = A_k y$
end

See [29] for a detailed "matrix factorization" treatment of the FFT.

The DFT matrix F_n is *data sparse* meaning that it can be represented with many fewer than n^2 numbers. Other examples of data sparsity include matrices that have low-rank and matrices that are Kronecker products. *Many tensor problems lead to matrix problems that are data sparse.*

2.2 Exploiting Structured Eigensystems

Suppose $A, F, G \in \mathbb{R}^{n\times n}$and that both F and G are symmetric. The matrix M defined by

$$M = \begin{bmatrix} A & F \\ G & -A^T \end{bmatrix} \qquad F = F^T,\ G = G^T$$

is said to be a *Hamiltonian* matrix. The eigenvalues of a Hamiltonian matrix come in plus-minus pairs and the eigenvectors associated with such a pair are related:

$$M \begin{bmatrix} y \\ z \end{bmatrix} = \lambda \begin{bmatrix} y \\ z \end{bmatrix} \quad \Rightarrow \quad M^T \begin{bmatrix} z \\ -y \end{bmatrix} = -\lambda \begin{bmatrix} z \\ -y \end{bmatrix}.$$

Hamiltonian structure can also be defined through a permutation similarity. If

$$J_{2n} = \begin{bmatrix} 0 & I_n \\ -I_n & 0 \end{bmatrix}$$

then $M \in \mathbb{R}^{2n\times 2n}$is Hamiltonian if $J_{2n}^T M J_{2n} = -M^T$. Under mild assumptions we can compute a structured Schur decomposition for a Hamiltonian matrix M:

$$Q^T M Q = \begin{bmatrix} Q_1 & Q_2 \\ -Q_2 & Q_1 \end{bmatrix}^T M \begin{bmatrix} Q_1 & Q_2 \\ -Q_2 & Q_1 \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} \\ 0 & -T_{11}^T \end{bmatrix}.$$

Here, Q is orthogonal and symplectic ($J_{2n}^T Q J_{2n} = Q^{-T}$) and T_{11} is upper quasi-triangular. Various Ricatti equation problems can be solved by exploiting this structured decomposition. See [22].

Tensors with multiple symmetries can be reshaped into matrices with multiple symmetries and these matrices have structured block factorizations.

2.3 Exploiting the Right Representation

Here is an example of a *Cauchy-like* matrix:

$$A = \begin{bmatrix} \frac{r_1 s_1}{\omega_1-\lambda_1} & \frac{r_1 s_2}{\omega_1-\lambda_2} & \frac{r_1 s_3}{\omega_1-\lambda_3} & \frac{r_1 s_4}{\omega_1-\lambda_4} \\ \frac{r_2 s_1}{\omega_2-\lambda_1} & \frac{r_2 s_2}{\omega_2-\lambda_2} & \frac{r_2 s_3}{\omega_2-\lambda_3} & \frac{r_2 s_4}{\omega_2-\lambda_4} \\ \frac{r_3 s_1}{\omega_3-\lambda_1} & \frac{r_3 s_2}{\omega_3-\lambda_2} & \frac{r_3 s_3}{\omega_3-\lambda_3} & \frac{r_3 s_4}{\omega_3-\lambda_4} \\ \frac{r_4 s_1}{\omega_4-\lambda_1} & \frac{r_4 s_2}{\omega_4-\lambda_2} & \frac{r_4 s_3}{\omega_4-\lambda_3} & \frac{r_4 s_4}{\omega_4-\lambda_4} \end{bmatrix}.$$

For this to be defined we must have $\{\lambda_1,\ldots,\lambda_n\} \cup \{\mu_1,\ldots,\mu_n\} = \emptyset$. Cauchy-like matrices are data sparse and a particularly clever characterization of this fact is to note that if $\Omega = \mathrm{diag}(\omega_i)$ and $\Lambda = \mathrm{diag}(\lambda_i)$, then

$$\Omega A - A\Lambda = rs^T \tag{2}$$

where $r, s \in \mathbb{R}^n$. If $\Omega A - A\Lambda$ has rank r, then we say that A has *displacement rank r* (with respect to Ω and Λ.) Thus, a Cauchy-like matrix has unit displacement rank.

Now let us consider the first step of Gaussian elimination. Ignoring pivoting this involves computing a row of U, a column of L, and a rank-1 update:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \ell_{21} & 1 & 0 & 0 \\ \ell_{31} & 0 & 1 & 0 \\ \ell_{41} & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & b_{22} & b_{23} & b_{24} \\ 0 & b_{32} & b_{33} & b_{34} \\ 0 & b_{42} & b_{43} & b_{44} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{14} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

It is easy to compute the required entries of L and U from the displacement rank representation (2). What about B? If we represent B conventionally as an array then that would involve $O(n^2)$ flops. Instead we exploit the fact that B also has unit displacement rank:

$$\tilde{\Omega}B - B\tilde{\Lambda} \;=\; \tilde{r}\tilde{s}^T.$$

It turns out that we can transition from A's representation $\{\Omega, \Lambda, r, s\}$ to B's representation $\{\tilde{\Omega}, \tilde{\Lambda}, \tilde{r}, \tilde{s}\}$ with $O(n)$ work and this enables us to compute the LU factorization of a Cauchy-like matrix with just $O(n^2)$ work. See [10, p. 682].

Being able to work with clever representations is often the key to having a successful solution framework for a tensor problem.

2.4 Exploiting Orthogonality Structures

Assume that the columns of the 2-by-1 block matrix

$$Q \;=\; \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix}$$

are orthonormal, i.e., $Q_1^TQ_1 + Q_2^TQ_2 \;=\; I$. The *CS decomposition* says that Q_1 and Q_2 have related SVDs:

$$\begin{bmatrix} U_1 & 0 \\ 0 & U_2 \end{bmatrix}^T \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} V \;=\; \begin{bmatrix} \mathrm{diag}(c_i) \\ \mathrm{diag}(s_i) \end{bmatrix} \qquad c_i^2 + s_i^2 = 1 \tag{3}$$

where U_1, U_2, and V are orthogonal. This truly remarkable hidden structure can be used to compute stably the *generalized singular value decomposition* (GSVD) of a $A_1 \in \mathbb{R}^{m_1\times n}$and $A_2 \in \mathbb{R}^{m_2\times n}$. In particular, suppose we compute the QR factorization

$$\begin{bmatrix} A_1 \\ A_2 \end{bmatrix} = \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} R,$$

and then the CS decomposition (3). By setting $X = R^TV$, we obtain the GSVD

$$A_1 = U_1 \cdot \mathrm{diag}(c_i) \cdot X^T \qquad A_2 = U_2 \cdot \mathrm{diag}(s_i) \cdot X^T.$$

For a more detailed discussion about the GSVD and the CS decomposition, see [10, p. 309].

A tensor decomposition can often be regarded as a simultaneous decomposition of its (many) matrix "layers".

2.5 Exploiting a Structured Data layout

Suppose we have a block matrix

$$A = \begin{bmatrix} A_{11} & A_{12} & \cdots & A_{1N} \\ A_{21} & A_{22} & \cdots & A_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ A_{M1} & A_{M2} & \cdots & A_{MN} \end{bmatrix}$$

that is stored in such a way that the data in each A_{ij} is contiguous in memory. Note that there is nothing structured about "A-the-matrix". However, "A-the-stored-array" does have an exploitable structure that we now illustrate by considering the computation of $C = A^T$. We start by transposing each of A's blocks:

$$\begin{bmatrix} B_{11} & B_{12} & \cdots & B_{1N} \\ B_{21} & B_{22} & \cdots & B_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ B_{M1} & B_{M2} & \cdots & B_{MN} \end{bmatrix} \leftarrow \begin{bmatrix} A_{11}^T & A_{12}^T & \cdots & A_{1N}^T \\ A_{21}^T & A_{22}^T & \cdots & A_{2N}^T \\ \vdots & \vdots & \ddots & \vdots \\ A_{M1}^T & A_{M2}^T & \cdots & A_{MN}^T \end{bmatrix}.$$

These block transpositions involve "local data" and this is important because moving data around in a large scale matrix computation is typically the dominant cost. Next, we transpose B as a block matrix:

$$\begin{bmatrix} C_{11} & C_{12} & \cdots & C_{1M} \\ C_{21} & C_{22} & \cdots & C_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ C_{N1} & C_{2N} & \cdots & C_{NM} \end{bmatrix} \leftarrow \begin{bmatrix} B_{11} & B_{21} & \cdots & B_{M1} \\ B_{12} & B_{22} & \cdots & B_{M2} \\ \vdots & \vdots & \ddots & \vdots \\ B_{1N} & B_{2N} & \cdots & B_{MN} \end{bmatrix}$$

Again, this is a "memory traffic friendly" maneuver because blocks of contiguous data are being moved. It is easy to verify that $C_{ij} = A_{ji}^T$.

What we have sketched is a "2-pass" transposition procedure. By blocking in a way that resonates with cache/local memory size and by breaking the overall transposition process down into a sequence of carefully designed passes, one can effectively manage the underlying dataflow. See [10, p. 711].

Transposition and looping are much more complicated with tensors because there are typically an exponential number of possible data structures and an exponential number of possible loop nestings. Software tools that facilitate reasoning in this space are essential. See [1, 24, 28].

3 Matrix-Tensor Connections

Tensor computations typically get "reshaped" into matrix computations. To operate in this venue we need terminology and mechanisms for matricizing the tensor data. In this section we introduce the notion of a tensor unfolding and we get comfortable with Kronecker products and their properties. For more details see [10, 16, 18, 25, 27].

3.1 Talking About Tensors

An order-d tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ is a real d-dimensional array

$$\mathcal{A}(1:n_1, \ldots, 1:n_d)$$

where the index range in the k-th *mode* is from 1 to n_k. Note that

$$\text{a} \left\{ \begin{array}{l} \text{scalar} \\ \text{vector} \\ \text{matrix} \end{array} \right\} \text{ is an } \left\{ \begin{array}{l} \text{order-0} \\ \text{order-1} \\ \text{order-2} \end{array} \right\} \text{ tensor.}$$

We use calligraphic font to designate tensors e.g., $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$, etc. Sometimes we will write $\mathcal{A}$ for matrix A if it makes things clear.

One way that tensors arise is through discretization. $\mathcal{A}(i,j,k,\ell)$ might house the value of $f(w,x,y,z)$ at $(w,x,y,z) = (w_i, x_j, y_k, z_\ell)$. In multiway analysis the value of $\mathcal{A}(i,j,k,\ell)$ could measure the interaction between four variables/factors. See [2] and [27].

3.2 Tensor Parts: Fibers and Slices

A *fiber* of a tensor $\mathcal{A}$ is a column vector obtained by fixing all but one $\mathcal{A}$'s indices. For example, if $\mathcal{A} = \mathcal{A}(1{:}3, 1{:}5, 1{:}4, 1{:}7) \in \mathbb{R}^{3\times5\times4\times7}$, then

$$\mathcal{A}(2,:,4,6) = \mathcal{A}(2,1{:}5,4,6) = \begin{bmatrix} \mathcal{A}(2,1,4,6) \\ \mathcal{A}(2,2,4,6) \\ \mathcal{A}(2,3,4,6) \\ \mathcal{A}(2,4,4,6) \\ \mathcal{A}(2,5,4,6) \end{bmatrix}$$

is a mode-2 fiber.

A *slice* of a tensor $\mathcal{A}$ is a matrix obtained by fixing all but two of $\mathcal{A}$'s indices. For example, if $\mathcal{A} = \mathcal{A}(1{:}3, 1{:}5, 1{:}4, 1{:}7)$, then

$$\mathcal{A}(:,3,:,6) = \begin{bmatrix} \mathcal{A}(1,3,1,6) & \mathcal{A}(1,3,2,6) & \mathcal{A}(1,3,3,6) & \mathcal{A}(1,3,4,6) \\ \mathcal{A}(2,3,1,6) & \mathcal{A}(2,3,2,6) & \mathcal{A}(2,3,3,6) & \mathcal{A}(2,3,4,6) \\ \mathcal{A}(3,3,1,6) & \mathcal{A}(3,3,2,6) & \mathcal{A}(3,3,3,6) & \mathcal{A}(3,3,4,6) \end{bmatrix}$$

is a slice.

3.3 Order-4 Tensors and Block Matrices

Block matrices with uniformly-sized blocks are reshaped order-4 tensors. For example, if

$$C = \left[\begin{array}{cc|cc|cc} c_{11} & c_{12} & c_{13} & c_{14} & c_{15} & c_{16} \\ c_{21} & c_{22} & c_{23} & c_{24} & c_{25} & c_{26} \\ \hline c_{31} & c_{32} & c_{33} & c_{34} & c_{35} & c_{36} \\ c_{41} & c_{42} & c_{43} & c_{44} & \boxed{c_{45}} & c_{46} \\ \hline c_{51} & c_{52} & c_{53} & c_{54} & c_{55} & c_{56} \\ c_{61} & c_{62} & c_{63} & c_{64} & c_{65} & c_{66} \end{array}\right]$$

then matrix entry c_{45} is entry (2,1) of block (2,3). Thus, we can think of $[C_{ij}]_{k\ell}$ as the (i,j,k,ℓ) entry of a tensor $\mathcal{C}$, e.g.,

$$c_{45} = [C_{23}]_{21} = \mathcal{C}(2,3,2,1).$$

Working in the other direction we can *unfold* an order-4 tensor into a block matrix. Suppose $\mathcal{A} \in \mathbb{R}^{n\times n\times n\times n}$. Here is its "$[1,2]\times[3,4]$" unfolding:

$$\mathcal{A}_{[1,2]\times[3,4]} = \left[\begin{array}{ccc|ccc|ccc} a_{1111} & a_{1112} & a_{1113} & a_{1121} & a_{1122} & a_{1123} & a_{1131} & a_{1132} & a_{1133} \\ a_{1211} & a_{1212} & a_{1213} & a_{1221} & a_{1222} & a_{1223} & a_{1231} & a_{1232} & a_{1233} \\ a_{1311} & a_{1312} & a_{1313} & a_{1321} & a_{1322} & a_{1323} & a_{1331} & a_{1332} & a_{1333} \\ \hline a_{2111} & a_{2112} & a_{2113} & a_{2121} & a_{2122} & a_{2123} & a_{2131} & a_{2132} & a_{2133} \\ a_{2211} & a_{2212} & a_{2213} & a_{2221} & a_{2222} & a_{2223} & a_{2231} & a_{2232} & a_{2233} \\ a_{2311} & a_{2312} & a_{2313} & a_{2321} & a_{2322} & a_{2323} & a_{2331} & a_{2332} & a_{2333} \\ \hline a_{3111} & a_{3112} & a_{3113} & a_{3121} & a_{3122} & a_{3123} & a_{3131} & a_{3132} & a_{3133} \\ a_{3211} & a_{3212} & a_{3213} & a_{3221} & a_{3222} & a_{3223} & a_{3231} & a_{3232} & a_{3233} \\ a_{3311} & a_{3312} & a_{3313} & a_{3321} & a_{3322} & a_{3323} & a_{3331} & a_{3332} & a_{3333} \end{array}\right].$$

If $A = \mathcal{A}_{[1,2]\times[3,4]}$ then the tensor-to-matrix mapping is given by

$$\mathcal{A}(i_1, i_2, i_3, i_4) \;\rightarrow\; A(i_1 + (i_2 - 1)n, i_3 + (i_4 - 1)n).$$

An alternate unfolding results if modes 1 and 3 are associated with rows and modes 2 and 4 are associated with columns:

$$\mathcal{A}_{[1,3]\times[2,4]} = \left[\begin{array}{ccc|ccc|ccc} a_{1111} & a_{1112} & a_{1113} & a_{1211} & a_{1212} & a_{1213} & a_{1311} & a_{1312} & a_{1313} \\ a_{1121} & a_{1122} & a_{1123} & a_{1221} & a_{1222} & a_{1223} & a_{1321} & a_{1322} & a_{1323} \\ a_{1131} & a_{1132} & a_{1133} & a_{1231} & a_{1232} & a_{1233} & a_{1331} & a_{1332} & a_{1333} \\ \hline a_{2111} & a_{2112} & a_{2113} & a_{2211} & a_{2212} & a_{2213} & a_{2311} & a_{2312} & a_{2313} \\ a_{2121} & a_{2122} & a_{2123} & a_{2221} & a_{2222} & a_{2223} & a_{2321} & a_{2322} & a_{2323} \\ a_{2131} & a_{2132} & a_{2133} & a_{2231} & a_{2232} & a_{2233} & a_{2331} & a_{2332} & a_{2333} \\ \hline a_{3111} & a_{3112} & a_{3113} & a_{3211} & a_{3212} & a_{3213} & a_{3311} & a_{3312} & a_{3313} \\ a_{3121} & a_{3122} & a_{3123} & a_{3221} & a_{3222} & a_{3223} & a_{3321} & a_{3322} & a_{3323} \\ a_{3131} & a_{3132} & a_{3133} & a_{3231} & a_{3232} & a_{3233} & a_{3331} & a_{3332} & a_{3333} \end{array}\right].$$

If $A = \mathcal{A}_{[1,3]\times[2,4]}$, then the tensor-to-matrix mapping is given by

$$\mathcal{A}(i_1, i_2, i_3, i_4) \;\rightarrow\; A(i_1 + (i_3 - 1)n, i_2 + (i_4 - 1)n).$$

If a tensor $\mathcal{A}$ is structured, then different unfoldings reveal that structure in different ways [31]. The idea of block unfoldings is discussed in [25].

3.4 Modal Unfoldings

A particularly important class of tensor unfoldings are the *modal* unfoldings. An order-d tensor has d modal unfoldings which we designate by $\mathcal{A}_{(1)}, \ldots, \mathcal{A}_{(d)}$. Let's look at the tensor-to-matrix mappings for the case $\mathcal{A} \in \mathbb{R}^{n_1\times n_2\times n_3\times n_4}$:

$$\mathcal{A}(i_1, i_2, i_3, i_4) \rightarrow \mathcal{A}_{(1)}(i_1, i_2 + (i_3 - 1)n_2 + (i_4 - 1)n_2n_3)$$

$$\mathcal{A}(i_1, i_2, i_3, i_4) \rightarrow \mathcal{A}_{(2)}(i_2, i_1 + (i_3 - 1)n_1 + (i_4 - 1)n_1n_3)$$

$$\mathcal{A}(i_1, i_2, i_3, i_4) \rightarrow \mathcal{A}_{(3)}(i_3, i_1 + (i_2 - 1)n_1 + (i_4 - 1)n_1n_2)$$

$$\mathcal{A}(i_1, i_2, i_3, i_4) \rightarrow \mathcal{A}_{(4)}(i_4, i_1 + (i_2 - 1)n_1 + (i_3 - 1)n_1n_2).$$

Note that if $N = n_1 \cdots n_d$, then $\mathcal{A}_{(k)}$ is n_k-by-N/n_k and its columns are mode-k fibers. Thus, if $n_2 = 4$ and $n_1 = n_3 = n_4 = 2$, then

$$\mathcal{A}_{(2)} = \begin{bmatrix} a_{1\mathbf{1}11} & a_{11\mathbf{2}1} & a_{1\mathbf{1}12} & a_{11\mathbf{2}2} & a_{2\mathbf{1}11} & a_{21\mathbf{2}1} & a_{2\mathbf{1}12} & a_{21\mathbf{2}2} \\ a_{1\mathbf{2}11} & a_{12\mathbf{2}1} & a_{1\mathbf{2}12} & a_{12\mathbf{2}2} & a_{2\mathbf{2}11} & a_{22\mathbf{2}1} & a_{2\mathbf{2}12} & a_{22\mathbf{2}2} \\ a_{1\mathbf{3}11} & a_{13\mathbf{2}1} & a_{1\mathbf{3}12} & a_{13\mathbf{2}2} & a_{2\mathbf{3}11} & a_{23\mathbf{2}1} & a_{2\mathbf{3}12} & a_{23\mathbf{2}2} \\ a_{1\mathbf{4}11} & a_{14\mathbf{2}1} & a_{1\mathbf{4}12} & a_{14\mathbf{2}2} & a_{2\mathbf{4}11} & a_{24\mathbf{2}1} & a_{2\mathbf{4}12} & a_{24\mathbf{2}2} \end{bmatrix}.$$

Modal unfoldings arise naturally in many multilinear optimization settings.

3.5 The vec Operation

The *vec* operator turns tensors into column vectors. The vec of a matrix is obtained by stacking its columns:

$$A \in \mathbb{R}^{3\times 2} \quad \Rightarrow \quad \operatorname{vec}(A) = \begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \\ a_{12} \\ a_{22} \\ a_{32} \end{bmatrix}.$$

The vec of an order-3 tensor $\mathcal{A} \in \mathbb{R}^{n_1\times n_2\times n_3}$ stacks the vecs of the slices $\mathcal{A}(:,:,1), \ldots, \mathcal{A}(:,:,n_3)$, e.g.,

$$\mathcal{A} \in \mathbb{R}^{2\times 2\times 2} \quad \Rightarrow \quad \operatorname{vec}(\mathcal{A}) = \begin{bmatrix} \operatorname{vec}(\mathcal{A}(:,:,1)) \\ \hline \operatorname{vec}(\mathcal{A}(:,:,2)) \end{bmatrix} = \begin{bmatrix} a_{111} \\ a_{211} \\ a_{121} \\ a_{221} \\ \hline a_{112} \\ a_{212} \\ a_{122} \\ a_{222} \end{bmatrix}.$$

In general, if $\mathcal{A} \in \mathbb{R}^{n_1\times\cdots\times n_d}$ and the order $d-1$ tensor $\mathcal{A}_k$ is defined by $\mathcal{A}_k = \mathcal{A}(:,\ldots,:,k)$, then we have the following recursive definition:

$$\operatorname{vec}(\mathcal{A}) = \begin{bmatrix} \operatorname{vec}(\mathcal{A}_1) \\ \vdots \\ \operatorname{vec}(\mathcal{A}_{n_d}) \end{bmatrix}.$$

Thus, vec unfolds a tensor into a column vector.

3.6 The Kronecker Product

Unfoldings enable us to reshape tensor computations as matrix computations and *Kronecker products* are very often part of the scene. The Kronecker product $A = B \otimes C$ of two matrices B and C is a block matrix (A_{ij}) where $A_{ij} = b_{ij}C$, e.g.,

$$A = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} \otimes \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} = \left[\begin{array}{c|c|c} b_{11}C & b_{12}C & b_{13}C \\ \hline b_{21}C & b_{22}C & b_{23}C \\ \hline b_{31}C & b_{32}C & b_{33}C \end{array}\right].$$

Of course, $B \otimes C$ is also a matrix of scalars:

$$A = \left[\begin{array}{cc|cc|cc} b_{11}c_{11} & b_{11}c_{12} & b_{12}c_{11} & b_{12}c_{12} & b_{13}c_{11} & b_{13}c_{12} \\ b_{11}c_{21} & b_{11}c_{22} & b_{12}c_{21} & b_{12}c_{22} & b_{13}c_{21} & b_{13}c_{22} \\ \hline b_{21}c_{11} & b_{21}c_{12} & b_{22}c_{11} & b_{22}c_{12} & b_{23}c_{11} & b_{23}c_{12} \\ b_{21}c_{21} & b_{21}c_{22} & b_{22}c_{21} & b_{22}c_{22} & b_{23}c_{21} & b_{23}c_{22} \\ \hline b_{31}c_{11} & b_{31}c_{12} & b_{32}c_{11} & b_{32}c_{12} & b_{33}c_{11} & b_{33}c_{12} \\ b_{31}c_{21} & b_{31}c_{22} & b_{32}c_{21} & b_{32}c_{22} & b_{33}c_{21} & b_{33}c_{22} \end{array}\right].$$

Note that every possible product $b_{ij}c_{k\ell}$ "shows up" in $B \otimes C$.

In general, if $A_1 \in \mathbb{R}^{m_1 \times n_1}$and $A_2 \in \mathbb{R}^{m_2 \times n_2}$, then $A = A_1 \otimes A_2$ is an $m_1 m_2$-by-$n_1 n_2$ matrix of scalars. It is also an m_1-by-n_1 block matrix with blocks that are m_2-by-n_2.

Kronecker products can be applied in succession. Thus, if

$$A = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{bmatrix} \otimes \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \\ c_{41} & c_{42} & c_{43} & c_{44} \end{bmatrix} \otimes \begin{bmatrix} d_{11} & d_{12} & d_{13} & d_{14} \\ d_{21} & d_{22} & d_{23} & d_{24} \\ d_{31} & d_{32} & d_{33} & d_{34} \end{bmatrix},$$

then A is a 3-by-2 block matrix whose entries are 4-by-4 block matrices whose entries are 3-by-4 matrices.

It is important to have a facility with the Kronecker product operation because they figure heavily in tensor computations. Here are three critical properties:

$$(A \otimes B)(C \otimes D) = AC \otimes BD$$

$$(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$$

$$(A \otimes B)^T = A^T \otimes B^T.$$

Of course, in these expressions the various matrix products and inversions have to be defined.

If the matrices A and B have structure, then their Kronecker product is typically structured in the same way. For example, if $Q_1 \in \mathbb{R}^{m_1 \times n_1}$ and $Q_2 \in \mathbb{R}^{m_2 \times n_2}$ have orthonormal columns, then $Q_1 \otimes Q_2$ has orthonormal columns:

$$\begin{aligned}(Q_1 \otimes Q_2)^T(Q_1 \otimes Q_2) &= (Q_1^T \otimes Q_2^T)(Q_1 \otimes Q_2) \\ &= (Q_1^T Q_1) \otimes (Q_2^T Q_2) = I_{n_1} \otimes I_{n_2} = I_{n_1 n_2}.\end{aligned}$$

Kronecker products often arise through the "vectorization" of a matrix equation, e.g.,

$$C = BXA^T \qquad \Leftrightarrow \qquad \mathrm{vec}(C) = (A \otimes B)\,\mathrm{vec}(X).$$

3.7 Perfect Shuffles, Kronecker Products, and Transposition

In general, $A_1 \otimes A_2 \neq A_2 \otimes A_1$. However, very structured permutation matrices P_1 and P_2 exist so that $P_1(A_1 \otimes A_2)P_2 = A_2 \otimes A_1$. Define the (p, q)-*perfect shuffle* matrix $\Pi_{p,q} \in \mathbb{R}^{pq \times pq}$ by

$$\Pi_{p,q} = \left[\, I_{pq}(:, 1 : q : pq) \mid I_{pq}(:, 2 : q : pq) \mid \cdots \mid I_{pq}(:, q : q : pq) \,\right].$$

Here is an example:

$$\Pi_{3,2} = \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right].$$

In general $A_1 \otimes A_2 \neq A_2 \otimes A_1$. However, if $A_1 \in \mathbb{R}^{m_1 \times n_1}$ and $A_2 \in \mathbb{R}^{m_2 \times n_2}$ then

$$\Pi_{m_1,m_2}(A_1 \otimes A_2)\Pi_{n_1,n_2}^T = A_2 \otimes A_1.$$

The perfect shuffle is also "behind the scenes" when the transpose of a matrix is taken, e.g.,

$$\Pi_{3,2}\,\mathrm{vec}(A) \;=\; \begin{bmatrix} 1&0&0&0&0&0\\ 0&0&0&1&0&0\\ 0&1&0&0&0&0\\ 0&0&0&0&1&0\\ 0&0&1&0&0&0\\ 0&0&0&0&0&1 \end{bmatrix} \begin{bmatrix} a_{11}\\ a_{21}\\ a_{31}\\ a_{12}\\ a_{22}\\ a_{32} \end{bmatrix} \;=\; \begin{bmatrix} a_{11}\\ a_{12}\\ a_{21}\\ a_{22}\\ a_{31}\\ a_{32} \end{bmatrix} \;=\; \mathrm{vec}(A^T).$$

In general, if $A \in \mathbb{R}^{m\times n}$ then $\Pi_{m,n}\mathrm{vec}(A) \;=\; \mathrm{vec}(A^T)$. See [12]. We return to these interconnections when we discuss tensor transposition in Sect. 6.

3.8 Tensor Notation

It is often perfectly adequate to illustrate a tensor computation idea using order-3 examples. For example, suppose $\mathcal{A} \in \mathbb{R}^{n_1\times n_2\times n_3}$, $X_1 \in \mathbb{R}^{m_1\times n_1}$, $X_2 \in \mathbb{R}^{m_2\times n_2}$, and $X_3 \in \mathbb{R}^{m_3\times n_3}$ are given and that we wish to compute

$$\mathcal{B}(i_1,i_2,i_3) \;=\; \sum_{j_1=1}^{n_1}\sum_{j_2=1}^{n_2}\sum_{j_3=1}^{n_3} \mathcal{A}(j_1,j_2,j_3)X_1(i_1,j_1)X_2(i_2,j_2)X_1(i_3,j_3)$$

where $1 \le i_1 \le m_1$, $1 \le i_2 \le m_2$, and $1 \le i_3 \le m_3$. Here we are using matrix-like subscript notation to spell out the definition of $\mathcal{B}$. We could probably use the same notation to describe the order-4 version of this computation. However, for higher-order cases we have to resort to the dot-dot-dot notation and it gets pretty unwieldy:

$$\mathcal{B}(i_1,\ldots,i_d) \;=\; \sum_{j_1=1}^{n_1}\sum_{j_2=1}^{n_2}\cdots\sum_{j_d=1}^{n_d} \mathcal{A}(j_1,\ldots,j_d)X_1(i_1,j_1)\cdots X_d(i_d,j_d).$$

$$1 \le i_1 \le m_1,\; 1 \le i_2 \le m_2, \ldots, 1 \le i_d \le m_d$$

One way to streamline the presentation of such a calculation is to "vectorize" the notation using bold font to indicate vectors of subscripts. Multiple summations can also be combined through vectorization. Thus, if

$$\mathbf{i} \;=\; [i_1,\ldots,i_d], \quad \mathbf{j} \;=\; [j_1,\ldots,j_d], \quad \mathbf{m} \;=\; [m_1,\ldots,m_d], \quad \mathbf{n} \;=\; [n_1,\ldots,n_d],$$

then the $\mathcal{B}$ tensor given above can be expressed as follows:

$$\mathcal{B}(\mathbf{i}) \;=\; \sum_{\mathbf{j}=\mathbf{1}}^{\mathbf{n}} \mathcal{A}(\mathbf{j})X_1(i_1,j_1)\cdots X_d(i_d,j_d), \qquad \mathbf{1} \le \mathbf{i} \le \mathbf{m}.$$

Here, $\mathbf{1} = [1,\ 1, \ldots,\ 1]$. As another example of this notation, if $\mathbf{n} = [n_1, \ldots, n_d\,]$ and $\mathcal{A} \in \mathbb{R}^{\mathbf{n}}$, then

$$\| \mathcal{A} \|_F \ = \ \sqrt{\sum_{\mathbf{i}=\mathbf{1}}^{\mathbf{n}} \mathcal{A}(\mathbf{i})^2}$$

is its Frobenius norm. We shall make use of this vectorized notation whenever it is necessary to hide detail and/or when we are working with tensors of arbitrary order.

Finally, it is handy to have a MATLAB "`reshape`" notation. Suppose $\mathbf{n} = [\,n_1, \ldots, n_d\,]$ and $\mathbf{m} = [\,m_1, \ldots, m_e\,]$. If $\mathcal{A} \in \mathbb{R}^{\mathbf{n}}$ and $n_1 \cdots n_d \ = \ m_1 \cdots m_e$, then

$$\mathcal{B} \ = \ \texttt{reshape}(\mathcal{A}, \mathbf{m})$$

is the $m_1 \times \cdots \times m_e$ tensor defined by $\text{vec}(\mathcal{A}) \ = \ \text{vec}(\mathcal{B})$.

3.9 The Tensor Product

On occasion it is handy to talk about operations between tensors without recasting the discussion in the language of matrices. Suppose $\mathbf{n} = [n_1, \ldots, n_d]$ and that $\mathcal{B}, \mathcal{C} \in \mathbb{R}^{\mathbf{n}}$. We can multiply a tensor by a scalar,

$$\mathcal{A} \ = \ \alpha\mathcal{B} \quad \Leftrightarrow \quad \mathcal{A}(\mathbf{i}) \ = \ \alpha\mathcal{B}(\mathbf{i}), \qquad \mathbf{1} \le \mathbf{i} \le \mathbf{n}$$

and we can add one tensor to another,

$$\mathcal{A} \ = \ \mathcal{B} + \mathcal{C} \quad \Leftrightarrow \quad \mathcal{A}(\mathbf{i}) \ = \ \mathcal{B}(\mathbf{i}) + \mathcal{C}(\mathbf{i}), \qquad \mathbf{1} \le \mathbf{i} \le \mathbf{n}.$$

Slightly more complicated is the *tensor product* which is a way of multiplying two tensors together to obtain a new, higher order tensor. For example if $\mathcal{B} \in \mathbb{R}^{m_1 \times m_2 \times m_3} \ = \ \mathbb{R}^{\mathbf{m}}$ and $\mathcal{C} \in \mathbb{R}^{n_1 \times n_2} \ = \ \mathbb{R}^{\mathbf{n}}$, then the tensor product $\mathcal{A} = \mathcal{B} \circ \mathcal{C}$ is defined by

$$\mathcal{A}(i_1, i_2, i_3, j_1, j_2) \ = \ \mathcal{B}(i_1, i_2, i_3)\mathcal{C}(j_1, j_2)$$

i.e., $\mathcal{A}(\mathbf{i}, \mathbf{j}) \ = \ \mathcal{B}(\mathbf{i})\,\mathcal{C}(\mathbf{j})$ for all $\mathbf{1} \le \mathbf{i} \le \mathbf{m}$ and $\mathbf{1} \le \mathbf{j} \le \mathbf{n}$.

If $\mathcal{B} \in \mathbb{R}^{\mathbf{m}}$ and $\mathcal{C} \in \mathbb{R}^{\mathbf{n}}$, then there is a connection between the tensor product $\mathcal{A} = \mathcal{B} \circ \mathcal{C}$ and its **m**-by-**n** unfolding:

$$\mathcal{A}_{\mathbf{m}\times\mathbf{n}} \ = \ \text{vec}(\mathcal{B})\text{vec}(\mathcal{C})^T.$$

There is also a connection between the tensor product of two vectors and their Kronecker product:

$$\text{vec}(x \circ y) \;=\; \text{vec}(xy^T) \;=\; y \otimes x.$$

Likewise, if $\mathcal{B}$ and $\mathcal{C}$ are order-2 tensors and $\mathcal{A} = \mathcal{B} \circ \mathcal{C}$, then

$$\mathcal{A}_{[1,3]\times[2,4]} \;=\; B \otimes C.$$

The point of all this notation-heavy discussion is to stress the importance of flexibility and point of view. Whether we write $\mathcal{A}(\mathbf{i})$ or $\mathcal{A}(i_1, i_2, i_3)$ or $a_{i_1 i_2 i_3}$ depends on the context, what we are trying to communicate, and what typesets nicely! Sometimes it will be handy to regard $\mathcal{A}$ as a vector such as $\text{vec}(\mathcal{A})$ and sometimes as a matrix such as $\mathcal{A}_{(3)}$. Algorithmic insights in tensor computations frequently require an ability to "reshape" how the problem at hand is viewed.

4 A Rank-1 Tensor Problem

Rank-1 matrices have a prominent role to play in matrix computations. For example, one step of Gaussian elimination involves a rank-1 update of a submatrix. The SVD decomposes a matrix into a sum of very special rank-1 matrices. Quasi-Newton methods for nonlinear systems involve rank-1 modifications of the current approximate Jacobian matrix.

In this section we introduce the concept of a rank-1 tensor and consider how we might approximate a given tensor with such an entity. This leads to a discussion (through an example) of multilinear optimization.

4.1 Rank-1 Matrices

If u and v are vectors, then $A = uv^T$ is a rank-1 matrix, e.g.,

$$A \;=\; \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix}^T \;=\; \begin{bmatrix} u_1 v_1 & u_1 v_2 \\ u_2 v_1 & u_2 v_2 \\ u_3 v_1 & u_3 v_2 \end{bmatrix}.$$

Note that if $A = uv^T$, then $\text{vec}(A) = v \otimes u$ and so we have

$$\begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \\ a_{12} \\ a_{22} \\ a_{32} \end{bmatrix} = \begin{bmatrix} u_1 v_1 \\ u_2 v_1 \\ u_3 v_1 \\ u_1 v_2 \\ u_2 v_2 \\ u_3 v_2 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \otimes \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}.$$

4.2 Rank-1 Tensors

How can we extend the rank-1 idea from matrices to tensors? In matrix computations we think of rank-1 matrices as *outer products*, i.e., $A = uv^T$ where u and v are vectors. Thinking of matrix A as tensor $\mathcal{A}$, we see that it is just the tensor product of u and v: $\mathcal{A}(i_1, i_2) = u(i_1)v(i_2)$. Thus, we have

$$\mathcal{A} = u \circ v = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \circ \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \quad \Leftrightarrow \quad \text{vec}(\mathcal{A}) = \begin{bmatrix} u_1 v_1 \\ u_2 v_1 \\ u_3 v_1 \\ u_1 v_2 \\ u_2 v_2 \\ u_3 v_2 \end{bmatrix} = v \otimes u.$$

Here is an order-3 example of the same idea:

$$\mathcal{A} = u \circ v \circ w = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} \circ \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \circ \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} \quad \Leftrightarrow \quad \text{vec}(\mathcal{A}) = \begin{bmatrix} u_1 v_1 w_1 \\ u_2 v_1 w_1 \\ u_3 v_1 w_1 \\ u_1 v_2 w_1 \\ u_2 v_2 w_1 \\ u_3 v_2 w_1 \\ u_1 v_1 w_2 \\ u_2 v_1 w_2 \\ u_1 v_2 w_2 \\ u_2 v_2 w_2 \\ u_3 v_2 w_2 \end{bmatrix} = w \otimes v \otimes u.$$

Each entry in $\mathcal{A}$ is a product of entries from u, v, and w: $\mathcal{A}(p, q, r) = u_p v_q w_r$.

In general, a rank-1 tensor is a tensor product of vectors. To be specific, if $x^{(i)} \in \mathbb{R}^{n_i}$ for $i = 1, \ldots, d$, then

$$\mathcal{A} = x^{(1)} \circ \cdots \circ x^{(d)}$$

is a rank-1 tensor whose entries are defined by $\mathcal{A}(\mathbf{i}) = x_{i_1}^{(1)} \cdots x_{i_d}^{(d)}$. In terms of the Kronecker product we have $\text{vec}(x^{(1)} \circ \cdots \circ x^{(d)}) = x^{(d)} \otimes \cdots \otimes x^{(1)}$.

4.3 The Nearest Rank-1 Problem for Matrices

Given a matrix $A \in \mathbb{R}^{m \times n}$, consider the minimization of

$$\phi(\sigma, u, v) = \| A - \sigma u v^T \|_F$$

where $u \in \mathbb{R}^m$ and $v \in \mathbb{R}^n$ have unit 2-norm and σ is a nonnegative scalar. This is an SVD problem for if $U^T A V = \Sigma = \text{diag}(\sigma_i)$ is the SVD of A and $\sigma_1 \geq \cdots \geq \sigma_n \geq 0$, then $\phi(\sigma, u, v)$ is minimized by setting $\sigma = \sigma_1$, $u = U(:, 1)$, and $v = V(:, 1)$.

Instead of explicitly computing the entire SVD, we can compute σ_1 and its singular vectors using an *alternating least squares* approach. The starting point is to realize that

$$\| A - \sigma u v^T \|_F^2 = \text{tr}(A^T A) - 2\sigma u^T A v + \sigma^2.$$

where $\text{tr}(M)$ indicates the trace of a matrix M, i.e., the sum of its diagonal entries. Note that

$$\phi_u(y) = \text{tr}(A^T A) - 2y^T A v + \| y \|_2^2$$

is minimized by setting $y = Av$ and that

$$\phi_v(x) = \text{tr}(A^T A) - 2u^T A x + \| x \|_2^2$$

is minimized by setting $x = A^T u$. This suggests the following iterative framework for minimizing $\| A - \sigma u v^T \|_F$:

Nearest Rank-1 Matrix

Given: $A \in \mathbb{R}^{m \times n}$, $v \in \mathbb{R}^n$, $\| v \|_2 = 1$

Repeat:

Fix v and choose σ and u to minimize $\| A - \sigma u v^T \|_F$:

$y = Av; \quad \sigma = \| y \|; \; u = y/\sigma$

Fix u and choose σ and v to minimize $\| A - \sigma u v^T \|_F$:

$x = A^T u; \; \sigma = \| x \|; \; v = x/\sigma$

$\sigma_{\text{opt}} = \sigma; \; u_{\text{opt}} = u; \; v_{\text{opt}} = v$

This is basically just the power method applied to the matrix

$$\text{sym}(A) \;=\; \begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix}.$$

The reason for bringing up this alternating least squares framework is that it readily extends to tensors.

4.4 A Nearest Rank-1 Tensor Problem

Given $\mathcal{A} \in \mathbb{R}^{m\times n\times p}$, we wish to determine unit vectors $u \in \mathbb{R}^m$, $v \in \mathbb{R}^n$, and $w \in \mathbb{R}^p$ and a scalar σ so that the following is minimized:

$$\| \mathcal{A} - \sigma \cdot u \circ v \circ w \|_F \;=\; \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{k=1}^{p}(a_{ijk} - u_i v_j w_k)^2}.$$

Noting that

$$\| \mathcal{A} - \sigma \cdot u \circ v \circ w \|_F \;=\; \| \text{vec}(\mathcal{A}) \;-\; \sigma \cdot w \;\otimes\; v \;\otimes\; u \|_2$$

we obtain the following alternating least squares framework:

Nearest Rank-1 Tensor

Given: $\mathcal{A} \in \mathbb{R}^{m\times n\times p}$ and unit vectors $v \in \mathbb{R}^n$ and $w \in \mathbb{R}^p$.

Repeat:

Determine $x \in \mathbb{R}^m$ that minimizes $\| \text{vec}(\mathcal{A}) \;-\; w \;\otimes\; v \;\otimes\; x \|_2$
and set $\sigma = \| x \|$ and $u = x/\sigma$.

Determine $y \in \mathbb{R}^n$ that minimizes $\| \text{vec}(\mathcal{A}) \;-\; w \;\otimes\; y \;\otimes\; u \|_2$
and set $\sigma = \| y \|$ and $v = y/\sigma$.

Determine $z \in \mathbb{R}^p$ that minimizes $\| \text{vec}(\mathcal{A}) \;-\; z \;\otimes\; v \;\otimes\; u \|_2$
and set $\sigma = \| z \|$ and $w = z/\sigma$.

$\sigma_{\text{opt}} = \sigma,\; u_{\text{opt}} = u,\; v_{\text{opt}} = v,\; w_{\text{opt}} = w$

It is instructive to examine some of the details associated with this iteration for the case $m = n = p = 2$. The objective function has the form

$$\phi(\sigma,\theta_1,\theta_2,\theta_3) = \left\| \begin{bmatrix} a_{111}\\ a_{211}\\ a_{121}\\ a_{221}\\ a_{112}\\ a_{212}\\ a_{122}\\ a_{222}\end{bmatrix} - \sigma\cdot w\otimes v\otimes u \right\| = \left\| \begin{bmatrix} a_{111}\\ a_{211}\\ a_{121}\\ a_{221}\\ a_{112}\\ a_{212}\\ a_{122}\\ a_{222}\end{bmatrix} - \sigma\cdot \begin{bmatrix} c_3c_2c_1\\ c_3c_2s_1\\ c_3s_2c_1\\ c_3s_2s_1\\ s_3c_2c_1\\ s_3c_2s_1\\ s_3s_2c_1\\ s_3s_2s_1\end{bmatrix} \right\|$$

where

$$u = \begin{bmatrix}\cos(\theta_1)\\ \sin(\theta_1)\end{bmatrix} = \begin{bmatrix} c_1\\ s_1\end{bmatrix},\quad v = \begin{bmatrix}\cos(\theta_2)\\ \sin(\theta_2)\end{bmatrix} = \begin{bmatrix} c_2\\ s_2\end{bmatrix},\quad w = \begin{bmatrix}\cos(\theta_3)\\ \sin(\theta_3)\end{bmatrix} = \begin{bmatrix} c_3\\ s_3\end{bmatrix}.$$

Let us look at the three structured *linear* least squares problems that arise during each iteration.

(1) To improve θ_1 and σ, we fix θ_2 and θ_3 and minimize

$$\left\| \begin{bmatrix} a_{111}\\ a_{211}\\ a_{121}\\ a_{221}\\ a_{112}\\ a_{212}\\ a_{122}\\ a_{222}\end{bmatrix} - \sigma\cdot \begin{bmatrix} c_3c_2c_1\\ c_3c_2s_1\\ c_3s_2c_1\\ c_3s_2s_1\\ s_3c_2c_1\\ s_3c_2s_1\\ s_3s_2c_1\\ s_3s_2s_1\end{bmatrix} \right\| = \left\| \begin{bmatrix} a_{111}\\ a_{211}\\ a_{121}\\ a_{221}\\ a_{112}\\ a_{212}\\ a_{122}\\ a_{222}\end{bmatrix} - \begin{bmatrix} c_3c_2 & 0\\ 0 & c_3c_2\\ c_3s_2 & 0\\ 0 & c_3s_2\\ s_3c_2 & 0\\ 0 & s_3c_2\\ s_3s_2 & 0\\ 0 & s_3s_2\end{bmatrix} \begin{bmatrix} x_1\\ y_1\end{bmatrix} \right\|$$

with respect to x_1 and y_1. We then set $\sigma = \sqrt{x_1^2 + y_1^2}$ and $u = [x_1\ y_1]^T/\sigma$.

(2) To improve θ_2 and σ, we fix θ_1 and θ_3 and minimize

$$\left\| \begin{bmatrix} a_{111}\\ a_{211}\\ a_{121}\\ a_{221}\\ a_{112}\\ a_{212}\\ a_{122}\\ a_{222}\end{bmatrix} - \sigma\cdot \begin{bmatrix} c_3c_2c_1\\ c_3c_2s_1\\ c_3s_2c_1\\ c_3s_2s_1\\ s_3c_2c_1\\ s_3c_2s_1\\ s_3s_2c_1\\ s_3s_2s_1\end{bmatrix} \right\| = \left\| \begin{bmatrix} a_{111}\\ a_{211}\\ a_{121}\\ a_{221}\\ a_{112}\\ a_{212}\\ a_{122}\\ a_{222}\end{bmatrix} - \begin{bmatrix} c_3c_1 & 0\\ c_3s_1 & 0\\ 0 & c_3c_1\\ 0 & c_3s_1\\ s_3c_1 & 0\\ s_3s_1 & 0\\ 0 & s_3c_1\\ 0 & s_3s_1\end{bmatrix} \begin{bmatrix} x_2\\ y_2\end{bmatrix} \right\|$$

with respect to x_2 and y_2. We then set $\sigma = \sqrt{x_2^2 + y_2^2}$ and $v = [x_2\ y_2]^T/\sigma$.

(3) To improve θ_3 and σ, we fix θ_1 and θ_2 and minimize

$$\left\| \begin{bmatrix} a_{111} \\ a_{211} \\ a_{121} \\ a_{221} \\ a_{112} \\ a_{212} \\ a_{122} \\ a_{222} \end{bmatrix} - \sigma \cdot \begin{bmatrix} c_3c_2c_1 \\ c_3c_2s_1 \\ c_3s_2c_1 \\ c_3s_2s_1 \\ s_3c_2c_1 \\ s_3c_2s_1 \\ s_3s_2c_1 \\ s_3s_2s_1 \end{bmatrix} \right\| = \left\| \begin{bmatrix} a_{111} \\ a_{211} \\ a_{121} \\ a_{221} \\ a_{112} \\ a_{212} \\ a_{122} \\ a_{222} \end{bmatrix} - \begin{bmatrix} c_2c_1 & 0 \\ c_2s_1 & 0 \\ s_2c_1 & 0 \\ s_2s_1 & 0 \\ 0 & c_2s_1 \\ 0 & c_2s_1 \\ 0 & s_2c_1 \\ 0 & s_2s_1 \end{bmatrix} \begin{bmatrix} x_3 \\ y_3 \end{bmatrix} \right\|$$

with respect to x_3 and y_3. We then set $\sigma = \sqrt{x_3^2 + y_3^2}$ and $w = [x_3\ y_3]^T/\sigma$.

Componentwise optimization is a common framework for many tensor-related computations. The basic idea is to choose a subset of the unknowns and (temporarily) freeze their value. This leads to a simplified optimization problem involving the other unknowns. The process is repeated using different subsets of unknowns each iteration until convergence. The framework is frequently successful, but there is a tendency for the iterates to get trapped near an uninteresting local minima.

5 The Variational Approach to Tensor Singular Values

If μ^2 is a zero of the characteristic polynomial $p(\lambda) = \det(A^TA - \lambda I)$, then μ is a singular value of A and the associated left and right singular vectors are eigenvectors for AA^T and A^TA respectively. How can we extend these notions to tensors? Is there a version of the characteristic polynomial that makes sense for tensors? What would be the analog of the matrices AA^T and A^TA? These are tough questions. Fortunately, there is a constructive way to avoid these difficulties and that is to take a *variational approach*. Singular values and vectors are solutions to a very tractable optimization problem.

5.1 Rayleigh Quotient/Power Method Ideas: The Matrix Case

The singular values and singular vectors of a general matrix $A \in \mathbb{R}^{m\times n}$ are the *stationary values and vectors* of the *Rayleigh quotient*

$$\frac{y^TAx}{\| x \|_2 \| y \|_2}.$$

It is slightly more convenient to pose this as a constrained optimization problem: singular values and singular vectors of a general matrix $A \in \mathbb{R}^{m\times n}$ are the stationary values and vectors of

$$\psi_A(x,y) \;=\; x^TAy \;=\; \sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}x_iy_j$$

subject to the constraints $\| x \|_2 \;=\; \| y \|_2 \;=\; 1$. To connect this "definition" to the SVD we use the method of Lagrange multipliers and that means looking at the gradient of

$$\tilde{\psi}_A(x,y) = \psi(x,y) - \frac{\lambda}{2}(x^Tx-1) - \frac{\mu}{2}(y^Ty-1).$$

Using the rearrangements

$$\psi_A(x,y) \;=\; \sum_{i=1}^{m} x_i\left(\sum_{j=1}^{n} a_{ij}y_j\right) \;=\; \sum_{j=1}^{n} y_j\left(\sum_{i=1}^{m} a_{ij}x_i\right),$$

it follows that

$$\nabla\tilde{\psi}_A(x,y) \;=\; \begin{bmatrix} Ay \;-\; \lambda x \\ A^Tx \;-\; \mu y \end{bmatrix}. \tag{4}$$

From the vector equation $\nabla\tilde{\psi}_A(x,y) \;=\; 0$ we conclude that $\lambda \;=\; \mu \;=\; x^TAy \;=\; \psi_A(x,y)$ and that x and y satisfy $Ay \;=\; (x^TAy)x$ and $A^Tx \;=\; (x^TAy)y$. That is to say, x is an eigenvector of A^TA and y is an eigenvector of AA^T and the associated eigenvalue in each case is $(y^TAx)^2$. These are exactly the conclusions that can be reached by equating columns in the SVD equation $AV = U\Sigma$. Indeed, from

$$A[\,v_1 \mid \cdots \mid v_n\,] \;=\; [\,u_1 \mid \cdots \mid u_n\,]\,\mathrm{diag}(\sigma_1,\ldots,\sigma_n)$$

we see that $Av_i \;=\; \sigma_iu_i$, $A^Tu_i = \sigma_iv_i$, and $\sigma_i = u_i^TAv_i$, for $i = 1:n$.

The power method for matrices can be designed to go after the largest singular value and associated singular vectors:

Power Method for Matrix Singular Values and Vectors

Given $A \in \mathbb{R}^{m\times n}$ and unit vector $y \in \mathbb{R}^n$.

Repeat:

$$\tilde{x} = Ay,\ x = \tilde{x}/\| \tilde{x} \|$$
$$\tilde{y} = A^T x,\ y = \tilde{y}/\| \tilde{y} \|$$
$$\sigma = \psi_A(x,y) = y^T A x$$

$\sigma_{opt} = \sigma,\ u_{opt} = y,\ v_{opt} = x$

This can be viewed as an alternating procedure for finding a zero for the gradient (4). Under mild assumptions, $\{\sigma_{opt}, u_{opt}, v_{opt}\}$ will approximate the largest singular value of A and the corresponding left and right singular vectors. In principle, deflation can be used to find other singular value triplets. Thus, by applying the power method to $A - \sigma_1 u_1 v_1^T$ we could obtain an estimate of $\{\sigma_2, u_2, v_2\}$.

5.2 Rayleigh Quotient/Power Method Ideas: The Tensor Case

Let us extend the Rayleigh quotient characterization for matrix singular values and vectors to tensors. We work out the order-3 situation for simplicity.

If $\mathcal{A} \in \mathbb{R}^{n_1\times n_2\times n_3}$, $x \in \mathbb{R}^{n_1}$, $y \in \mathbb{R}^{n_2}$, and $z \in \mathbb{R}^{n_3}$, then the singular values and vectors of $\mathcal{A}$ are the stationary values and vectors of

$$\psi_{\mathcal{A}}(x,y,z) = \sum_{i=1}^{n_1}\sum_{j=1}^{n_2}\sum_{k=1}^{n_3} a_{ijk}\, x_i y_j z_k \tag{5}$$

subject to the constraints $\| x \|_2 = \| y \|_2 = \| z \|_2 = 1$. Before we start taking gradients we present three alternative formulations of this summation. Each highlights a different unfolding of the tensor $\mathcal{A}$.

The Mode-1 Formulation:

$$\psi_{\mathcal{A}}(x,y,z) = \sum_{i=1}^{n_1} x_i \left(\sum_{j=1}^{n_2}\sum_{k=1}^{n_3} a_{ijk}\, y_j z_k \right) = x^T \mathcal{A}_{(1)} z \otimes y \tag{6}$$

where $\mathcal{A}_{(1)}(i, j + (k-1)n_2) = a_{ijk}$. For the case $\mathbf{n} = [4, 3, 2]$ we have

$$\mathcal{A}_{(1)} = \begin{bmatrix} a_{111} & a_{121} & a_{131} & a_{112} & a_{122} & a_{132} \\ a_{211} & a_{221} & a_{231} & a_{212} & a_{222} & a_{232} \\ a_{311} & a_{321} & a_{331} & a_{312} & a_{322} & a_{332} \\ a_{411} & a_{421} & a_{431} & a_{412} & a_{422} & a_{432} \end{bmatrix}.$$

$$\begin{matrix} (1,1) & (2,1) & (3,1) & (1,2) & (2,2) & (3,2) \end{matrix}$$

The Mode-2 Formulation:

$$\psi_{\mathcal{A}}(x, y, z) = \sum_{j=1}^{n_2} y_j \left(\sum_{i=1}^{n_1} \sum_{k=1}^{n_3} a_{ijk}\, x_i z_k \right) = y^T \mathcal{A}_{(2)} z \otimes x \tag{7}$$

where $\mathcal{A}_{(2)}(j, i + (k-1)n_1) = a_{ijk}$. For the case $\mathbf{n} = [4, 3, 2]$ we have

$$\mathcal{A}_{(2)} = \begin{bmatrix} a_{111} & a_{211} & a_{311} & a_{411} & a_{112} & a_{212} & a_{312} & a_{412} \\ a_{121} & a_{221} & a_{321} & a_{421} & a_{122} & a_{222} & a_{322} & a_{422} \\ a_{131} & a_{231} & a_{331} & a_{431} & a_{132} & a_{232} & a_{332} & a_{432} \end{bmatrix}.$$

$$\begin{matrix} (1,1) & (2,1) & (3,1) & (4,1) & (1,2) & (2,2) & (3,2) & (4,2) \end{matrix}$$

The Mode-3 Formulation:

$$\psi_{\mathcal{A}}(x, y, z) = \sum_{k=1}^{p} z_k \left(\sum_{i=1}^{m} \sum_{j=1}^{n} a_{ijk}\, x_i y_j \right) = z^T \mathcal{A}_{(3)} y \otimes x \tag{8}$$

where $\mathcal{A}_{(3)}(k, i + (j-1)n_1) = a_{ijk}$. For the case $\mathbf{n} = [4, 3, 2]$ we have

$$\mathcal{A}_{(3)} = \begin{bmatrix} a_{111} & a_{211} & a_{311} & a_{411} & a_{121} & a_{221} & a_{321} & a_{421} & a_{131} & a_{231} & a_{331} & a_{431} \\ a_{112} & a_{212} & a_{312} & a_{412} & a_{122} & a_{222} & a_{322} & a_{422} & a_{132} & a_{232} & a_{332} & a_{432} \end{bmatrix}.$$

$$\begin{matrix} (1,1) & (2,1) & (3,1) & (4,1) & (1,2) & (2,2) & (3,2) & (4,2) & (1,3) & (2,3) & (3,3) & (4,3) \end{matrix}$$

The matrices $\mathcal{A}_{(1)}$, $\mathcal{A}_{(2)}$, and $\mathcal{A}_{(3)}$ are the mode-1, mode-2, and mode-3 unfoldings of $\mathcal{A}$ that we introduced in Sect. 3.4. It is handy to identify columns with multi-indices as we have shown.

We return to the constrained minimization of the objective function $\psi_{\mathcal{A}}(x, y, z))$ that is defined in (5). Using the method of Lagrange multipliers we set the gradient of

$$\tilde{\psi}_{\mathcal{A}}(x, y, z) = \psi_{\mathcal{A}}(x, y, z) - \frac{\lambda}{2}(x^T x - 1) - \frac{\mu}{2}(y^T y - 1) - \frac{\tau}{2}(z^T z - 1)$$

to zero. Using (6)–(8) we get

$$\nabla\tilde{\psi}_{\mathcal{A}} \;=\; \begin{bmatrix} \mathcal{A}_{(1)}(z \otimes y) - \lambda x \\ \mathcal{A}_{(2)}(z \otimes x) - \mu y \\ \mathcal{A}_{(3)}(y \otimes x) - \tau z \end{bmatrix} \;=\; \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}. \tag{9}$$

Since x, y, and z are unit vectors it follows that $\lambda = \mu = \tau = \psi(x, y, z)$. In this case we say that $\sigma = \psi(x, y, z)$ is a singular value of $\mathcal{A}$ and x, y, and z are the associated singular vectors. How might we solve this (highly structured) system of nonlinear equations? The triplet of matrix-vector products:

$$A_{(1)} \cdot (z \otimes y) \;=\; \sigma \cdot x \quad A_{(2)} \cdot (z \otimes x) \;=\; \sigma \cdot y \quad A_{(3)} \cdot (y \otimes x) \;=\; \sigma \cdot z$$

suggests a componentwise solution strategy:

Power Method for Tensor Singular Values and Vectors

Given: $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and unit vectors $y \in \mathbb{R}^{n_2}$ and $z \in \mathbb{R}^{n_3}$.

Repeat:

$$\begin{aligned} \tilde{x} &= \mathcal{A}_{(1)}(z \otimes y), \quad \sigma = \| \tilde{x} \|, \; x = \tilde{x}/\sigma \\ \tilde{y} &= \mathcal{A}_{(2)}(z \otimes x), \quad \sigma = \| \tilde{y} \|, \; y = \tilde{y}/\sigma \\ \tilde{z} &= \mathcal{A}_{(3)}(y \otimes x), \quad \sigma = \| \tilde{z} \|, \; z = \tilde{z}/\sigma \end{aligned}$$

$\sigma_{opt} = \sigma$, $x_{opt} = x$, $y_{opt} = y$, $z_{opt} = z$

See [7, 17] for details.

For matrices, the SVD expansion

$$A \;=\; U\Sigma V^T \;=\; \sum_{i=1}^{\operatorname{rank}(A)} \sigma_i u_i v_i^T$$

has an important optimality property. In particular, the Eckhart-Young theorem tells us that

$$A_r \;=\; \sum_{i=1}^{r} \sigma_i u_i v_i^T \qquad r \leq \operatorname{rank}(A)$$

is the closest rank-r matrix to A in either the 2-norm or Frobenius norm. Moreover, the closest rank-1 matrix to A_{r-1} is $\sigma_r u_r v_r^T$. Thus, it would be possible (in principle) to compute the full SVD by solving a sequence of closest rank-1 matrix problems.

This idea does *not* work for tensors. In other words, if $\sigma_1 u_1 \circ v_1 \circ w_1$ is the closest rank-1 tensor to $\mathcal{A}$ and $\sigma_2 u_1 \circ v_2 \circ w_2$ is the closest rank-1 tensor to

$$\tilde{\mathcal{A}} \;=\; \mathcal{A} \;-\; \sigma_1 u_1 \circ v_1 \circ w_1,$$

then

$$\mathcal{A}_2 \;=\; \sigma_1 u_1 \circ v_1 \circ w_1 + \sigma_2 u_2 \circ v_2 \circ w_2$$

is *not* necessarily the closest rank-2 tensor to $\mathcal{A}$. We need an alternative approach to formulating a "tensor SVD".

5.3 A First Look at Tensor Rank

Since matrix rank ideas do not readily extend to the tensor setting, we should look more carefully at tensor rank to appreciate the "degree of difficulty" associated with the formulation of illuminating low-rank tensor expansions.

We start with a definition. Suppose the tensor $\mathcal{A}$ can be written as the sum of r rank-1 tensors and that r is minimal in this regard. In this case we say that $\text{rank}(\mathcal{A}) = r$. Let us explore this concept in the simplest possible setting: $\mathcal{A} \in \mathbb{R}^{2\times 2\times 2}$. For this problem the goal is to find three thin-as-possible matrices $X, Y, Z \in \mathbb{R}^{2\times r}$ so that

$$\mathcal{A} \;=\; \sum_{k=1}^{r} X(:,k) \circ Y(:,k) \circ Z(:,k), \tag{10}$$

i.e.,

$$\text{vec}(\mathcal{A}) \;=\; \begin{bmatrix} a_{111} \\ a_{211} \\ a_{121} \\ a_{221} \\ a_{112} \\ a_{212} \\ a_{122} \\ a_{222} \end{bmatrix} \;=\; \sum_{k=1}^{r} Z(:,k) \;\otimes\; Y(:,k) \;\otimes\; X(:,k).$$

This vector equation can also be written as a pair of matrix equations:

$$\mathcal{A}(:,1) = \begin{bmatrix} a_{111} & a_{121} \\ a_{211} & a_{221} \end{bmatrix} = \sum_{k=1}^{r} Z(1,k)X(:,k)Y(:,k)^T$$

$$\mathcal{A}(:,2) = \begin{bmatrix} a_{112} & a_{122} \\ a_{212} & a_{222} \end{bmatrix} = \sum_{k=1}^{r} Z(2,k)X(:,k)Y(:,k)^T.$$

Readers familiar with the generalized eigenvalue problem should see a connection to our 2-by-2-by-2 rank($\mathcal{A}$) problem. Indeed, it can be shown that

$$\det\left(\begin{bmatrix} a_{111} & a_{121} \\ a_{211} & a_{221} \end{bmatrix} - \lambda \begin{bmatrix} a_{112} & a_{122} \\ a_{212} & a_{222} \end{bmatrix}\right) = 0$$

has real distinct roots with probability 0.79 and complex conjugate roots with probability 0.21 when the matrix entries are randomly selected using the MATLAB `randn` function. If this 2-by-2 generalized eigenvalue problem has real distinct eigenvalues, then it is possible to find nonsingular matrices S and T so that

$$\begin{bmatrix} a_{111} & a_{121} \\ a_{211} & a_{221} \end{bmatrix} = S \begin{bmatrix} \alpha_1 & 0 \\ 0 & \alpha_2 \end{bmatrix} T^T$$

$$\begin{bmatrix} a_{112} & a_{122} \\ a_{212} & a_{222} \end{bmatrix} = S \begin{bmatrix} \beta_1 & 0 \\ 0 & \beta_2 \end{bmatrix} T^T.$$

This shows that the rank-1 expansion (10) for $\mathcal{A}$ holds with $r = 2$, $X = S$, $Y = T$ and

$$Z = \begin{bmatrix} \alpha_1 & \alpha_2 \\ \beta_1 & \beta_2 \end{bmatrix}.$$

Thus, for 2-by-2-by-2 tensors, rank equals two with probability about 0.79. A similar generalized eigenvalue analysis shows that the rank is three with probability 0.21. This is a very different situation than with matrices where an n-by-n matrix has rank n with probability 1. The subtleties associated with tensor rank are discussed further in Sect. 8.3.

6 Tensor Symmetry

Symmetry for a matrix is defined through transposition: $A = A^T$. This is a nice shorthand way of saying that $A(i,j) = A(j,i)$ for all possible i and j that satisfy $1 \le i \le n$ and $1 \le j \le n$.

How do we extend this notion to tensors? Transposition moves indices around so we need a way of talking about what happens when we (say) interchange $\mathcal{A}(i,j,k)$ with $\mathcal{A}(j,i,k)$ or $\mathcal{A}(k,j,i)$ or $\mathcal{A}(i,k,j)$ or $\mathcal{A}(j,k,i)$ or $\mathcal{A}(k,i,j)$. It looks like we will have to contend with an exponential number of transpositions and an exponential number of partial symmetries.

6.1 Tensor Transposition

If $\mathcal{C} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, then there are $3! = 6$ possible transpositions identified by the notation $\mathcal{C}^{<[i\,j\,k]>}$ where $[i\,j\,k]$ is a permutation of $[1\ 2\ 3]$:

$$\mathcal{B} = \begin{Bmatrix} \mathcal{C}^{<[1\ 2\ 3]>} \\ \mathcal{C}^{<[1\ 3\ 2]>} \\ \mathcal{C}^{<[2\ 1\ 3]>} \\ \mathcal{C}^{<[2\ 3\ 1]>} \\ \mathcal{C}^{<[3\ 1\ 2]>} \\ \mathcal{C}^{<[3\ 2\ 1]>} \end{Bmatrix} \quad \Longrightarrow \quad \begin{Bmatrix} b_{ijk} \\ b_{ikj} \\ b_{jik} \\ b_{jki} \\ b_{kij} \\ b_{kji} \end{Bmatrix} = c_{ijk}$$

for $i = 1:n_1,\ j = 1:n_2,\ k = 1:n_3$.

For order-d tensors there are $d!$ possibilities. Suppose $\mathbf{v} = [v_1, v_2, \ldots, v_d]$ is a permutation of the integer vector $1:d = [1, 2, \ldots, d]$. If $\mathcal{C} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, then

$$\mathcal{B} = \mathcal{C}^{<\mathbf{v}>} \qquad \Rightarrow \qquad \mathcal{B}(\mathbf{i}(\mathbf{v})) = \mathcal{C}(\mathbf{i}) \qquad \mathbf{1} \le \mathbf{i} \le \mathbf{n}.$$

6.2 Symmetric Tensors

An order-d tensor $\mathcal{C} \in \mathbb{R}^{n \times \cdots \times n}$ is *symmetric* if $\mathcal{C}^{<\mathbf{v}>} = \mathcal{C}$ for all permutations $\mathbf{v}$ of $1:d$. If $d = 3$ this means that

$$c_{ijk} = c_{ikj} = c_{jik} = c_{jki} = c_{kij} = c_{kji}.$$

To get a feel for the level of redundancy in a symmetric tensor, we see that a symmetric $\mathcal{C} \in \mathbb{R}^{3\times3\times3}$ has at most ten distinct values:

$$\begin{aligned}
&c_{111}\\
&c_{112} = c_{121} = c_{211}\\
&c_{113} = c_{131} = c_{311}\\
&c_{222}\\
&c_{221} = c_{212} = c_{122}\\
&c_{223} = c_{232} = c_{322}\\
&c_{333}\\
&c_{331} = c_{313} = c_{133}\\
&c_{332} = c_{323} = c_{233}\\
&c_{123} = c_{132} = c_{213} = c_{231} = c_{312} = c_{321}.
\end{aligned}$$

The modal unfoldings of a symmetric tensor are all the same. For example, the (2,3) entry in each of the matrices

$$C_{(1)} = \begin{bmatrix} c_{111} & c_{121} & c_{131} & c_{112} & c_{122} & c_{132} & c_{113} & c_{123} & c_{133} \\ c_{211} & c_{221} & c_{231} & c_{212} & c_{222} & c_{232} & c_{213} & c_{223} & c_{233} \\ c_{311} & c_{321} & c_{331} & c_{312} & c_{322} & c_{332} & c_{113} & c_{323} & c_{333} \end{bmatrix}$$

$$C_{(2)} = \begin{bmatrix} c_{111} & c_{211} & c_{311} & c_{112} & c_{212} & c_{312} & c_{113} & c_{213} & c_{313} \\ c_{121} & c_{221} & c_{321} & c_{122} & c_{222} & c_{322} & c_{123} & c_{223} & c_{323} \\ c_{131} & c_{231} & c_{331} & c_{132} & c_{232} & c_{332} & c_{113} & c_{233} & c_{333} \end{bmatrix}$$

$$C_{(3)} = \begin{bmatrix} c_{111} & c_{211} & c_{311} & c_{121} & c_{221} & c_{321} & c_{131} & c_{231} & c_{331} \\ c_{112} & c_{212} & c_{312} & c_{122} & c_{222} & c_{322} & c_{132} & c_{232} & c_{332} \\ c_{113} & c_{213} & c_{313} & c_{123} & c_{223} & c_{323} & c_{133} & c_{233} & c_{333} \end{bmatrix}$$

are equal: $c_{231} = c_{321} = c_{312}$.

6.3 Symmetric Rank

An order-d symmetric rank-1 tensor $\mathcal{C} \in \mathbb{R}^{n\times\cdots\times n}$ has the form

$$\mathcal{C} = \underbrace{x \circ \cdots \circ x}_{d \text{ times}}$$

where $x \in \mathbb{R}^n$. In this case we clearly have

$$\mathcal{C}(i_1, \ldots, i_d) \;=\; x_{i_1} x_{i_2} \cdots x_{i_d}$$

and

$$\mathrm{vec}(\mathcal{C}) \;=\; \underbrace{x \otimes \cdots \otimes x}_{d\,\text{times}}\,.$$

An order-3 symmetric tensor $\mathcal{C}$ has *symmetric rank r* if there exists $x_1, \ldots, x_r \in \mathbb{R}^n$ and $\sigma \in \mathbb{R}^r$ such that

$$\mathcal{C} \;=\; \sum_{k=1}^{r} \sigma_k \cdot x_k \circ x_k \circ x_k$$

and no shorter sum of symmetric rank-1 tensors exists. Symmetric rank is denoted by $\mathrm{rank}_S(\mathcal{C})$. Note, in contrast to what we would expect for matrices, there may be a shorter sum of general rank-1 tensors that add up to $\mathcal{C}$:

$$\mathcal{C} \;=\; \sum_{k=1}^{\tilde{r}} \tilde{\sigma}_k \cdot \tilde{x}_k \circ \tilde{y}_k \circ \tilde{z}_k.$$

The symmetric rank of a symmetric tensor is more tractable than the (general) rank of a general tensor. For example, if $\mathcal{C} \in \mathbb{C}^{n\times\cdots\times n}$ is an order-d symmetric tensor, then with probability one we have

$$\mathrm{rank}_S(\mathcal{C}) \;=\; \begin{cases} f(d,m)+1 \text{ if (d,n)= } (3,5),(4,3),(4,4), or (4,5) \\ \\ f(d,n) \qquad \text{otherwise} \end{cases}$$

where

$$f(d,n) \;=\; \mathrm{ceil}\left(\frac{\binom{n+d-1}{d}}{n} \right).$$

See [4, 5] for deeper discussions of symmetric rank.

6.4 The Eigenvalues of a Symmetric Tensor

For a symmetric matrix C the stationary values of $\phi_C(x) = x^T Cx$ subject to the constraint that $\| x \|_2 = 1$ are the eigenvalues of C. The associated stationary vectors are eigenvectors. We extend this idea to symmetric tensors and the order-3 case is good enough to illustrate the main ideas.

If $\mathcal{C} \in \mathbb{R}^{n\times n\times n}$ is a symmetric tensor, then we define the stationary values of

$$\phi_{\mathcal{C}}(x) = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} c_{ijk}x_i x_j x_k = x^T C_{(1)}(x \otimes x)$$

subject to the constraint that $\| x \|_2 = 1$ to be the eigenvalues of $\mathcal{C}$. The associated stationary vectors are eigenvectors. Using the method of Lagrange multipliers it can be shown that if x is a stationary vector for $\phi_{\mathcal{C}}$ then

$$x = \phi_{\mathcal{C}}(x)C_{(1)}(x \otimes x)$$

This leads to an iteration of the following form:

Power Method for Tensor Eigenvalues and Eigenvectors

Given: Symmetric $\mathcal{C} \in \mathbb{R}^{n\times n\times n}$ and unit vector $x \in \mathbb{R}^n$.

Repeat:

$$\tilde{x} = \mathcal{C}_{(1)}(x \otimes x)$$
$$\lambda = \| \tilde{x} \|_2$$
$$x = \tilde{x}/\lambda$$

$\lambda_{opt} = \lambda, x_{opt} = x.$

There are some convergence results for this iteration, For example, it can be shown that if the order of $\mathcal{C}$ is even and M is a square unfolding, then the iteration converges if M is positive definite [14].

6.5 Symmetric Embeddings

In the matrix case there are connections between the singular values and vectors of $A \in \mathbb{R}^{n_1\times n_2}$ and the eigenvalues and vectors of

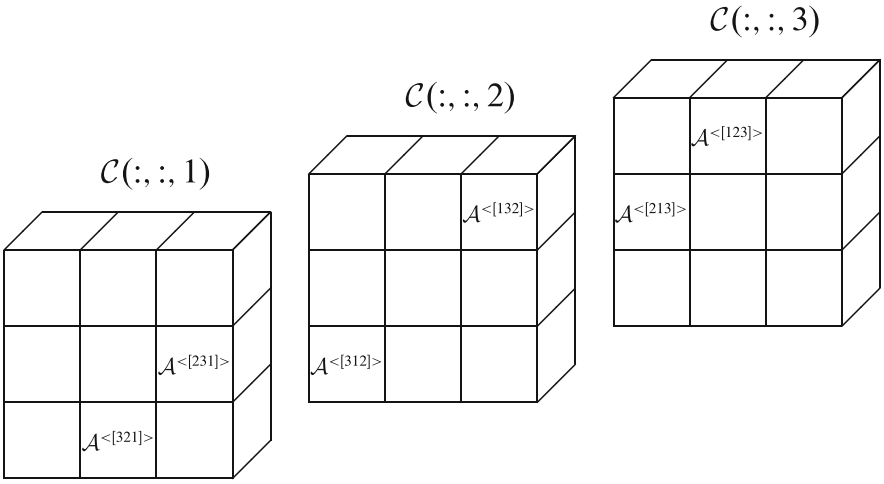
$$\text{sym}(A) = \begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix} \in \mathbb{R}^{(n_1+n_2)\times(n_1+n_2)}$$

If $A = U \cdot \mathrm{diag}(\sigma_i) \cdot V^T$ is the SVD of $A \in \mathbb{R}^{n_1 \times n_2}$, then for $k = 1 : \mathrm{rank}(A)$

$$\begin{bmatrix} 0 & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} u_k \\ \pm v_k \end{bmatrix} = \pm\sigma_k \begin{bmatrix} u_k \\ \pm v_k \end{bmatrix}$$

where $u_k = U(:,k)$ and $v_k = V(:,k)$.

It turns out that symmetric tensor $\mathrm{sym}(\mathcal{A})$ can be "built" from of a general tensor $\mathcal{A}$ by judiciously positioning $\mathcal{A}$ and all of its transposes. Here is a depiction of the order-3 case:



We can think of $\mathrm{sym}(\mathcal{A})$ as a 3-by-3-by-3 block tensor. As a tensor of scalars, it is N-by-N-by-N where $N = n_1 n_2 n_3$. If

$$\left\{ \sigma, \begin{bmatrix} u \\ v \\ z \end{bmatrix} \right\}$$

is a stationary pair for $\mathrm{sym}(\mathcal{A})$, then so are

$$\left\{ \sigma, \begin{bmatrix} u \\ -v \\ -z \end{bmatrix} \right\}, \quad \left\{ -\sigma, \begin{bmatrix} u \\ -v \\ z \end{bmatrix} \right\}, \quad \left\{ -\sigma, \begin{bmatrix} u \\ v \\ -z \end{bmatrix} \right\}.$$

This is a nice generalization of the result for matrices. There are interesting connections between power methods with $\mathcal{A}$ and power methods with $\mathrm{sym}(\mathcal{A})$. There are also connections between the rank of $\mathcal{A}$ and the symmetric rank of

sym($\mathcal{A}$):

$$d!\ \mathrm{rank}(\mathcal{A}) \ \leq\ \mathrm{rank}_S(\mathrm{sym}(\mathcal{A})).$$

It is not clear if the inequality can be replaced by equality. See [26].

7 The Tucker Decomposition

We have seen that it is possible to extend the definition of singular values and vectors to tensors by using variational principles. However, these insights did not culminate in the production of a tensor SVD and that is disappointing when we consider the power of the matrix SVD:

1. It can turn a given problem into an equivalent easy-to-solve problem. For example, the $\min \| Ax - b \|_2$ problem can be converted into an equivalent diagonal least squares problem using the SVD.
2. It can uncover hidden relationships that exist in matrix-encoded data. For example, the SVD can show that a data matrix has a low rank structure.

In the next several sections we produce various SVD-like tensor decompositions. We start with the Tucker decomposition because it involves orthogonality and has a strong resemblance to the matrix SVD. We use order-3 tensors to illustrate the main ideas.

7.1 Tucker Representations: The Matrix Case

Given a matrix $A \in \mathbb{R}^{n_1 \times n_2}$, the Tucker representation problem involves finding a *core matrix* $S \in \mathbb{R}^{r_1 \times r_2}$ and matrices $U_1 \in \mathbb{R}^{n_1 \times r_1}$ and $U_2 \in \mathbb{R}^{n_2 \times r_2}$ such that

$$A(i_1, i_2) \ = \sum_{j_1=1}^{r_1} \sum_{j_2=1}^{r_2} \mathcal{S}(j_1, j_2) \cdot U_1(i_1, j_1) \cdot U_2(i_2, j_2).$$

Part of the "deal" involves choosing the integers r_1 and r_2 and perhaps replacing the "=" with "≈". Regardless, it is possible to reformulate the right hand side as a sum of rank-1 matrices,

$$A \ = \sum_{j_1=1}^{r_1} \sum_{j_2=1}^{r_2} \mathcal{S}(j_1, j_2) \cdot U_1(:, j_1) \cdot U_2(:, j_2)^T, \tag{11}$$

or as a sum of vector Kronecker products,

$$\text{vec}(A) \;=\; \sum_{j_1=1}^{r_1} \sum_{j_2=1}^{r_2} S(j_1,j_2) U_2(:,j_2) \;\otimes\; U_1(:,j_2),$$

or as a single matrix-vector product,

$$\text{vec}(A) \;=\; (U_2 \;\otimes\; U_1) \cdot \text{vec}(S).$$

The tensor versions of these reformulations get us to think the right way about how we might generalize the matrix SVD.

7.2 Tucker Representations: The Tensor Case

Given a tensor $A \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, the Tucker representation problem involves finding a *core tensor* $\mathcal{S} \in \mathbb{R}^{r_1 \times r_2 \times r_3}$ and matrices $U_1 \in \mathbb{R}^{n_1 \times r_1}$, $U_2 \in \mathbb{R}^{n_2 \times r_2}$, and $U_3 \in \mathbb{R}^{n_3 \times r_3}$ such that

$$\mathcal{A}(i_1,i_2,i_3) \;=\; \sum_{j_1=1}^{r_1} \sum_{j_2=1}^{r_2} \sum_{j_3=1}^{r_3} \mathcal{S}(j_1,j_2,j_3) \cdot U_1(i_1,j_1) \cdot U_2(i_2,j_2) \cdot U_3(i_3,j_3).$$

As in the matrix case above, we can rewrite this as a sum of rank-1 tensors,

$$\mathcal{A} \;=\; \sum_{j_1=1}^{r_1} \sum_{j_2=1}^{r_2} \sum_{j_3=1}^{r_3} \mathcal{S}(j_1,j_2,j_3) \cdot U_1(:,j_1) \circ U_2(:,j_2) \circ U_3(:,j_3),$$

or as the sum of vector Kronecker products,

$$\text{vec}(\mathcal{A}) \;=\; \sum_{j_1=1}^{r_1} \sum_{j_2=1}^{r_2} \sum_{j_3=1}^{r_3} \mathcal{S}(j_1,j_2,j_3) \cdot U_3(:,j_3) \;\otimes\; U_2(:,j_2) \;\otimes\; U_1(:,j_1)$$

or as a single matrix-vector product,

$$\text{vec}(\mathcal{A}) \;=\; (U_3 \;\otimes\; U_2 \;\otimes\; U_1) \cdot \text{vec}(\mathcal{S}).$$

The challenge is to design the representation so that it is illuminating and computable.

Before we proceed it is instructive to revisit the matrix case. If we set $r_1 = n_1$ and $r_2 = n_2$ in (11) and assume the U matrices are orthogonal, then the Tucker

representation has the form

$$A = U_1(U_1^T A U_2)U_2^T = \sum_{j_1=1}^{n_1}\sum_{j_2=1}^{n_2} S(j_1,j_2)U_1(:,j_1)U_2(:,j_2)^T$$

where $S = U_1^T A U_2$. To make this an "illuminating" representation of A we strive for a diagonal core matrix S and that, of course, leads to the SVD:

$$A = \sum_{k=1}^{\text{rank}(A)} S(k,k)U_1(:,k)U_2(:,k)^T.$$

From there we prove various optimality theorems and conclude that

$$A_r = \sum_{k=1}^{r} S(k,k)U_1(:,k)U_2(:,k)^T \qquad r \le \text{rank}(A)$$

is the closest rank-r matrix to A in (say) the Frobenius norm.

On the algorithmic front methods typically determine U_1 and U_2 through a sequence of updates. Thus, if $A = U_1 S U_2^T$ is the "current" representation of A we proceed to compute orthogonal Δ_1 and Δ_2 so that $\tilde{S} = \Delta_1^T S \Delta_2$ is "more diagonal" than S. We then update the representation:

$$S \leftarrow {\Delta_1}^T S \Delta_2, \qquad U_1 \leftarrow U_1\Delta_1, \qquad U_2 \leftarrow U_2\Delta_2.$$

Our plan is to mimic this sequence of events for tensors focusing first on the connection between the core tensor $\mathcal{S}$ and the U matrices.

7.3 The Mode-k Product

Updating a Tucker representation involves updating the current core tensor $\mathcal{S}$ and the associated U-matrices. Regarding the former we anticipate the need to design a relevant tensor-matrix product. The *mode-k product* turns out to be that operation and we motivate the main idea with an example. Suppose $\mathcal{S} \in \mathbb{R}^{4\times3\times2}$ and consider its mode-2 unfolding:

$$\mathcal{S}_{(2)} = \begin{bmatrix} s_{111} & s_{211} & s_{311} & s_{411} & s_{112} & s_{212} & s_{312} & s_{412} \\ s_{121} & s_{221} & s_{321} & s_{421} & s_{122} & s_{222} & s_{322} & s_{422} \\ s_{131} & s_{231} & s_{331} & s_{431} & s_{132} & s_{232} & s_{332} & s_{432} \end{bmatrix}.$$

Its columns are the mode-2 fibers of $\mathcal{S}$. Suppose we apply a 5-by-3 matrix M to each of those fibers:

$$\begin{bmatrix} t_{111} & t_{211} & t_{311} & t_{411} & t_{112} & t_{212} & t_{312} & t_{412} \\ t_{121} & t_{221} & t_{321} & t_{421} & t_{122} & t_{222} & t_{322} & t_{422} \\ t_{131} & t_{231} & t_{331} & t_{431} & t_{132} & t_{232} & t_{332} & t_{432} \\ t_{141} & t_{241} & t_{341} & t_{441} & t_{142} & t_{242} & t_{342} & t_{442} \\ t_{151} & t_{251} & t_{351} & t_{451} & t_{152} & t_{252} & t_{352} & t_{452} \end{bmatrix}$$

$$= \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \\ m_{41} & m_{42} & m_{43} \\ m_{51} & m_{52} & m_{53} \end{bmatrix} \begin{bmatrix} s_{111} & s_{211} & s_{311} & s_{411} & s_{112} & s_{212} & s_{312} & s_{412} \\ s_{121} & s_{221} & s_{321} & s_{421} & s_{122} & s_{222} & s_{322} & s_{422} \\ s_{131} & s_{231} & s_{331} & s_{431} & s_{132} & s_{232} & s_{332} & s_{432} \end{bmatrix}.$$

This defines a new tensor $\mathcal{T} \in \mathbb{R}^{4\times 5\times 2}$ that is totally specified by the equation

$$\mathcal{T}_{(2)} \;=\; M \cdot \mathcal{S}_{(2)}$$

and referred to as the mode-2 product of a tensor $\mathcal{S}$ with a matrix M. In general, if $\mathcal{S}$ is an $n_1 \times \cdots \times n_d$ tensor and $M \in \mathbb{R}^{m_k \times n_k}$ for some k that satisfies $1 \le k \le d$, then the mode-k product of $\mathcal{S}$ with M is a new tensor $\mathcal{T}$ defined by

$$\mathcal{T}_{(k)} \;=\; M \cdot \mathcal{S}_{(k)}.$$

To indicate this operation we use the notation

$$\mathcal{T} \;=\; \mathcal{S} \times_k M.$$

Note that $\mathcal{T}$ is an $n_1 \times \cdots \times n_{k-1} \times m_k \times n_{k+1} \times n_d$ tensor. To illustrate more characterizations of the mode-k product we drop down to the order-3 case.

– *Mode-1 Product.* If $\mathcal{S} \in \mathbb{R}^{n_1\times n_2\times n_3}$ and $M_1 \in \mathbb{R}^{m_1\times n_1}$, then $\mathcal{T} = \mathcal{S}\times_1 M_1$ is an $m_1 \times n_2 \times n_3$ tensor that is equivalently defined by

$$\mathcal{T}(i_1, i_2, i_3) = \sum_{k=1}^{n_1} M_1(i_1, k)\mathcal{S}(k, i_2, i_3)$$

$$\mathrm{vec}(\mathcal{T}) = (I_{n_3} \otimes I_{n_2} \otimes M_1)\mathrm{vec}(\mathcal{S}). \tag{12}$$

– *Mode-2 Product.* If $\mathcal{S} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and $M_2 \in \mathbb{R}^{m_2 \times n_2}$, then $\mathcal{T} = \mathcal{S} \times_2 M_2$ is an $n_1 \times m_2 \times n_3$ tensor that is equivalently defined by

$$\mathcal{T}(i_1, i_2, i_3) = \sum_{k=1}^{n_2} M_2(i_2, k)\mathcal{S}(i_1, k, i_3)$$

$$\text{vec}(\mathcal{T}) = (I_{n_3} \otimes M_2 \otimes I_{n_1})\text{vec}(\mathcal{S}) \tag{13}$$

– *Mode-3 Product.* If $\mathcal{S} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and $M_3 \in \mathbb{R}^{m_3 \times n_3}$, then $\mathcal{T} = \mathcal{A} \times_3 M_3$ is an $n_1 \times n_2 \times m_3$ tensor that is is equivalently defined by

$$\mathcal{T}(i_1, i_2, i_3) = \sum_{k=1}^{n_3} M_3(i_3, k)\mathcal{S}(i_1, i_2, k)$$

$$\text{vec}(\mathcal{T}) = (M_3 \otimes I_{n_2} \otimes I_{n_1})\text{vec}(\mathcal{S}) \tag{14}$$

The modal products have two important properties that we will be using later. The first concerns successive products in the *same* mode. If $\mathcal{S} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ and $M_1, M_2 \in \mathbb{R}^{n_k \times n_k}$, then

$$(\mathcal{S} \times_k M_1) \times_k M_2 \;=\; \mathcal{S} \times_k (M_1 M_2).$$

The second property concerns successive products in *different* modes. If $\mathcal{S} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, $M_k \in \mathbb{R}^{n_k \times n_k}$, $M_j \in \mathbb{R}^{n_j \times n_j}$, and $k \neq j$, then

$$(\mathcal{S} \times_k M_k) \times_j M_j \;=\; (\mathcal{S} \times_j M_j) \times_k M_k$$

The order is not important so we just write $\mathcal{S} \times_j M_j \times_k M_k$ or $\mathcal{S} \times_k M_k \times_j M_j$.

7.4 The Core Tensor

Suppose we have a Tucker representation

$$\mathcal{A}(\mathbf{i}) \;=\; \sum_{\mathbf{j}=\mathbf{1}}^{\mathbf{n}} \mathcal{S}(\mathbf{j}) \cdot U_1(i_1, j_1) \cdot U_2(i_2, j_2) \cdot U_3(i_3, j_3),$$

where $\mathcal{A} \in \mathbb{R}^{\mathbf{n}}$, $\mathbf{n} = [n_1,\, n_2,\, n_3]$, and $U_1 \in \mathbb{R}^{n_1 \times n_1}$, $U_2 \in \mathbb{R}^{n_2 \times n_2}$, and $U_3 \in \mathbb{R}^{n_3 \times n_3}$. It follows that

$$\begin{aligned}\text{vec}(\mathcal{A}) &= (U_3 \otimes U_2 \otimes U_1)\text{vec}(\mathcal{S}) \\ &= (U_3 \otimes I_{n_2} \otimes I_{n_1})(I_{n_3} \otimes U_2 \otimes I_{n_1})(I_{n_3} \otimes I_{n_2} \otimes U_1)\text{vec}(\mathcal{S}).\end{aligned}$$

This factored product enables us to relate the core tensor $\mathcal{S}$ to $\mathcal{A}$ via a triplet of modal products. Indeed, if

$$\text{vec}(\mathcal{S}^{(1)}) = (I_{n_3} \otimes I_{n_2} \otimes U_1)\text{vec}(\mathcal{S})$$

$$\text{vec}(\mathcal{S}^{(2)}) = (I_{n_3} \otimes U_2 \otimes I_{n_1})\text{vec}(\mathcal{S}^{(1)})$$

$$\text{vec}(\mathcal{S}^{(3)}) = (U_3 \otimes I_{n_2} \otimes I_{n_1})\text{vec}(\mathcal{S}^{(2)})$$

then $\mathcal{A} = \mathcal{S}^{(3)}$. But from (12), (13), and (14) this means that

$$\mathcal{S}^{(1)} = \mathcal{S}\times_1 U_1 \qquad \mathcal{S}^{(2)} = \mathcal{S}^{(1)}\times_2 U_2 \qquad \mathcal{S}^{(3)} = \mathcal{S}^{(2)}\times_3 U_3$$

and so

$$\mathcal{A} = \mathcal{S}\times_1 U_1\times_2 U_2\times_3 U_3.$$

If the U's are nonsingular then

$$\begin{aligned}\mathcal{A} &= \mathcal{A}\times_1 (U_1^{-1}U_1)\times_2 (U_2^{-1}U_2)\times_3 (U_3^{-1}U_3)\\ &= \left(\mathcal{A}\times_1 U_1^{-1}\times_2 U_2^{-1}\times_3 U_3^{-1}\right)\times_1 U_1\times_2 U_2\times_3 U_3\end{aligned}$$

and so $\mathcal{S} = \mathcal{A}\times_1 U_1^{-1}\times_2 U_2^{-1}\times_3 U_3^{-1}$.

If the U's are orthogonal and $\mathcal{A} \in \mathbb{R}^{n_1\times n_2\times n_3}$ and $U_1 \in \mathbb{R}^{n_1\times n_1}$, $U_2 \in \mathbb{R}^{n_2\times n_2}$, and $U_3 \in \mathbb{R}^{n_3\times n_3}$ are orthogonal, then

$$\mathcal{A} = \mathcal{S}\times_1 U_1\times_2 U_2\times_3 U_3$$

where

$$\mathcal{S} = \mathcal{A}\times_1 U_1^T\times_2 U_2^T\times_3 U_3^T. \tag{15}$$

or equivalently

$$\mathcal{A}_{(1)} = U_1\mathcal{S}_{(1)}(U_3 \otimes U_2)^T \tag{16}$$

$$\mathcal{A}_{(2)} = U_2\mathcal{S}_{(2)}(U_3 \otimes U_1)^T \tag{17}$$

$$\mathcal{A}_{(3)} = U_3\mathcal{S}_{(3)}(U_2 \otimes U_1)^T \tag{18}$$

With this choice we are representing $\mathcal{A}$ as a Tucker product of a core tensor $\mathcal{S}$ and three orthogonal matrices. Things are beginning to look "SVD-like".

7.5 The Higher-Order SVD

How do we choose the U matrices in (15) so that the core tensor $\mathcal{S}$ reveals things about the structure of $\mathcal{A}$? It is instructive to look at the matrix case where we know the answer to this question. Suppose we have the SVDs

$$U_1^T \mathcal{A}_{(1)} V_1 = \Sigma_1, \qquad U_2^T \mathcal{A}_{(2)} V_2 = \Sigma_2.$$

Since $\mathcal{A}_1 = A$ and $\mathcal{A}_{(2)} = A^T$ it follows that we may set $V_1 = U_2$. In other words, we can (in principle) compute the SVD of A by computing the SVD of the modal unfoldings $\mathcal{A}_{(1)}$ and $\mathcal{A}_{(2)}$. The U matrices from the modal SVDs are the "right" choice.

This suggests a strategy for picking good U matrices for the tensor case $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$. Namely, compute the SVD of the modal unfoldings

$$\mathcal{A}_{(1)} = U_1 \Sigma_1 V_1^T \qquad \mathcal{A}_{(2)} = U_2 \Sigma_2 V_2^T \qquad \mathcal{A}_{(3)} = U_3 \Sigma_3 V_3^T \tag{19}$$

and set

$$\mathcal{S} = \mathcal{A} \times_1 U_1^T \times_2 U_2^T \times_3 U_3^T.$$

The resulting decomposition

$$\mathcal{A} = \mathcal{S} \times_1 U_1 \times_2 U_2 \times_3 U_3,$$

is the *higher-order SVD* (HOSVD) of $\mathcal{A}$. If $\mathcal{A} = \mathcal{S} \times_1 U_1 \times_2 U_2 \times_3 U_3$ is the HOSVD of $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, then

$$\mathcal{A} = \sum_{\mathbf{j}=\mathbf{1}}^{\mathbf{r}} \mathcal{S}(\mathbf{j}) \cdot U_1(:, j_1) \circ U_2(:, j_2) \circ U_3(:, j_3) \tag{20}$$

where $r_1 = \text{rank}(A_{(1)})$, $r_2 = \text{rank}(A_{(2)})$, and $r_3 = \text{rank}(A_{(3)})$. The triplet of modal ranks $[r_1, r_2, r_3]$ is called the *multilinear rank* of $\mathcal{A}$.

The core tensor in the HOSVD has important properties. By combining (16)–(19) we have

$$\mathcal{S}_{(1)} = \Sigma_1 V_1 (U_3 \otimes U_2)$$

$$\mathcal{S}_{(2)} = \Sigma_2 V_2 (U_3 \otimes U_1)$$

$$\mathcal{S}_{(3)} = \Sigma_3 V_3 (U_2 \otimes U_1)$$

from which we conclude that

$$\| \mathcal{S}(j,:,:) \|_F = \sigma_j(\mathcal{A}_{(1)}) \qquad j = 1:n_1$$

$$\| \mathcal{S}(:,j,:) \|_F = \sigma_j(\mathcal{A}_{(2)}) \qquad j = 1:n_2$$

$$\| \mathcal{S}(:,:,j) \|_F = \sigma_j(\mathcal{A}_{(3)}) \qquad j = 1:n_3.$$

Here, σ_j(C) denotes the jth largest singular value of the matrix C. Notice that the norms of the tensor's slices are getting smaller as we "move away" from $\mathcal{A}(1,1,1)$.

This suggests that we can use the grading in $\mathcal{S}$ to truncate the HOSVD:

$$\mathcal{A} \approx \mathcal{A}_{\tilde{\mathbf{r}}} = \sum_{\mathbf{j}=\mathbf{1}}^{\tilde{\mathbf{r}}} \mathcal{S}(\mathbf{j}) \cdot U_1(:,j_1) \circ U_2(:,j_2) \circ U_3(:,j_3)$$

where $\tilde{\mathbf{r}} \leq \mathbf{r}$, i.e., $\tilde{r}_1 \leq r_1$, $\tilde{r}_2 \leq r_2$, and $\tilde{r}_3 \leq r_3$. As with SVD-based low-rank approximation in the matrix case, we simply need a tolerance to determine how to abbreviate the summation in (20. For a deeper discussion of the HOSVD, see [8].

7.6 The Tucker Nearness Problem

Suppose we are given $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and $\mathbf{r} = [r_1, r_2, r_3] \leq [n_1, n_2, n_3] = \mathbf{n}$. In the *Tucker nearness problem* we determine $\mathcal{S} \in \mathbb{R}^{r_1 \times r_2 \times r_3}$ and matrices $U_1 \in \mathbb{R}^{n_1 \times r_1}$, $U_2 \in \mathbb{R}^{n_2 \times r_2}$, and $U_3 \in \mathbb{R}^{n_3 \times r_3}$ with orthonormal columns such that

$$\phi(U_1, U_2, U_3) = \left\| \mathcal{A} - \sum_{\mathbf{j}=\mathbf{1}}^{\mathbf{r}} \mathcal{S}(\mathbf{j}) \cdot U_1(:,j_1) \circ U_2(:,j_2) \circ U_3(:,j_3) \right\|_F$$

is minimized. It is easy to show that

$$\phi(U_1, U_2, U_3) = \| \operatorname{vec}(\mathcal{A}) - (U_3 \otimes U_2 \otimes U_1)\operatorname{vec}(\mathcal{S}) \|_2$$

and so from using normal equations we see that

$$\mathcal{S} = \left(U_3^T \otimes U_2^T \otimes U_1^T\right) \cdot \operatorname{vec}(\mathcal{A}).$$

This is why the objective function ϕ does not have $\mathcal{S}$ as an argument: the "best $\mathcal{S}$" is determined by U_1, U_2, and U_3. The goal is to minimize

$$\phi(U_1, U_2, U_3) = \| \left(I - (U_3 \otimes U_2 \otimes U_1)\left(U_3^T \otimes U_2^T \otimes U_1^T\right)\right) \mathrm{vec}(\mathcal{A}) \|_2.$$

Since $U_3 \otimes U_2 \otimes U_1$ has orthonormal columns, it follows that minimizing this norm is the same as maximizing

$$\phi(U_1, U_2, U_3) = \| \left(U_3^T \otimes U_2^T \otimes U_1^T\right) \cdot \mathrm{vec}(\mathcal{A}) \|_2.$$

The reformulations

$$\phi(U_1, U_2, U_3) = \begin{cases} \| U_1^T \cdot A_{(1)} \cdot (U_3 \otimes U_2) \|_F \\ \| U_2^T \cdot A_{(2)} \cdot (U_3 \otimes U_1) \|_F \\ \| U_3^T \cdot A_{(3)} \cdot (U_2 \otimes U_1) \|_F \end{cases}$$

set the stage for a componentwise optimization approach:

Fix U_2 and U_3 and choose U_1 to maximize $\| U_1^T \cdot A_{(1)} \cdot (U_3 \otimes U_2) \|_F$.

Fix U_1 and U_3 and choose U_2 to maximize $\| U_2^T \cdot A_{(2)} \cdot (U_3 \otimes U_1) \|_F$.

Fix U_1 and U_2 and choose U_3 to maximize $\| U_3^T \cdot A_{(3)} \cdot (U_2 \otimes U_1) \|_F$.

These optimization problems can be solved using the SVD. Consider the problem of maximizing $\| Q^T M \|_F$ where $Q \in \mathbb{R}^{m \times r}$ has orthonormal columns and $M \in \mathbb{R}^{m \times n}$ is given. If

$$M = U \Sigma V^T$$

is the SVD of M, then

$$\| Q^T M \|_F^2 = \| Q^T U \Sigma V^T \|_F^2 = \| Q^T U \Sigma \|_F^2 = \sum_{k=1}^{r} \sigma_k^2 \| Q^T U(:, k) \|_2^2.$$

It is clear that we can maximize the summation by setting $Q = U(:, 1 : r)$. Putting it all together we obtain the following framework.

The Tucker Nearness Problem

Given $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$

Compute the HOSVD of $\mathcal{A}$ and determine $\mathbf{r} = [r_1, r_2, r_3] \leq \mathbf{n}$.

Set $U_1 = U_1(:, 1:r_1)$, $U_2 = U_2(:, 1:r_2)$, and $U_3 = U_3(:, 1:r_3)$.

Repeat:

Compute the SVD $\mathcal{A}_{(1)} \cdot (U_3 \otimes U_2) = \tilde{U}_1 \Sigma_1 V_1^T$

and set $U_1 = \tilde{U}_1(:, 1:\tilde{r}_1)$.

Compute the SVD $\mathcal{A}_{(2)} \cdot (U_3 \otimes U_1) = \tilde{U}_2 \Sigma_2 V_2^T$

and set $U_2 = \tilde{U}_2(:, 1:\tilde{r}_2)$.

Compute the SVD $\mathcal{A}_{(3)} \cdot (U_2 \otimes U_1) = \tilde{U}_3 \Sigma_3 V_3^T$

and set $U_3 = \tilde{U}_3(:, 1:\tilde{r}_3)$.

$U_1^{(opt)} = U_1$, $U_2^{(opt)} = U_2$, $U_3^{(opt)} = U_3$

Using the HOSVD to generate an initial guess makes sense given the discussion in Sect. 7.5. The matrix-matrix products, e.g., $\mathcal{A}_{(1)} \cdot (U_3 \otimes U_2)$, are rich in exploitable Kronecker structure. See [27] for further details.

7.7 A Jacobi Approach

Solving the Tucker Nearness problem is not equivalent to maximizing the "diagonal mass" of the core tensor $\mathcal{S}$. We briefly describe a Jacobi-like procedure that does. To motivate the main idea, consider the problem of maximizing $\text{tr}(U_1^T A U_2)$ where $A \in \mathbb{R}^{n_1 \times n_2}$ is given, $U_1 \in \mathbb{R}^{n_1 \times n_1}$ is orthogonal, and $U_2 \in \mathbb{R}^{n_2 \times n_2}$ is orthogonal. It is easy to show that the optimum U_1 and U_2 have the property that $U_1^T A U_2$ is diagonal with nonnegative diagonal entries. Thus, the SVD solves this particular "max trace" problem.

Now suppose $\mathcal{C}$ is n-by-n-by-n and define

$$\psi(\mathcal{C}) = \sum_{i=1}^{n} c_{iii}.$$

Given $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, our goal is to compute orthogonal $U_1 \in \mathbb{R}^{n_1 \times n_1}$, $U_2 \in \mathbb{R}^{n_2 \times n_2}$, and $U_3 \in \mathbb{R}^{n_3 \times n_3}$ so that if the tensor $\mathcal{S}$ is defined by

$$\text{vec}(\mathcal{S}) \;=\; (U_3 \;\otimes\; U_2 \;\otimes\; U_1)^T \text{vec}(\mathcal{A})$$

then $\phi(\mathcal{S})$ is maximized. Here is a Jacobi-like strategy for updating the "current" orthogonal triplet $\{\, U_1,\; U_2,\; U_3 \,\}$ so that new core tensor has a larger trace.

A Jacobi Framework for Computing a Compressed Tucker Representation

Given: $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$

Set $U_1 = I_n$, $U_2 = I_n$, $U_3 = I_n$, and $\mathcal{S} = \mathcal{A}$.

Repeat:

Find "simple" orthogonal $\tilde{U}_1$, $\tilde{U}_2$, and $\tilde{U}_3$ so that

$$\text{tr}(\mathcal{S} \times_1 \tilde{U}_1 \times_2 \tilde{U}_2 \times_3 \tilde{U}_3) > \text{tr}(\mathcal{S})$$

Update:

$$\mathcal{S} = \mathcal{S} \times_1 \tilde{U}_1 \times_2 \tilde{U}_2 \times_3 \tilde{U}_3$$

$$U_1 = U_1 \tilde{U}_1,\; U_2 = U_2 \tilde{U}_2,\; U_3 = U_3 \tilde{U}_3$$

$U_1^{(opt)} = U_1$, $U_2^{(opt)} = U_2$, $U_3^{(opt)} = U_3$.

We say that this iteration strives for a "compressed" Tucker representation because there is an explicit attempt to compress the information in $\mathcal{A}$ into a relatively small number of near-the-diagonal entries in $\mathcal{S}$. One idea for $\tilde{U}_3 \;\otimes\; \tilde{U}_2 \;\otimes\; \tilde{U}_1$ is to use carefully designed Jacobi rotations, e.g.,

$$\tilde{U}_3 \;\otimes\; \tilde{U}_2 \;\otimes\; \tilde{U}_1 \;=\; \begin{cases} \quad I_n \;\otimes J_{pq}(\beta) \;\otimes J_{pq}(\alpha) \\ J_{pq}(\beta) \;\otimes \quad I_n \quad \otimes J_{pq}(\alpha) \\ J_{pq}(\beta) \;\otimes J_{pq}(\alpha) \;\otimes I_n \end{cases}.$$

Here, $J_{pq}(\theta)$ is a Jacobi rotation in planes p and q. These updates modify only two diagonal entries: s_{ppp} and s_{qqq}. Sines and cosines can be chosen to increase the resulting trace and their determination leads to a 2-by-2-by-2 Jacobi rotation subproblem.

For example, determine $c_\alpha = \cos(\alpha)$, $s_\alpha = \sin(\alpha)$, $c_\beta = \cos(\beta)$, and $s_\beta = \sin(\beta)$, so that if

$$\begin{bmatrix} \sigma_{ppp} & \sigma_{pqp} \\ \sigma_{qpp} & \sigma_{qqp} \end{bmatrix} = \begin{bmatrix} c_\alpha & s_\alpha \\ -s_\alpha & c_\alpha \end{bmatrix}^T \begin{bmatrix} s_{ppp} & s_{pqp} \\ s_{qpp} & s_{qqp} \end{bmatrix} \begin{bmatrix} c_\beta & s_\beta \\ -s_\beta & c_\beta \end{bmatrix}$$

and

$$\begin{bmatrix} \sigma_{ppq} & \sigma_{pqq} \\ \sigma_{qpq} & \sigma_{qqq} \end{bmatrix} = \begin{bmatrix} c_\alpha & s_\alpha \\ -s_\alpha & c_\alpha \end{bmatrix}^T \begin{bmatrix} s_{ppq} & s_{pqq} \\ s_{qpq} & s_{qqq} \end{bmatrix} \begin{bmatrix} c_\beta & s_\beta \\ -s_\beta & c_\beta \end{bmatrix}$$

then $\sigma_{ppp} + \sigma_{qqq}$ is maximized. See [19].

8 The CP Decomposition

As with the Tucker representation, the CP representation of a tensor expresses the tensor as a sum-of-rank-one tensors. However, it does not involve orthogonality and the core tensor is truly diagonal, e.g., $s_{ijk} = 0$ unless $i = j = k$.

A note about terminology before we begin. The ideas behind the CP decomposition are very similar to the ideas behind the *C*ANDECOMP (Canonical Decomposition) and the *P*ARAFAC (Parallel Factors Decomposition). Thus, "CP" is an effective way to acknowledge the connections.

8.1 CP Representations: The Matrix Case

For matrices, the SVD

$$A = U_1 \Sigma U_2^T = \sum_i \sigma_i U_1(:,i) U_2(:,i)^T$$

is an example of a CP decomposition. But an eigenvalue decomposition also qualifies. If A is diagonalizable, then we have

$$A = U_1 \text{diag}(\lambda_i) U_2^T = \sum_i \lambda_i U_1(:,i) U_2(:,i)^T$$

where $U_2^T = U_1^{-1}$. Of course orthogonality is part of the SVD and biorthogonality ($U_2^T U_1 = I$) figures in eigenvalue diagonalization. This kind of structure falls by the wayside when we graduate to tensors.

8.2 CP Representation: The Tensor Case

We use the order-3 situation to expose the key ideas. The CP representation for an $n_1 \times n_2 \times n_3$ tensor $\mathcal{A}$ has the form

$$\mathcal{A} = \sum_{k=1}^{r} \lambda_k U_1(:,k) \circ U_2(:,k) \circ U_3(:,k)$$

where λ's are real scalars and $U_1 \in \mathbb{R}^{n_1 \times r}$, $U_2 \in \mathbb{R}^{n_2 \times r}$, and $U_3 \in \mathbb{R}^{n_3 \times r}$ have unit 2-norm columns. Alternatively, we have

$$\mathcal{A}(i_1,i_2,i_3) = \sum_{j=1}^{r} \lambda_j \cdot U_1(i_1,j) \cdot U_2(i_2,j) \cdot U_3(i_3,j)) \tag{21}$$

$$\text{vec}(\mathcal{A}) = \sum_{j=1}^{r} \lambda_j \cdot U_3(:,j) \otimes U_2(:,j) \otimes U_1(:,j) \tag{22}$$

In contrast to the Tucker representation,

$$\mathcal{A} = \sum_{j_1=1}^{r_1} \sum_{j_2=1}^{r_2} \sum_{j_3=1}^{r_3} \mathcal{S}(j_1,j_2,j_3) \cdot U_1(:,j_1) \circ U_2(:,j_2) \circ U_3(:,j_3),$$

we see that the CP representation involves a diagonal core tensor. The Tucker representation gives that up in exchange for orthonormal U matrices.

8.3 More About Tensor Rank

As we mentioned in Sect. 5.3, the rank of a tensor $\mathcal{A}$ is the minimum number of rank-1 tensors that sum to $\mathcal{A}$. *Thus, the length of the shortest possible CP representation of a tensor is its rank.* Our analysis of the 2-by-2-by-2 situation indicates that there are several fundamental differences between tensor rank and matrix rank. Here are some more anomalies:

Anomaly 1. The largest rank attainable for an n_1-by-$\ldots$-n_d tensor is called the maximum rank. It is *not* a simple formula that depends on the dimensions $n_1,\ldots,n_d$. Indeed, its precise value is only known for small examples. Maximum rank does not equal $\min\{n_1,\ldots,n_d\}$ unless $d \leq 2$.

Anomaly 2. If the set of rank-k tensors in $\mathbb{R}^{n_1 \times \cdots \times n_d}$ has positive Lebesgue measure, then k is a typical rank. Here are some examples where this quantity is known:

Size	Typical ranks
$2 \times 2 \times 2$	2,3
$3 \times 3 \times 3$	4
$3 \times 3 \times 4$	4,5
$3 \times 3 \times 5$	5,6

For n_1-by-n_2 matrices, typical rank and maximal rank are both equal to the smaller of n_1 and n_2.

Anomaly 3. The rank of a particular tensor over the real field may be different than its rank over the complex field.

Anomaly 4. It is possible for a tensor with a given rank to be arbitrarily close to a tensor with lesser rank. Such a tensor is said to be *degenerate*.

For more on the issue of tensor rank, see [9] and [13].

8.4 The Nearest CP Problem

Suppose $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and r are given. The *nearest CP approximation* problem involves finding a vector $\lambda \in \mathbb{R}^r$ and matrices $U_1 \in \mathbb{R}^{n_1 \times r}$, $U_2 \in \mathbb{R}^{n_2 \times r}$, and $U_3 \in \mathbb{R}^{n_3 \times r}$ (with unit 2-norm columns) so that

$$\phi(U_1, U_2, U_3, \lambda) = \left\| \mathcal{A} - \sum_{j=1}^{r} \lambda_j \cdot U_1(:,j) \circ U_2(:,j) \circ U_3(:,j) \right\|_F$$

is minimized. The objective function for this multilinear optimization problem has three different formulations:

$$\phi(U_1, U_2, U_3, \lambda) = \left\{ \begin{array}{l} \left\| \mathcal{A}_{(1)} - \sum_{j=1}^{r} \lambda_j \cdot U_1(:,j) \otimes (U_3(:,j) \otimes U_2(:,j))^T \right\|_F \\ \left\| \mathcal{A}_{(2)} - \sum_{j=1}^{r} \lambda_j \cdot U_2(:,j) \otimes (U_3(:,j) \otimes U_1(:,j))^T \right\|_F \\ \left\| \mathcal{A}_{(3)} - \sum_{j=1}^{r} \lambda_j \cdot U_3(:,j) \otimes (U_2(:,j) \otimes U_1(:,j))^T \right\|_F \end{array} \right. . \tag{23}$$

The summations in these expressions are highly structured matrix-matrix products. To facilitate the discussion we introduce a special variant of the Kronecker product.

8.5 The Khatri-Rao Product

If $B = \left[\, b_1 \middle| \cdots \middle| b_r \,\right] \in \mathbb{R}^{n_1 \times r}$ and $C = \left[\, c_1 \middle| \cdots \middle| c_r \,\right] \in \mathbb{R}^{n_2 \times r}$, then the *Khatri-Rao product* of B and C is given by

$$B \odot C \;=\; \left[\, b_1 \otimes c_1 \middle| \cdots \middle| b_r \otimes c_r \,\right] \in \mathbb{R}^{n_1 n_2 \times r}.$$

Thus, the kth column of $B \odot C$ is $B(:,k) \otimes C(:,k)$ The Khatri-Rao product is a submatrix of the Kronecker product. To see this, observe that

$$\left[\, b_1 \middle| b_2 \middle| b_3 \,\right] \otimes \left[\, c_1 \middle| c_2 \middle| c_3 \,\right]$$

$$= \left[\, b_1 \otimes c_1 \middle| b_1 \otimes c_2 \middle| b_1 \otimes c_3 \middle| b_2 \otimes c_1 \middle| b_2 \otimes c_2 \middle| b_2 \otimes c_3 \middle| b_3 \otimes c_1 \middle| b_3 \otimes c_2 \middle| b_3 \otimes c_3 \,\right].$$

In general, if $B \in \mathbb{R}^{n_1 \times r}$, $C \in \mathbb{R}^{n_2 \times r}$, and $A = B \odot C$, then $A = \tilde{A}(:, 1 : r+1 : r^2)$ where $\tilde{A} = B \otimes C$.

The Khatri-Rao least square problem

$$\min \| (B \odot C)x = d \|_2 \qquad d \in \mathbb{R}^{n_1 n_2}$$

can be solved very fast if we use the method of normal equations:

$$(B \odot C)^T (B \odot C)x \;=\; (B \odot C)^T d.$$

To see this, observe that the matrix of coefficients is a pointwise product of r-by-r matrices:

$$(B \odot C)^T (B \odot C) \;=\; (B^T B) .* (C^T C).$$

This is an $O((n_1 + n_2)r^2)$ operation. The structure of the right hand side can also be exploited. Indeed

$$(B \odot C)^T d \;=\; \begin{bmatrix} c_1^T D b_1 \\ \vdots \\ c_r^T D b_r \end{bmatrix}$$

where $D = \texttt{reshape}(z, [n_2, n_1])$. This can be computed with $O((n_1 + n_2)r^2)$ work. Overall it requires $O((n_1 + n_2)r^2)$ work to set up the r-by-r normal equation system and $O(r^3)$ flops to solve it. The naive method would involve $O((n_1 n_2)r^2)$ work.

8.6 Equivalent Formulations

We are now ready to formulate an alternating least squares framework for solving the nearest CP problem. Combining our discussion of the Khatri-Rao product with (23) we see that

$$\phi(U_1, U_2, U_3, \lambda) = \begin{cases} \| \mathcal{A}_{(1)}^T - (U_3 \odot U_2) \cdot (\mathrm{diag}(\lambda_j) \cdot U_1^T) \|_F \\ \| \mathcal{A}_{(2)}^T - (U_3 \odot U_1) \cdot (\mathrm{diag}(\lambda_j) \cdot U_2^T) \|_F \\ \| \mathcal{A}_{(3)}^T - (U_2 \odot U_1) \cdot (\mathrm{diag}(\lambda_j) \cdot U_3^T) \|_F \end{cases}.$$

Repeatedly minimizing these expressions with respect to U_1, U_2, and U_3 gives rise to the following framework for solving the nearest CP problem:

The Nearest CP Problem

Given: $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and a positive integer r

Set $U_1 = I_{n_1}(:, 1:r)$, $U_2 = I_{n_2}(:, 1:r)$, $U_3 = I_{n_3}(:, 1:r)$

Repeat:

Let X minimize $\| \mathcal{A}_{(1)}^T - (U_3 \odot U_2)X \|_F$.

for $j = 1:r$

$\lambda_j = \| X(j,:) \|_2$, $U_1(:,j) = X(j,:)^T/\lambda_j$

Let Y minimize $\| \mathcal{A}_{(2)}^T - (U_3 \odot U_1)Y \|_F$

for $j = 1:r$

$\lambda_j = \| Y(j,:) \|_2$, $U_2(:,j) = Y(j,:)^T/\lambda_j$.

Let Z minimize $\| \mathcal{A}_{(3)}^T - (U_2 \odot U_1)Z \|_F$

for $j = 1:r$

$\lambda_j = \| Z(j,:) \|_2$, $U_3(:,j) = Z(j,:)^T/\lambda_j$.

$U_1^{(opt)} = U_1$, $U_2^{(opt)} = U_2$, $U_3^{(opt)} = U_3$, and $\lambda^{(opt)} = \lambda$.

Notice that the least squares problems for X, Y, and Z are each multiple right hand side Khatri-Rao least squares problems. See [27] for more details.

9 The Kronecker Product SVD

Suppose A is a block matrix with uniformly sized blocks. The Kronecker product SVD expresses A as an "optimal" sum of Kronecker products [23, 30]. Recalling that a block matrix A with uniformly sized blocks is a reshaped order-4 tensor, the KSVD can essentially be used to produce an exact representation for order-4 tensors that is a sum of tensor products between matrices.

9.1 The Nearest Kronecker Product Problem

Suppose $A = (A_{ij})$ is an m_1-by-n_1 block matrix whose blocks are m_2-by-n_2, i.e.,

$$A = \begin{bmatrix} A_{11} & \cdots & A_{1,n_1} \\ \vdots & \ddots & \vdots \\ A_{m_1,1} & \cdots & A_{m_1,n_1} \end{bmatrix}, \qquad A_{ij} \in \mathbb{R}^{m_2 \times n_2}. \tag{24}$$

The nearest Kronecker product problem with respect to this blocking involves finding $B \in \mathbb{R}^{m_1 \times n_1}$ and $C \in \mathbb{R}^{m_2 \times n_2}$ such that

$$\phi_A(B, C) \;=\; \| A - B \otimes C \|_F$$

is minimized. This problem can be reshaped into an equivalent nearest-rank-1 problem. Here is an example:

$$\phi_A(B,C) = \left\| \left[\begin{array}{cc|cc} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ \hline a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \\ \hline a_{51} & a_{52} & a_{53} & a_{54} \\ a_{61} & a_{62} & a_{63} & a_{64} \end{array}\right] - \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{bmatrix} \otimes \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} \right\|_F$$

$$= \left\| \left[\begin{array}{cccc} a_{11} & a_{21} & a_{12} & a_{22} \\ \hline a_{31} & a_{41} & a_{32} & a_{42} \\ \hline a_{51} & a_{61} & a_{52} & a_{62} \\ \hline a_{13} & a_{23} & a_{14} & a_{24} \\ \hline a_{33} & a_{43} & a_{34} & a_{44} \\ \hline a_{53} & a_{63} & a_{54} & a_{64} \end{array}\right] - \begin{bmatrix} b_{11} \\ b_{21} \\ b_{31} \\ b_{12} \\ b_{22} \\ b_{32} \end{bmatrix} \begin{bmatrix} c_{11} & c_{21} & c_{12} & c_{22} \end{bmatrix} \right\|_F$$

$$= \left\| \tilde{A} \;-\; \mathrm{vec}(B) \cdot \mathrm{vec}(C)^T \right\|_F .$$

This is a believable result since in both formulations every a_{ij} is uniquely differenced with a product of some B-entry and some C-entry.

There is a method behind the set-up of $\tilde{A}$. The rows of $\tilde{A}$ are vec's of the blocks stacked in the "vec order", e.g.,

$$\tilde{A} = \left[\begin{array}{cccc} a_{11} & a_{21} & a_{12} & a_{22} \\ \hline a_{31} & a_{41} & a_{32} & a_{42} \\ \hline a_{51} & a_{61} & a_{52} & a_{62} \\ \hline a_{13} & a_{23} & a_{14} & a_{24} \\ \hline a_{33} & a_{43} & a_{34} & a_{44} \\ \hline a_{53} & a_{63} & a_{54} & a_{64} \end{array}\right] = \left[\begin{array}{c} \mathrm{vec}(A_{11})^T \\ \mathrm{vec}(A_{21})^T \\ \mathrm{vec}(A_{31})^T \\ \mathrm{vec}(A_{12})^T \\ \mathrm{vec}(A_{22})^T \\ \mathrm{vec}(A_{32})^T \end{array}\right].$$

Since the closest rank-1 matrix to $\tilde{A} \in \mathbb{R}^{m_1 n_1 \times m_2 n_2}$ is given by its largest singular value and vectors, we obtain the following solution framework:

The $\min \| A - B \otimes C \|_F$ *Problem*

Given: $A \in \mathbb{R}^{m_1 m_2 \times n_1 n_2}$.

Compute the SVD $\tilde{A} = U \Sigma V^T = \sum_{k=1}^{r_{KP}} \sigma_k u_k v_k^T$.

Define $B_{opt} \in \mathbb{R}^{m_1 \times n_1}$ by $\mathrm{vec}(B) = \sqrt{\sigma_1}\, u_1$

Define $C_{opt} \in \mathbb{R}^{m_2 \times n_2}$ by $\mathrm{vec}(C) = \sqrt{\sigma_1}\, v_1$

There is no need to actually compute the full SVD of $\tilde{A}$ since we only require σ_1, u_1, and v_1. The Lanczos SVD process can be applied to compute these quantities [10, p. 571]. This is a particularly attractive strategy if A (and hence $\tilde{A}$) is large and sparse.

There are important special cases where the Kronecker factor matrices B and C inherit properties of A. For example, if A is symmetric and positive definite, then the same can be said of both B and C. If A is block banded with uniformly banded blocks, then B and C are banded. If A has positive entries, then B and C have positive entries, etc.

We mention that the same "tilde-matrix technology" can be applied to the minimization of

$$\phi(X) = \| A - X \otimes X \|_F$$

and

$$\phi(X) = \| A - (X \otimes Y + Y \otimes X) \|_F$$

provided $\tilde{A}$ is square. The Schur decomposition is involved in the corresponding "tilde" optimization problem.

9.2 The Kronecker Product SVD (KPSVD)

We can obtain a complete Kronecker product representation of A is we use the complete SVD of $\tilde{A} \in \mathbb{R}^{m_1 n_1 \times m_2 n_2}$:

$$\tilde{A} \;=\; U \Sigma V^T \;=\; \sum_{k=1}^{r_{KP}} \sigma_k u_k v_k^T.$$

If we define the matrices B_k and C_k by $\mathrm{vec}(B_k) = u_k$ and $\mathrm{vec}(C_k) = v_k$, then

$$A \;=\; \sum_{k=1}^{r_{KP}} \sigma_k B_k \;\otimes\; C_k.$$

We refer to r_{KP} as the *Kronecker rank* of A with respect to the chosen blocking (24). If $r \leq r_{KP}$, then in the Frobenius norm the matrix

$$A_r \;=\; \sum_{k=1}^{r} \sigma_k B_k \;\otimes\; C_k$$

is the nearest matrix to A that has Kronecker rank r.

9.3 Order-4 Tensor Approximation Using the KPSVD

If we unfold $\mathcal{A} \in \mathbb{R}^{n \times n \times n \times n}$ into an n^2-by-n^2 matrix A and compute its KPSVD, then we obtain an expansion of $\mathcal{A}$ that is a sum of matrix-matrix tensor products. For example, if

$$\mathcal{A}_{[1,3]\times[2,4]} \;=\; \sum_{k=1}^{r_{KP}} \sigma_k \, B_k \;\otimes\; C_k \qquad B_k, C_k \in \mathbb{R}^{n \times n}$$

then

$$\mathcal{A} \;=\; \sum_{k=1}^{r_{KP}} \sigma_k \, \mathcal{C}_k \circ \mathcal{B}_k,$$

i.e.,

$$\mathcal{A}(i_1, i_2, j_1, j_2) \;=\; \sum_{k=1}^{r_{KP}} \sigma_k \, C_k(i_1, i_2) B_k(j_1, j_2).$$

The summations in the above can be abbreviated to obtain best approximations using the optimality features of the KPSVD.

Is it possible to extend this "order-4 technology" to higher order tensors? Preliminary thinking on this leads to various alternating least squares frameworks. For example, suppose $\mathcal{A} \in \mathbb{R}^{\mathbf{n}}$ where $\mathbf{n} = [n, n, n, n, n, n]$ and that we wish to minimize

$$\phi_A(B, C, D) \;=\; \| \, A - B \otimes C \otimes D \, \|_F$$

where $B, C, D \in \mathbb{R}^{n \times n}$ and

$$A \;=\; \mathcal{A}_{[1\ 3\ 5] \times [2\ 4\ 6]} \in \mathbb{R}^{n^3 \times n^3}.$$

If we regard $A = (A_{ij})$ as an n-by-n block matrix with n^2-by-n^2 blocks, then

$$\phi_A(B, C, D)^2 \;=\; \sum_{i=1}^{n} \sum_{j=1}^{n} \| \, A_{ij} - b_{ij}(C \otimes D) \, \|_F^2.$$

If we fix C and D then we can minimize ϕ_A by setting

$$b_{ij} \;=\; \frac{\mathrm{tr}((C \otimes D)^T A_{ij})}{\| \, C \, \|_F^2 \| \, D \, \|_F^2}.$$

Similar expressions can be given for the optimum C given that B and D are fixed and for the optimum D given that B and C are fixed. Thus, we could approach the minimization of ϕ_A with a framework that cycles through these componentwise optimizations.

10 The Tensor Train SVD

The idea behind the tensor train representation is to approximate a high-order tensor with a collection of low-order tensors that are linked together through simple, 'nearest neighbor" summations [20, 21]. It is a topic worth discussing because it addresses directly the "curse of dimensionality", see [2, 3, 11].

A tensor train is a special case of a *tensor network*. In a general tensor network the nodes are low-order tensors and each edge represents a single-index summation between the two nodes that it connects The notation associated with a general tensor network is a major challenge but is quite tractable tensor trains.

10.1 Tensor Trains and Data Sparsity

Suppose we are given the following matrices and tensors:

$$\begin{aligned}
&\mathcal{G}_1 : n_1 \times r_1\\
&\mathcal{G}_2 : r_1 \times n_2 \times r_2\\
&\mathcal{G}_3 : r_2 \times n_3 \times r_3\\
&\mathcal{G}_4 : r_3 \times n_4 \times r_4\\
&\mathcal{G}_5 : r_4 \times n_5.
\end{aligned}$$

Define the integer vectors $\mathbf{n}$ and $\mathbf{r}$ by

$$\mathbf{n} \;=\; [n_1,\, n_2,\, n_3,\, n_4,\, n_5]$$

and

$$\mathbf{r} \;=\; [r_1,\, r_2,\, r_3,\, r_4].$$

The tensor $\mathcal{T} \in \mathbb{R}^{\mathbf{n}}$ defined by

$$\mathcal{T}(\mathbf{i}) \;=\; \sum_{\mathbf{k=1}}^{\mathbf{r}} \mathcal{G}_1(i_1,k_1)\cdot \mathcal{G}_2(k_1,i_2,k_2)\cdot \mathcal{G}_3(k_2,i_3,k_3)\cdot \mathcal{G}_4(k_3,i_4,k_4)\cdot \mathcal{G}_5(k_4,i_5)$$

is a *tensor train* with *carriages* $\mathcal{G}_1$, $\mathcal{G}_2$, $\mathcal{G}_3$, $\mathcal{G}_4$, and $\mathcal{G}_5$. Note that if $\bar{n} = \max\{n_1,n_2,n_3,n_4,n_5\}$ and

$$(r_1 + r_1r_2 + r_2r_3 + r_3r_4 + r_4)\bar{n} \;<<\; n_1n_2n_3n_4n_5$$

then $\mathcal{T}$ is data sparse. Under what circumstances can we approximate a given tensor $\mathcal{A}$ with a data sparse tensor train? We need a mechanism that exposes the redundancies in $\mathcal{A}$ and which determines the parameters $r_1,\ldots,r_4$ along the way. The procedure involves a sequence of matrix SVDs and careful unfoldings. The carriages turn out to be reshapings of SVD U-matrices.

10.2 Computing an SVD-Based Tensor Train Representation

We outline a framework that can be used to construct a data sparse tensor train approximation to a given tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times \cdots \times n_5}$. There are four steps:

Step 1. Set $M_1 = \texttt{reshape}(\mathcal{A}, [n_1, n_2 n_3 n_4 n_5])$ and compute the SVD

$$M_1 \;=\; U_1 \Sigma_1 V_1^T = U_1 Z_1$$

where $U_1 \in \mathbb{R}^{n_1 \times r_1}$, $Z_1 = \Sigma_1 V_1^T \in \mathbb{R}^{r_1 \times n_2 n_3 n_4 n_5}$ and $r_1 = \text{rank}(M_1)$. Define

$$\mathcal{G}_1 = U_1.$$

Step 2. Set $M_2 = \texttt{reshape}(Z_1, [r_1 n_2, n_3 n_4 n_5])$ and compute the SVD

$$M_2 \;=\; U_2 \Sigma_2 V_2^T = U_2 Z_2$$

where $U_2 \in \mathbb{R}^{r_1 n_2 \times r_2}$, $Z_2 = \Sigma_2 V_2^T \in \mathbb{R}^{r_2 \times n_3 n_4 n_5}$ and $r_2 = \text{rank}(M_2)$. Define

$$\mathcal{G}_2 \;=\; \texttt{reshape}(U_2, [r_1, n_2, r_2])$$

Step 3. Set $M_3 = \texttt{reshape}(Z_2, [r_2 n_3, n_4 n_5])$ and compute the SVD

$$M_3 \;=\; U_3 \Sigma_3 V_3^T = U_3 Z_3$$

where $U_3 \in \mathbb{R}^{r_2 n_3 \times r_3}$, $Z_3 = \Sigma_3 V_3^T \in \mathbb{R}^{r_3 \times n_4 n_5}$ and $r_3 = \text{rank}(M_3)$. Define

$$\mathcal{G}_3 \;=\; \texttt{reshape}(U_3, [r_2, n_3, r_3])$$

Step 4. Set $M_4 = \texttt{reshape}(Z_3, [r_3 n_4, n_5])$ and compute the SVD

$$M_4 \;=\; U_4 \Sigma_4 V_4^T = U_4 Z_4$$

where $U_4 \in \mathbb{R}^{r_3 n_4 \times r_4}$, $Z_4 \in \mathbb{R}^{r_4 \times n_5}$ and $r_4 = \text{rank}(M_4)$. Define

$$\mathcal{G}_4 \;=\; \texttt{reshape}(U_4, [r_3, n_4, r_4]) \qquad \mathcal{G}_5 \;=\; Z_4$$

Verification that the $\mathcal{G}$'s form a tensor train for $\mathcal{A}$ is somewhat involved and we refer the reader to [10, p.742]. However, to acquire some insight, let us assume that $n_1 = \cdots = n_5 = n$ and tabulate the sizes of the various matrices that arise in the tensor train computation:

i	$\texttt{Size}(M_i)$	$r_i = \text{rank}(M_i)$	$\texttt{Size}(U_i)$	$\texttt{Size}(Z_i)$	$\texttt{Size}(\mathcal{G}_i)$
1	n-by-n^4	$r_1 \leq n$	n-by-r_1	r_1-by-n^4	n-by-r_1
2	$r_1 n$-by-n^3	$r_2 \leq r_1 n \leq n^2$	n^2-by-r_2	r_2-by-n^3	r_1-by-n-by-r_2
3	$r_2 n$-by-n^2	$r_3 \leq \min\{r_2 n, n^2\}$	n^3-by-r_3	r_3-by-n^2	r_2-by-n-by-r_3
4	$r_3 n$-by-n	$r_4 \leq n$	n^4-by-r_4	r_4-by-n	r_3-by-n-by-r_4
5	—	—	—	—	r_4-by-n

Notice that the "rate" at which the M_i get thinner and thinner depends upon the rank deficiencies that the SVDs discover along the way. This is important since the amount of work in step i depends upon the dimensions of M_i. The amount of data in the M_i depend upon the r_i:

$$\text{If } N = n^5 \text{ then the amount of data in } \begin{Bmatrix} M_1 \\ M_2 \\ M_3 \\ M_4 \end{Bmatrix} \text{ is } \begin{Bmatrix} N \\ (r_1/n)N \\ (r_2/n^2)N \\ (r_3/n^3)N \end{Bmatrix}.$$

There are important applications for which the factors r_i/n^i are very small.

11 Tensor Problems with Multiple Symmetries

In dense matrix computations, the presence of symmetry ($A = A^T$) usually means that work and storage requirements are halved. Further economies can be realized if additional symmetries are around. For example, a centrosymmetric matrix is symmetric about both its diagonal *and* antidiagonal. It turns out that this can reduce work and storage requirements by a factor of four.

Matrix problems with multiple symmetries arise in tensor problems when the tensor in question has multiple symmetries. For example, if $\mathcal{A} \in \mathbb{R}^{n\times n\times n\times n}$ and

$$\mathcal{A}(i_1, i_2, i_3, i_4) = \mathcal{A}(i_2, i_1, i_3, i_4) = \mathcal{A}(i_1, i_2, i_4, i_3) = \mathcal{A}(i_3, i_4, i_1, i_2),$$

then certain unfoldings give rise to n^2-by-n^2 matrices that possess multiple symmetries. This creates interesting challenges. For example, can we efficiently compute structured low-rank approximations to $\mathcal{A}$ by computing structured low-rank approximations to its structured unfoldings? The answer is "yes".

11.1 A First Look at Multiple Symmetries

A matrix $A \in \mathbb{R}^{n\times n}$ is *centrosymmetric* if $A = A^T$ and $A = E_nAE_n$ where $E_n = I_n(:, n:-1:1)$. For example,

$$E_4 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \qquad A = \begin{bmatrix} a & b & c & d \\ b & e & f & c \\ c & f & e & b \\ d & c & b & a \end{bmatrix}.$$

Suppose $n = 2m$. It can be shown that

$$Q_E = \frac{1}{\sqrt{2}} \left[\begin{array}{c|c} I_m & I_m \\ E_m & -E_m \end{array}\right]$$

is orthogonal and

$$Q_E^T \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} Q_E = = \begin{bmatrix} A_{11} + A_{12}E_m & 0 \\ 0 & A_{11} - A_{12}E_m \end{bmatrix}.$$

This kind of "free" block diagonalization is at the heart of all structure-exploiting algorithms for centrosymmetric matrix problems. The original problem is basically replaced by a pair of half-sized problems, one for the (1,1) block $A_{11} + A_{12}E_m$ and the other for the (2,2) block $A_{11} - A_{12}E_m$. Thus, we could compute the Schur decomposition of A by computing two half-sized Schur decompositions. Since the complexity of such a calculation is cubic, work will be reduced by a factor of four.

11.2 A Tensor Problem with Multiple Symmetries

We now consider a quantum chemistry problem that gives rise to an order-4 tensor that has several different symmetries. Given a basis $\{\phi_i(\mathbf{r})\}_{i=1}^n$ of atomic orbital functions, we consider the following order-4 tensor:

$$\mathcal{A}(i_1, i_2, i_3, i_4) = \int_{\mathbf{R}^3}\int_{\mathbf{R}^3} \frac{\phi_{i_1}(\mathbf{r}_1)\phi_{i_2}(\mathbf{r}_1)\phi_{i_3}(\mathbf{r}_2)\phi_{i_4}(\mathbf{r}_2)}{\|\mathbf{r}_1 - \mathbf{r}_2\|} d\mathbf{r}_1 d\mathbf{r}_2. \tag{25}$$

This is called the *TEI tensor* and it plays an important role in electronic structure theory and ab initio quantum chemistry. By looking at the integrand it is easy to

show that

$$\mathcal{A}(i_1,i_2,i_3,i_4) \;=\; \begin{cases} \mathcal{A}(i_2.i_1.i_3,i_4) \\ \mathcal{A}(i_1,i_2,i_4,i_3) \\ \mathcal{A}(i_3,i_4.i_1,i_2) \end{cases}.$$

We say that $\mathcal{A}$ is ((12)(34))-*symmetric*.

A common calculation involves switching from the atomic orbital basis to a molecular orbital basis $\{\psi_i(\mathbf{r})\}_{i=1}^n$. If

$$\psi_i(\mathbf{r}) \;=\; \sum_{k=1}^{n} X(i,k)\phi_k(\mathbf{r}) \qquad i = 1,2,\ldots,n$$

then the molecular orbital basis tensor

$$\mathcal{B}(j_1,j_2,j_3,j_4) \;=\; \int_{\mathbf{R}^3}\int_{\mathbf{R}^3} \frac{\psi_{j_1}(\mathbf{r}_1)\psi_{j_2}(\mathbf{r}_1)\psi_{j_3}(\mathbf{r}_2)\psi_{j_4}(\mathbf{r}_2)}{\|\mathbf{r}_1-\mathbf{r}_2\|} d\mathbf{r}_1 d\mathbf{r}_2$$

is given by

$$\mathcal{B}(\mathbf{j}) \;=\; \sum_{i_1=1}^{n}\sum_{i_2=1}^{n}\sum_{i_3=1}^{n}\sum_{i_4=1}^{n} \mathcal{A}(\mathbf{i})\cdot X(i_1,j_1)\cdot X(i_2,j_2)\cdot X(i_3,j_3)\cdot X(i_4,j_4).$$

It can be shown that $\mathcal{B}$ is also ((12)(34))-symmetric.

The computation of $\mathcal{B}$ from $\mathcal{A}$ is neatly expressed in terms of the $[1\ 3]\times[2\ 4]$ unfolding:

$$\mathcal{B}_{[1,3]\times[2,4]} \;=\; (X\otimes X)^T \mathcal{A}_{[1,3]\times[2,4]}(X\otimes X).$$

This unfolding is based on the tensor-to-matrix mapping

$$\mathcal{A}(i_1,i_2,i_3,i_4) \;\rightarrow\; A(i_1+(i_3-1)n, i_2+(i_4-1)n)$$

and has a nice block-level interpretation. If we regard $A = \mathcal{A}_{[1,3]\times[2,4]}$ as an n-by-n block matrix (A_{rs}) with n-by-n blocks, then

$$\mathcal{A}(p,q,r,s) \;\leftrightarrow\; [\,A_{rs}\,]_{pq}\,.$$

It follows from the symmetries in tensor $\mathcal{A}$ that the blocks of matrix A are symmetric ($A_{rs}^T = A_{rs}$) and that A is block-symmetric ($A_{rs} = A_{sr}$). Less obvious is that the submatrices

$$\tilde{A}_{ij} = A(i:n:n^2, j:n:n^2)$$

are also symmetric. Here is an $n = 3$ example that showcases the three symmetries:

$$A = \left[\begin{array}{ccc|ccc|ccc} 11 & 12 & 13 & 12 & 17 & 18 & 13 & 18 & 22 \\ 12 & 14 & \mathbf{15} & 17 & 19 & \mathbf{20} & 18 & 23 & \mathbf{24} \\ 13 & 15 & 16 & 18 & 20 & 21 & 22 & 24 & 25 \\ \hline 12 & 17 & 18 & 14 & 19 & 23 & 15 & 20 & 24 \\ 17 & 19 & \mathbf{20} & 19 & 26 & \mathbf{27} & 20 & 27 & \mathbf{29} \\ 18 & 20 & 21 & 23 & 27 & 28 & 24 & 29 & 30 \\ \hline 13 & 18 & 22 & 15 & 20 & 24 & 16 & 21 & 25 \\ 18 & 23 & \mathbf{24} & 20 & 27 & \mathbf{29} & 21 & 28 & \mathbf{30} \\ 22 & 24 & 25 & 24 & 29 & 30 & 25 & 30 & 31 \end{array}\right].$$

Also of interest is the $[1,2]\times[3,4]$ unfolding $A = \mathcal{A}_{[1,2]\times[3,4]}$ defined by the mapping

$$\mathcal{A}(i_1, i_2, i_3, i_4) \;\rightarrow\; A(i_1 + (i_2 - 1)n, i_3 + (i_4 - 1)n).$$

Here is an example:

$$A = \left[\begin{array}{ccc|ccc|ccc} 11 & 12 & 13 & 12 & 14 & 15 & 13 & 15 & 16 \\ 12 & 17 & 18 & 17 & 19 & 20 & 18 & 20 & 21 \\ 13 & 18 & 22 & 18 & 23 & 24 & 22 & 24 & 25 \\ \hline 12 & 17 & 18 & 17 & 19 & 20 & 18 & 20 & 21 \\ 14 & 19 & 23 & 19 & 26 & 27 & 23 & 27 & 28 \\ 15 & 20 & 24 & 20 & 27 & 29 & 24 & 29 & 30 \\ \hline 13 & 18 & 22 & 18 & 23 & 24 & 22 & 24 & 25 \\ 15 & 20 & 24 & 20 & 27 & 29 & 24 & 29 & 30 \\ 16 & 21 & 25 & 21 & 28 & 30 & 25 & 30 & 31 \end{array}\right].$$

This unfolding of a $((1,2),(3,4))$ symmetric tensor is symmetric and has the property that each column reshapes to a symmetric matrix, e.g.,

$$\texttt{reshape}(A(:,1),[3\ 3]) \;=\; \left[\begin{array}{ccc} 11 & 12 & 13 \\ 12 & 14 & 15 \\ 13 & 15 & 16 \end{array}\right]$$

We call this *perfect shuffle symmetry*.

11.3 Perfect Shuffle Symmetry

An n^2 -by- n^2 matrix A is *PS-symmetric* if it is symmetric and satisfies

$$A = \Pi_{n,n} A \Pi_{n,n}$$

where $\Pi_{n,n}$ is the perfect shuffle permutation

$$\Pi_{n,n} = I_{n^2}(:, \mathbf{v}), \qquad \mathbf{v} = [\, 1 : n : n^2 \mid 2 : n : n^2 \mid \cdots \mid n : n : n^2 \,].$$

Here is an example:

$$\Pi_{3,3} = \left[\begin{array}{ccc|ccc|ccc} 1&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0\\ \hline 0&1&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&1&0\\ \hline 0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&1 \end{array}\right].$$

Because $\Pi_{n,n}$ is symmetric it has just two eigenvalues: $+1$ and -1. Consider the eigenvector equation

$$\Pi_{3,3} x = \left[\begin{array}{ccc|ccc|ccc} 1&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0\\ \hline 0&1&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0\\ 0&0&0&0&0&0&0&1&0\\ \hline 0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&1 \end{array}\right] \left[\begin{array}{c} x_{11}\\ x_{21}\\ x_{31}\\ \mathbf{x_{12}}\\ x_{22}\\ x_{32}\\ x_{13}\\ x_{23}\\ x_{33} \end{array}\right] = \pm \left[\begin{array}{c} x_{11}\\ x_{12}\\ x_{13}\\ \mathbf{x_{21}}\\ x_{22}\\ x_{23}\\ x_{31}\\ x_{32}\\ x_{33} \end{array}\right].$$

If $\Pi_{n,n} x = x$, then $\texttt{reshape}(x, [n, n])$ is symmetric. If $\Pi_{n,n} x = -x$, then $\texttt{reshape}(x, [n, n])$ is skew-symmetric.

11.4 Block Diagonalization

Suppose $A \in \mathbb{R}^{n^2 \times n^2}$ is PS-symmetric and λ is a distinct eigenvalue. Thus,

$$Ax = \lambda x \qquad \Rightarrow \qquad A(\Pi_{n,n} x) = \lambda (\Pi_{n,n} x)$$

from which we may conclude that either $\Pi_{n,n}x = x$ or $\Pi_{n,n}x = -x$. The first case says that x reshapes to an n-by-n symmetric matrix while the second case says that x reshapes to an n-by-n skew-symmetric matrix. From this we may conclude that the subspaces

$$S_{\text{sym}} = \{x \in \mathbb{R}^{n^2} \mid \texttt{reshape}(x, [n\ n]) \text{ is symmetric } \}$$

$$S_{\text{skew}} = \{x \in \mathbb{R}^{n^2} \mid \texttt{reshape}(x, [n\ n]) \text{ is skew-symmetric } \}$$

are invariant for A. Moreover, $S_{\text{sym}} = S_{\text{skew}}^{\perp}$. Here is an orthogonal matrix whose columns span these subspaces:

$$Q_{3,3} = \frac{1}{\sqrt{2}} \left[\begin{array}{cccccc|ccc} \sqrt{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & \sqrt{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & -1 \\ 0 & 0 & \sqrt{2} & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] = [\, Q_{\text{sym}} \mid Q_{\text{skew}} \,] \tag{26}$$

Here are a pair of column reshapings taken from this matrix:

$$Q_{3,3}(:,4) \equiv \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad Q_{3,3}(:,7) \equiv \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

It follows that if $A \in \mathbb{R}^{n^2 \times n^2}$ is PS-symmetric, then

$$Q_{n,n}^T A Q_{n,n} = \left[\begin{array}{cccccc|ccc} \times & \times & \times & \times & \times & \times & 0 & 0 & 0 \\ \times & \times & \times & \times & \times & \times & 0 & 0 & 0 \\ \times & \times & \times & \times & \times & \times & 0 & 0 & 0 \\ \times & \times & \times & \times & \times & \times & 0 & 0 & 0 \\ \times & \times & \times & \times & \times & \times & 0 & 0 & 0 \\ \times & \times & \times & \times & \times & \times & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & 0 & 0 & \times & \times & \times \end{array}\right] = \begin{bmatrix} A_{\text{sym}} & 0 \\ 0 & A_{\text{skew}} \end{bmatrix}. \tag{27}$$

This "free" block diagonalization can be effectively exploited as we now show.

11.5 Low-Rank PS-Symmetric Approximation

In certain important quantum chemistry applications, the n^2-by-n^2 matrix A in (27) is positive definite and very near a rank-n matrix. Our plan is to approximate the diagonal blocks A_{sym} and A_{skew} using Cholesky with diagonal pivoting.

Recall that pivoted LDL can be used to compute an approximate rank-r approximation to a positive definite matrix:

$$PAP^T \approx LDL^T \qquad \begin{cases} P \text{ is a permutation} \\ L \in \mathbb{R}^{n^2 \times r} \text{is unit lower triangular} \\ D = \text{diag}(d_i),\ d_1 \geq d_2 \geq \cdots \geq d_r > 0 \end{cases}$$

e.g.,

$$PAP^T \approx \begin{bmatrix} \times & 0 & 0 \\ \times & \times & 0 \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix} \begin{bmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times & \times & \times & \times \end{bmatrix}$$

See [10, p.167].

It follows from (26) and (27) that if we compute the low-rank LDL decompositions

$$A_{\text{sym}} \approx P_{\text{sym}}^T L_{\text{sym}} D_{\text{sym}} L_{\text{sym}}^T P_{\text{sym}}$$

$$A_{\text{skew}} \approx P_{\text{skew}}^T L_{\text{skew}} D_{\text{skew}} L_{\text{skew}}^T P_{\text{skew}}$$

then

$$A \approx V_{\text{sym}} D_{\text{sym}} V_{\text{sym}}^T + V_{\text{skew}} D_{\text{skew}} V_{\text{skew}}^T \tag{28}$$

where

$$V_{\text{sym}} = Q_{\text{sym}} P_{\text{sym}}^T L_{\text{sym}}$$

and

$$V_{\text{skew}} = Q_{\text{skew}} P_{\text{skew}}^T L_{\text{skew}}$$

It is easy to verify that these low-rank matrices are also PS-symmetric. When the structured approximation (28) to $A = \mathcal{A}_{[1,2]\times[3,4]}$ is substituted into

$$\mathcal{B}_{[1,2]\times[3,4]} \;=\; (X \otimes X)^T \mathcal{A}_{[1,2]\times[3,4]} (X \otimes X).$$

then the volume of work is greatly reduced. See [31].

References

1. G. Baumgartner, A. Auer, D. Bernholdt, A. Bibireata, V. Choppella, D. Cociorva, X. Gao, R. Harrison, S. Hirata, S. Krishnamoorthy, S. Krishnan, C. Lam, Q. Lu, M. Nooijen, R. Pitzer, J. Ramanujam, P. Sadayappan, A. Sibiryakov , Synthesis of high-performance parallel programs for a class of ab initio quantum chemistry models. Proc. IEEE **93**(2), 276–292 (2005)
2. G. Beylkin, M.J. Mohlenkamp, Numerical operator calculus in higher dimensions. Proc. Natl. Acad. Sci. **99**(16), 10246–10251 (2002)
3. G. Beylkin, M.J. Mohlenkamp, Algorithms for numerical analysis in high dimensions. SIAM J. Sci. Comp. **26**, 2133–2159 (2005)
4. P. Comon, G. Golub, L.-H. Lim, B. Mourrain, Genericity and rank deficiency of high order symmetric tensors. Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP '06) **31**(3), 125–128 (2006)
5. P. Comon, G. Golub, L.-H. Lim, B. Mourrain, Symmetric tensors and symmetric tensor rank. SIAM J. Matrix Anal. Appl. **30**, 1254–1279 (2008)
6. L. de Lathauwer, Signal Processing Based on Multilinear Algebra. Ph.D. thesis, K.U. Leuven, 1997
7. L. De Lathauwer, P. Comon, B. De Moor, J. Vandewalle, Higher-order power method–application in independent component analysis, in *Proceedings of the International Symposium on Nonlinear Theory and Its Applications (NOLTA '95)*, Las Vegas, NV (1995), pp. 91–96
8. L. De Lathauwer, B. De Moor, J. Vandewalle, A multilinear singular value decomposition. SIAM J. Matrix Anal. Appl. **21**, 1253–1278 (2000)
9. V. De Silva, L.-H. Lim, Tensor rank and the ill-posedness of the best low-rank approximation problem. SIAM J. Matrix Anal. Appl. **30**, 1084–1127 (2008)
10. G.H. Golub, C.F. Van Loan, *Matrix Computations*, 4th edn. (Johns Hopkins University Press, Baltimore, MD, 2013)
11. W. Hackbusch, B.N. Khoromskij, Tensor-product approximation to operators and functions in high dimensions. J. Complexity **23**, 697–714 (2007)
12. H.V. Henderson, S.R. Searle, The vec-permutation matrix, the vec operator and Kronecker products: a review. Linear Multilinear Algebra **9**, 271–288 (1981)
13. C.J. Hillar, L.-H. Lim, Most tensor problems are NP-hard. J. ACM **60**(6), Art. 33–47 (2013)
14. E. Kofidis, P.A. Regalia, On the best rank-1 approximation of higher-order supersymmetric tensors. SIAM J. Matrix Anal. Appl. **23**, 863–884 (2002)
15. T.G. Kolda, B.W. Bader, Algorithm 862: MATLAB tensor classes for fast algorithm prototyping. ACM Trans. Math. Softw. **32**, 635–653 (2006)
16. T.G. Kolda, B.W. Bader, Tensor decompositions and applications. SIAM Rev. **51**, 455–500 (2009)
17. L.-H. Lim, Singular values and eigenvalues of tensors: a variational approach, in *Proceedings of the IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP '05)*, vol. 1 (2005), pp. 129–132
18. L.-H. Lim, Tensors and hypermatrices, in *Handbook of Linear Algebra*, Chap. 15, 2nd edn., ed. by L. Hogben (CRC Press, Boca Raton, FL, 2013), 30 pp.

19. C. Martin, C. Van Loan, A Jacobi-type method for computing orthogonal tensor decompositions. SIAM J. Matrix Anal. Appl. **30**, 1219–1232 (2008)
20. I.V. Oseledets, E.E. Tyrtyshnikov, Breaking the curse of dimensionality, or how to use SVD in many dimensions. SIAM J. Sci. Comput. **31**, 3744–3759 (2008)
21. I. Oseledets, E. Tyrtyshnikov, TT-cross approximation for multidimensional arrays, Linear Algebra Appl. **432**, 70–88 (2010)
22. C.C. Paige, C.F. Van Loan, A Schur decomposition for Hamiltonian matrices. Linear Algebra Appl. **41**,11–32 (1981)
23. N. Pitsianis, C. Van Loan, Approximations with Kronecker products, in *Linear Algebra for Large Scale and Real-Time Applications*, ed. by M.S. Moonen, G.H. Golub (Kluwer, Dordrecht, 1993), pp. 293–314
24. J. Poulson, B. Marker, R.A. van de Geijn, J.R. Hammond, N.A. Romero, Elemental: A new framework for distributed memory dense matrix computations. ACM Trans. Math. Softw. **39**(2), 13:1–13:24, February 2013 (2014). arXiv:1301.7744
25. S. Ragnarsson, C.F. Van Loan, Block tensor unfoldings. SIAM J. Matrix Anal. Appl. **33**(1), 149–169 (2012)
26. S. Ragnarsson, C.F. Van Loan, Block tensors and symmetric embeddings. Linear Algebra Appl. **438**, 853–874 (2013)
27. A. Smilde, R. Bro, P. Geladi, *Multi-way Analysis with Applications in the Chemical Sciences* (Wiley, Chichester, 2004)
28. E. Solomonik, D. Matthews, J. Hammond, J. Demmel, Cyclops Tensor Framework: reducing communication and eliminating load imbalance in massively parallel contractions, Berkeley Technical Report No. UCB/EECS-2013-1 (2013)
29. C.F. Van Loan, *Computational Frameworks for the Fast Fourier Transform* (SIAM, Philadelphia, PA, 1992)
30. C. Van Loan, The ubiquitous Kronecker product. J. Comput. Appl. Math. **123**, 85–100 (2000)
31. C.F. Van Loan, J.P. Vokt, Approximating matrices with multiple symmetries. SIAM J. Matrix Anal. Appl. **36**(3), 974–993 (2015)

http://www.springer.com/978-3-319-49886-7

Exploiting Hidden Structure in Matrix Computations: Algorithms and Applications
Cetraro, Italy 2015
Benzi, M.; Bini, D.; Kressner, D.; Munthe-Kaas, H.; Van Loan, C. - Benzi, M.; Simoncini, V. (Eds.)
2016, IX, 406 p. 57 illus., 46 illus. in color., Softcover
ISBN: 978-3-319-49886-7