From Matrix to Tensor:  
The Transition to Numerical Multilinear Algebra

Lecture 2. Tensor Unfoldings

Charles F. Van Loan  
Cornell University

The Gene Golub SIAM Summer School 2010  
Selva di Fasano, Brindisi, Italy
What is this Lecture About?

A Common Framework for Tensor Computations...

1. Turn tensor $\mathcal{A}$ into a matrix $A$.
2. Through matrix computations, discover things about $A$.
3. Draw conclusions about tensor $\mathcal{A}$ based on what is learned about matrix $A$.

Let us begin to get ready for this...
Recurring themes...

Given tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, there are many ways to assemble its entries into a matrix $A \in \mathbb{R}^{N_1 \times N_2}$ where $N_1 N_2 = n_1 \cdots n_d$.

For example, $A$ could be a block matrix whose entries are $\mathcal{A}$-slices.

A facility with block matrices and tensor indexing is required to understand the layout possibilities.

Computations with the unfolded tensor frequently involve the Kronecker product. A portion of Lecture 3 is devoted to this important “bridging the gap” matrix operation.
Example: Gradients of Multilinear Forms

If $f: \mathbb{R}^{n_1} \times \mathbb{R}^{n_2} \times \mathbb{R}^{n_3} \to \mathbb{R}$ is defined by

$$f(u, v, w) = \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \sum_{i_3=1}^{n_3} A(i_1, i_2, i_3) u(i_1) v(i_2) w(i_3)$$

and $A \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, then its gradient is given by

$$\nabla f(u, v, w) = \begin{bmatrix} A(1) \otimes v \\ A(2) \otimes u \\ A(3) \otimes u \end{bmatrix}$$

where $A(1), A(2),$ and $A(3)$ are matrix unfoldings of $A$. 
What is this Lecture About?

Tensor Unfoldings of $\mathcal{A} \in \mathbb{R}^{4 \times 3 \times 2}$

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

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

\[
\mathcal{A}_{(3)} = \begin{bmatrix}
  a_{111} & a_{211} & a_{311} & a_{411} & a_{112} & a_{212} & a_{312} & a_{412} \\
  a_{112} & a_{212} & a_{312} & a_{412} & a_{121} & a_{221} & a_{321} & a_{421} \\
  a_{111} & a_{211} & a_{311} & a_{411} & a_{121} & a_{221} & a_{321} & a_{421} \\
  a_{112} & a_{212} & a_{312} & a_{412} & a_{121} & a_{221} & a_{321} & a_{421}
\end{bmatrix}
\]
Lecture 1. Introduction to Tensor Computations
Lecture 2. Tensor Unfoldings
Lecture 3. Transpositions, Kronecker Products, Contractions
Lecture 4. Tensor-Related Singular Value Decompositions
Lecture 5. The CP Representation and Rank
Lecture 6. The Tucker Representation
Lecture 7. Other Decompositions and Nearness Problems
Lecture 8. Multilinear Rayleigh Quotients
Lecture 9. The Curse of Dimensionality
Lecture 10. Special Topics
Definition

A block matrix is a matrix whose entries are matrices.

A 3-by-2 Block Matrix...

\[ A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \\ A_{31} & A_{32} \end{bmatrix} \quad A_{ij} \in \mathbb{R}^{50 \times 70} \]

Regarded as a matrix of scalars, \( A \) is 150-by-140.
The blocks in a block matrix do not have to have uniform size.

\[
A = \begin{bmatrix}
A_{11} & A_{12} \\
A_{21} & A_{22} \\
A_{31} & A_{32}
\end{bmatrix} = \begin{bmatrix}
50\text{-by-70} & 50\text{-by-10} \\
50\text{-by-70} & 50\text{-by-10} \\
20\text{-by-70} & 20\text{-by-10}
\end{bmatrix}
\]
A Block Column Vector...

\[
A = \begin{bmatrix}
A_1 \\
A_2 \\
A_3
\end{bmatrix}
\quad A_i \in \mathbb{R}^{100 \times 50}
\]

A Block Row Vector...

\[
A = \begin{bmatrix}
A_1 & A_2 & A_3
\end{bmatrix}
\quad A_i \in \mathbb{R}^{100 \times 50}
\]
A Column-Partitioned Matrix...

\[ A = \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix} \quad a_i \in \mathbb{R}^{100} \]

A Row-Partitioned Matrix

\[ A = \begin{bmatrix} a_1^T \\ a_2^T \\ a_3^T \end{bmatrix} \quad a_i \in \mathbb{R}^{100} \]
Block Matrices: Addition

A 3-by-2 Example...

\[
\begin{bmatrix}
B_{11} & B_{12} \\
B_{21} & B_{22} \\
B_{31} & B_{32}
\end{bmatrix} + \begin{bmatrix}
C_{11} & C_{12} \\
C_{21} & C_{22} \\
C_{31} & C_{32}
\end{bmatrix} = \begin{bmatrix}
B_{11} + C_{11} & B_{12} + C_{12} \\
B_{21} + C_{21} & B_{22} + C_{22} \\
b_{31} + C_{31} & B_{32} + C_{32}
\end{bmatrix}
\]

The matrices must be partitioned conformably.
An Example...

\[
\begin{bmatrix}
B_{11} & B_{12} \\
B_{21} & B_{22} \\
B_{31} & B_{32}
\end{bmatrix}
\begin{bmatrix}
C_{11} & C_{12} \\
C_{21} & C_{22}
\end{bmatrix}
= 
\begin{bmatrix}
B_{11}C_{11} + B_{12}C_{21} & B_{11}C_{12} + B_{12}C_{22} \\
B_{21}C_{11} + B_{22}C_{21} & B_{21}C_{12} + B_{22}C_{22} \\
B_{31}C_{11} + B_{32}C_{21} & B_{31}C_{12} + B_{32}C_{22}
\end{bmatrix}
\]

The matrices must be partitioned conformably.
An Example...

$$\begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \\ B_{31} & B_{32} \end{bmatrix}^T = \begin{bmatrix} B_{11}^T & B_{21}^T & B_{31}^T \\ B_{12}^T & B_{22}^T & B_{32}^T \end{bmatrix}$$

Different from Block Transposition...

$$\begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \\ B_{31} & B_{32} \end{bmatrix}^{[T]} = \begin{bmatrix} B_{11} & B_{21} & B_{31} \\ B_{12} & B_{22} & B_{32} \end{bmatrix}$$

(More on this later.)
A block matrix can have block matrix entries.

\[
A = \begin{bmatrix}
A_{11} & A_{12} \\
A_{21} & A_{22}
\end{bmatrix}
\]

\[
A_{11} = \begin{bmatrix}
A_{1111} & A_{1112} \\
A_{1121} & A_{1122}
\end{bmatrix}
\]

\[
A_{12} = \begin{bmatrix}
A_{1211} & A_{1212} \\
A_{1221} & A_{1222}
\end{bmatrix}
\]

\[
A_{21} = \begin{bmatrix}
A_{2111} & A_{2112} \\
A_{2121} & A_{2122}
\end{bmatrix}
\]

\[
A_{22} = \begin{bmatrix}
A_{2211} & A_{2212} \\
A_{2221} & A_{2222}
\end{bmatrix}
\]

Example: The Discrete Fourier Transform Matrix \( F_n \)

\[
F_{2m} P = \begin{bmatrix}
F_m & \Omega \cdot F_m \\
F_m & -\Omega \cdot F_m
\end{bmatrix}
\]

\( P = \) permutation, \( \Omega = \) diagonal
The \textit{vec} Operation

Turns matrices into vectors by stacking columns...

\[
X = \begin{bmatrix} 1 & 10 \\ 2 & 20 \\ 3 & 30 \end{bmatrix} \Rightarrow \text{vec}(X) = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 10 \\ 20 \\ 30 \end{bmatrix}
\]
MATLAB: Vec of a matrix using the reshape function.

% If A is a matrix, then the following
% assigns vec(A) to a ...
[n1,n2] = size(A);
a = reshape(A,n1*n2,1);

It can be shown that

\[
a(i_1 + (i_2 - 1)n_1) = A(i_1, i_2)
\]
The vec Operation

Turns tensors into vectors by stacking mode-1 fibers...

\[ \mathcal{A} \in \mathbb{R}^{2 \times 3 \times 2} \Rightarrow \text{vec}(\mathcal{A}) = \begin{bmatrix} \mathcal{A}(::,1,1) \\
\mathcal{A}(::,2,1) \\
\mathcal{A}(::,3,1) \\
\mathcal{A}(::,1,2) \\
\mathcal{A}(::,2,2) \\
\mathcal{A}(::,3,2) \end{bmatrix} = \begin{bmatrix} a_{111} \\
a_{211} \\
a_{121} \\
a_{221} \\
a_{131} \\
a_{231} \\
a_{112} \\
a_{212} \\
a_{122} \\
a_{222} \\
a_{132} \\
a_{232} \end{bmatrix} \]

We need to specify the order of the stacking...
Tensor Notation: Subscript Vectors

Reference

If \( A \in \mathbb{R}^{n_1 \times \cdots \times n_d} \) and \( \mathbf{i} = (i_1, \ldots, i_d) \) with \( 1 \leq i_k \leq n_k \) for \( k = 1:d \), then

\[
A(\mathbf{i}) \equiv A(i_1, \ldots, i_k)
\]

We say that \( \mathbf{i} \) is a subscript vector. Bold font will be used to designate subscript vectors.

Bounds

If \( \mathbf{L} \) and \( \mathbf{R} \) are subscript vectors having the same dimension, then \( \mathbf{L} \leq \mathbf{R} \) means that \( L_k \leq R_k \) for all \( k \).

Special Cases

A subscript vector of all ones is denoted by \( \mathbf{1} \). (Dimension clear from context.) If \( N \) is an integer, then \( \mathbf{N} = N \cdot \mathbf{1} \).
The Index-Mapping Function col

**Definition**
If \( \mathbf{i} \) and \( \mathbf{n} \) are length-\( d \) subscript vectors, then the integer-valued function \( \text{col}(\mathbf{i}, \mathbf{n}) \) is defined by

\[
\text{col}(\mathbf{i}, \mathbf{n}) = i_1 + (i_2 - 1)n_1 + (i_3 - 1)n_1n_2 + \cdots + (i_d - 1)n_1 \cdots n_{d-1}
\]

**The Formal Specification of Vec**
If \( \mathcal{A} \in \mathbb{IR}^{n_1 \times \cdots \times n_d} \) and \( \mathbf{v} = \text{vec}(\mathcal{A}) \), then

\[
a(\text{col}(\mathbf{i}, \mathbf{n})) = \mathcal{A}(\mathbf{i}) \quad 1 \leq i \leq n
\]
MATLAB: **Vec of a tensor using the reshape function**

```matlab
% If $A$ is a $n(1) \times \cdots \times n(d)$, then the
% following assigns vec($A$) to a ...

n = size(A);
a = reshape(A, prod(n), 1);
```

If $A \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ and $a = \text{vec}(A)$ then

$$a(i_1 + (i_2 - 1)n_1 + (i_3 - 1)n_1 n_2 + \cdots + (i_d - 1)n_1 \cdots \cdots n_{d-1}) = A(i_1, i_2, \ldots, i_d)$$
Fibers

A fiber of a tensor $\mathcal{A}$ is a vector obtained by fixing all but one $\mathcal{A}$'s indices. For example, if $\mathcal{A} = \mathcal{A}(1:3, 1:5, 1:4, 1:7)$, 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 fiber. We adopt the convention that a fiber is a column-vector.

**Problem 2.1.** How many fibers are there in the tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$?
MATLAB: Fiber Extraction

```
n = [3 5 4 7];
A = randn(n);

v0 = A(2,:,:4,6);  % v0(1,j,1,1) = A(2,j,4,6)
v1 = squeeze(v0);  % v1(j) = A(2,j,4,6)
[r,c] = size(v1);
if r==1
    v = v1';
else
    v = v1;
end
```

The function `squeeze` removes “singleton dimensions”. In the above, $v_0$ is a fourth-order tensor that has dimension 1 in modes 1, 3, and 4. Sometimes `squeeze` produces a row vector. The if-else is necessary because (by convention) we insist that fibers are column vectors.
Problem 2.2. Write a recursive MATLAB function \( \text{alpha} = \text{Tensor1norm}(A) \) that returns the maximum value of \( \| f \|_1 \) where \( f \) is a fiber of the input tensor \( A \in \mathbb{R}^{m_1 \times \cdots \times n_d} \).
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

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

is a slice. We adopt the convention that the first unfixed index in the tensor is the row index of the slice and the second unfixed index in the tensor is the column index of the slice.

**Problem 2.3.** How many slices are there in the tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ if $n_1 = \cdots = n_d = N$?
**MATLAB: Slice Extraction**

```matlab
n = [3 5 4 7];
A = randn(n);
C0 = A(2,:,:,:);
C = squeeze(C0);
```

In the above, \( C0 \) is a fourth-order tensor that has dimension 1 in modes 1 and 3. \( C \) is a 5-by-7 matrix, a.k.a., 2nd order tensor.

**Problem 2.4.** Write a recursive MATLAB function \( \alpha = \text{MaxFnorm}(A) \) that returns the maximum value of \( \| B \|_F \) where \( B \) is a slice of the input tensor \( A \in \mathbb{R}^{n_1 \times \cdots \times n_d} \).
**Subtensors**

**Subscript Vectors**

If $\mathbf{i} = [i_1, \ldots, i_d]$ and $\mathbf{j} = [j_1, \ldots, j_d]$ are integer vectors, then $\mathbf{i} \leq \mathbf{j}$ means that $i_k \leq j_k$ for $k = 1:d$.

Suppose $\mathcal{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ with $\mathbf{n} = [n_1, \ldots, n_d]$. If $1 \leq \mathbf{i} \leq \mathbf{n}$, then

$$\mathcal{A}(\mathbf{i}) = \mathcal{A}(i_1, \ldots, i_d).$$

**Specification of Subtensors**

Suppose $\mathcal{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ with $\mathbf{n} = [n_1, \ldots, n_d]$. If $1 \leq \mathbf{L} \leq \mathbf{R} \leq \mathbf{n}$, then $\mathcal{A}(\mathbf{L}:\mathbf{R})$ denotes the subtensor

$$B = \mathcal{A}(L_1:R_1, \ldots, L_d:R_d).$$
MATLAB: Assignment to Fibers, Slices, and Subtensors

\[
n = [3 \ 5 \ 4 \ 7];
\]
\[
A = zeros(n);
\]
\[
A(2,5,:,:6) = ones(4,1);
\]
\[
A(1,:,:3,:) = rand(5,7);
\]
\[
A(2:3,4:5,1:2,5:7) = zeros(2,2,2,3);
\]

Problem 2.5. Write a MATLAB function \( A = \text{Sym4}(N) \) that returns a random 4th order tensor \( A \in \mathbb{R}^{N\times N\times N\times N} \) with the property that \( A(i) = A(j) \) whenever \( j(1:2) \) is a permutation of \( i(1:2) \) and \( j(3:4) \) is a permutation of \( i(3:4) \).
The MATLAB Tensor Toolbox

Why?
It provides an excellent environment for learning about tensor computations and for building a facility with multi-index reasoning.

Who?
Bruce W. Bader and Tammy G. Kolda, Sandia Laboratories

Where?
http://csmr.ca.sandia.gov/~tkolda/TensorToolbox/
MATLAB Tensor Toolbox: A Tensor is a Structure

\[
\begin{align*}
& \gg n = [3 \ 5 \ 4 \ 7]; \\
& \gg A\_array = \text{randn}(n); \\
& \gg A = \text{tensor}(A\_array); \\
& \gg \text{fieldnames}(A) \\
\end{align*}
\]

\[
\begin{align*}
\text{ans} &= \\
& \quad '\text{data}' \\
& \quad '\text{size}'
\end{align*}
\]

The .data field is a multi-dimensional array. The .size field is an integer vector that specifies the modal dimensions. In the above, A.data and A.array have same value and A.size and n have the same value.
The Frobenius norm of a 3-tensor

```matlab
function sigma = NormFro3(A)
n = A.size;
s = 0;
for i1=1:n(1)
    for i2=1:n(2)
        for i3=1:n(3)
            s = s + A(i1,i2,i3)^2;
        end
    end
end
s = sqrt(s);
```
MATLAB Tensor Toolbox: Order and Dimension

```matlab
>> n = [3 5 4 7];
>> A_array = randn(n);
>> A = tensor(A_array)
>> d = ndims(A)

d =
    4

>> n = size(A)
n =
    3     5     4     7
```
Problem 2.6. If \( A \) is a \( d \)-dimensional array in Matlab, then what is \( \text{sum}(A) \)? Explain why the following function returns the Frobenius norm of the input tensor.

```matlab
function sigma = NormFro(A)
% A is a tensor.
% sigma is the square root of the sum of the
% squares of its entries.
B = A.data.^2;
for k=1:ndims(A)
    B = sum(B);
end
sigma = sqrt(B);
```
MATLAB Tensor Toolbox: Tensor Operations

\[ n = [3 \ 5 \ 4 \ 7]; \]
% Extracting Fibers and Slices
\[ a_{\text{Fiber}} = A(2,:,:,3,6); \quad a_{\text{Slice}} = A(3,:,:,2,:); \]

% Familiar initializations...
\[ X = \text{tenzeros}(n); \quad Y = \text{tenones}(n); \]
\[ A = \text{tenrandn}(n); \quad B = \text{tenrandn}(n); \]

% These operations are legal...
\[ C = 3*A; \quad C = -A; \quad C = A+1; \quad C = A.^2; \]
\[ C = A + B; \quad C = A./B; \quad C = A.*B; \quad C = A.^B; \]

% Applying a Function to Each Entry...
\[ F = \text{tenfun}(\@\sin,A); \]
\[ G = \text{tenfun}(\@(x) \sin(x)./\exp(x),A); \]
Problem 2.7. Suppose $A = A(1:n_1, 1:n_2, 1:n_3)$. We say $A(i)$ is an interior entry if $1 < i < n$. Otherwise, we say $A(i)$ is an edge entry. If $A(i_1, i_2, i_3)$ is an interior entry, then it has six neighbors: $A(i_1 \pm 1, i_2, i_3), A(i_1, i_2 \pm 1, i_3)$, and $A(i_1, i_2, i_3 \pm 1)$. Implement the following function so that it performs as specified:

```matlab
function B = Smooth(A)
% A is a third order tensor
% B is a third order tensor with the property that each
% interior entry B(i) is the average of A(i)'s six
% neighbors. If B(i) is an edge entry, then B(i) = A(i).
```

Strive for an implementation that does not have any loops.

Problem 2.8. Formulate and solve an order-$d$ version of Problem 2.7.
Example 1. $A = A(1:n_1, 1:n_2, 1:n_3)$

ima_matrix = $\begin{bmatrix}
A(1, :, :)
& \cdots & A(n_1, :, :)
\end{bmatrix}$

ima_matrix = $\begin{bmatrix}
A(:, 1, :)
& \cdots & A(:, n_2, :)
\end{bmatrix}$

ima_matrix = $\begin{bmatrix}
A(:, :, 1)
& \cdots & A(:, :, n_3)
\end{bmatrix}$
Example 2. $\mathcal{A} = \mathcal{A}(1:n_1, 1:n_2, 1:2, 1:2, 1:2)$

$\text{ima\_matrix} = \begin{bmatrix} 
\mathcal{A}(:, :, 1, 1, 1) \\
\mathcal{A}(:, :, 2, 1, 1) \\
\mathcal{A}(:, :, 1, 2, 1) \\
\mathcal{A}(:, :, 2, 2, 1) \\
\mathcal{A}(:, :, 1, 1, 2) \\
\mathcal{A}(:, :, 2, 1, 2) \\
\mathcal{A}(:, :, 1, 2, 2) \\
\mathcal{A}(:, :, 2, 2, 2) 
\end{bmatrix}$
Example 3. $\mathcal{A} = \mathcal{A}(1:n_1, 1:n_2, 1:n_3, 1:n_4)$

$$\text{ima\_matrix} = \begin{bmatrix}
\mathcal{A}(:, :, 1, 1) & \mathcal{A}(:, :, 1, 2) & \cdots & \mathcal{A}(:, :, 1, n_4) \\
\mathcal{A}(:, :, 2, 1) & \mathcal{A}(:, :, 2, 2) & \cdots & \mathcal{A}(:, :, 2, n_4) \\
\vdots & \vdots & \ddots & \vdots \\
\mathcal{A}(:, :, n_3, 1) & \mathcal{A}(:, :, n_3, 2) & \cdots & \mathcal{A}(:, :, n_3, n_4)
\end{bmatrix}$$

$\mathcal{A}(i, j, k, \ell)$ is the $(i, j)$ entry of block $(k, \ell)$
What are they?

A tensor unfolding of \( \mathcal{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d} \) is obtained by assembling \( \mathcal{A} \)'s entries into a matrix \( A \in \mathbb{R}^{N_1 \times N_2} \) where \( N_1 N_2 = n_1 \cdots n_d \).

Obviously, there are many ways to unfold a tensor.

An important family of tensor unfoldings are the mode-\( k \) unfoldings.

In a mode-\( k \) unfolding, the mode-\( k \) fibers are assembled to produce an \( n_k \)-by-\( (N/n_k) \) matrix where \( N = n_1 \cdots n_d \).

The tensor toolbox function `tenmat` can be used to produce modal unfoldings and other, more general unfoldings.
Example: A Mode-1 Unfolding of $A \in \mathbb{R}^{4 \times 3 \times 2}$

\[
\begin{bmatrix}
    a_{11} & a_{12} & a_{13} & a_{14} & a_{15} & a_{16} \\
    a_{21} & a_{22} & a_{23} & a_{24} & a_{25} & a_{26} \\
    a_{31} & a_{32} & a_{33} & a_{34} & a_{35} & a_{36} \\
    a_{41} & a_{42} & a_{43} & a_{44} & a_{45} & a_{46}
\end{bmatrix}
\]

\((1,1), (2,1), (3,1), (1,2), (2,2), (3,2)\)

Notice how the fibers are ordered.
Example: A Mode-2 Unfolding of $A \in \mathbb{R}^{4 \times 3 \times 2}$

$\text{tenmat}(A,2)$ sets up

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

(1,1) (2,1) (3,1) (4,1) (1,2) (2,2) (3,2) (4,2)

Notice how the fibers are ordered.
Example: A Mode-3 Unfolding of $\mathcal{A} \in \mathbb{R}^{4 \times 3 \times 2}$

`tenmat(A,3)` sets up

$$
\begin{bmatrix}
  a_{111} & a_{211} & a_{311} & a_{411} & a_{121} & a_{221} & a_{321} & a_{421} & a_{131} & a_{231} & a_{331} & a_{431} \\
  a_{112} & a_{212} & a_{312} & a_{412} & a_{122} & a_{222} & a_{322} & a_{422} & a_{132} & a_{232} & a_{332} & a_{432} \\
  (1,1) & (2,1) & (3,1) & (4,1) & (1,2) & (2,2) & (3,2) & (4,2) & (1,3) & (2,3) & (3,3) & (4,3)
\end{bmatrix}
$$

Notice how the fibers are ordered.
Precisely how are the fibers ordered?

If $A \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, $N = n_1 \cdots n_d$, and $B = \text{tenmat}(A,k)$, then $B$ is the matrix $A(k) \in \mathbb{R}^{n_k \times (N/n_k)}$ with

\[ A(k)(i_k, \text{col}(\tilde{i}_k, \tilde{n}_k)) = A(i) \]

where

\[ \tilde{i}_k = [i_1, \ldots, i_{k-1}, i_{k+1}, \ldots, i_d] \]
\[ \tilde{n}_k = [n_1, \ldots, n_{k-1}, n_{k+1}, \ldots, m_d] \]

Recall that the col function maps multi-indices to integers. For example, if $n = [2 \ 3 \ 2]$ then

<table>
<thead>
<tr>
<th>$i$</th>
<th>$1, 1, 1$</th>
<th>$2, 1, 1$</th>
<th>$1, 2, 1$</th>
<th>$2, 2, 1$</th>
<th>$1, 3, 1$</th>
<th>$2, 3, 1$</th>
<th>$1, 1, 2$</th>
<th>$2, 1, 2$</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>col($i$, $n$)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>...</td>
</tr>
</tbody>
</table>
MATLAB Tensor Toolbox: A Tenmat Is a Structure

A2.tsize = size of the unfolded tensor = [3 5 4 7]
A2.rindices = mode indices that define A2’s rows = [2]
A2.cindices = mode indices that define A2’s columns = [1 3 4]
A2.data = the matrix that is the unfolded tensor.
Other Modal Unfoldings Using \texttt{tenmat}

\textbf{Mode-}k\textbf{ Unfoldings of }\mathbf{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}

A particular mode-\(k\) unfolding is defined by a permutation \(\mathbf{v}\) of \([1:k−1\ k+1:d]\). Tensor entries get mapped to matrix entries as follows

\[ \mathbf{A}(i) \rightarrow \mathbf{A}(i_k, \text{col}(i(\mathbf{v}), \text{n}(\mathbf{v}))) \]

<table>
<thead>
<tr>
<th>Name</th>
<th>(\mathbf{v})</th>
<th>How to get it...</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\mathbf{A}(k)) (Default)</td>
<td>([1:k−1\ k+1:d])</td>
<td>\texttt{tenmat(A,k)}</td>
</tr>
<tr>
<td>Forward Cyclic</td>
<td>([k + 1:d\ 1:k−1])</td>
<td>\texttt{tenmat(A,k,'fc')}</td>
</tr>
<tr>
<td>Backward Cyclic</td>
<td>([k − 1:−1:1\ d:−1:k+1])</td>
<td>\texttt{tenmat(A,k,'bc')}</td>
</tr>
<tr>
<td>Arbitrary</td>
<td>(\mathbf{v})</td>
<td>\texttt{tenmat(A,k,v)}</td>
</tr>
</tbody>
</table>
Problem 2.9. Given that $\mathcal{A} = \mathcal{A}(1:n_1, 1:n_2, 1:n_3)$, show how `tenmat` can be used to compute the following unfoldings:

$$A_1 = \begin{bmatrix} \mathcal{A}(1,:,:) & \cdots & \mathcal{A}(n_1,:,:) \end{bmatrix}$$

$$A_2 = \begin{bmatrix} \mathcal{A}(:,1,:) & \cdots & \mathcal{A}(:,n_2,:) \end{bmatrix}$$

$$A_3 = \begin{bmatrix} \mathcal{A}(:,,:),1 & \cdots & \mathcal{A}(:,,:),n_3 \end{bmatrix}$$

Problem 2.10. How can `tenmat` be used to compute the `vec` of a tensor?
A block matrix with uniformly sized blocks

\[
A = \begin{bmatrix}
A_{11} & \cdots & A_{1,c_2} \\
\vdots & \ddots & \vdots \\
A_{r_2,1} & \cdots & A_{r_2,c_2}
\end{bmatrix}
\]

has several natural reshapings:

\[
A_{i_2,j_2}(i_1,j_1) \leftrightarrow A(i_1,i_2,j_1,j_2) \quad A \in \mathbb{R}^{r_1 \times r_2 \times c_1 \times c_2}
\]

The function \texttt{tenmat} can be used to set up these unfoldings...
A 4th Order Tensor is a Block Matrix

Example 1.

If $A \in \mathbb{R}^{n_1 \times n_2 \times n_3 \times n_4}$, then $\text{tenmat}(A, [1\ 2], [3\ 4])$ sets up

$$
\begin{bmatrix}
A(1, 1, ::) & \cdots & A(1, n_2, ::) \\
\vdots & \ddots & \vdots \\
A(n_1, 1, ::) & \cdots & A(n_1, n_2, ::)
\end{bmatrix}
$$

Example 2.

If $A \in \mathbb{R}^{n_1 \times n_2 \times n_3 \times n_4}$, then $\text{tenmat}(A, [1\ 3], [2\ 4])$ sets up

$$
\begin{bmatrix}
A(1, ::, 1) & \cdots & A(1, ::, n_3) \\
\vdots & \ddots & \vdots \\
A(n_1, ::, 1) & \cdots & A(n_1, ::, n_3)
\end{bmatrix}
$$
Even More General Unfoldings

Example: \( A = A(1:2, 1:3, 1:2, 1:2, 1:3) \)

\[
B = \text{tenmat}(A, [1 2 3], [4 5])
\]
MATLAB Tensor Toolbox: General Unfoldings Using `tenmat`

```matlab
function A_unfolded = Unfold(A,rIdx,cIdx)

% A = A(1:n(1),...,1:n(d)) is a tensor.
% rIdx and cIdx are integer vectors with the
% property that [rIdx cIdx] is a
% permutation of 1:d.

% A_unfolded is an nRows-by-nCols matrix where
% nRows = prod(n(rIdx))
% nCols = prod(n(cIdx))
% if 1<=i<=n, then
% A(i) = A_unfolded(r,c)
% with
% r = col(i(rIdx),n(rIdx))
% c = col(i(cIdx),n(cIdx))
A_unfolded = tenmat(A,rIdx,cIdx)
```

⊗ Transition to Computational Multilinear Algebra
⊗ Lecture 2. Tensor Unfoldings
A block matrix is a matrix whose entries are matrices.

A fiber of a tensor is a column vector defined by fixing all but one index and varying what’s left. A slice of a tensor is a matrix defined by fixing all but two indices and varying what’s left.

A mode-$k$ unfolding of a tensor is obtained by assembling all the mode-$k$ fibers into a matrix.

tensor is a Tensor Toolbox command used to construct tensors.

tenmat is a Tensor Toolbox command that is used to construct unfoldings of a tensor.