# Tensor Methods for Computational Efficiency

James A. Nguyen
University of California, Irvine

December 5, 2024

# Table of Contents (Summary)


1 Define Tensor

2 Efficient Matrix Operations

3 Tensor Properties

4 Integration

# What is a Tensor?

Often reduced to an array, generalized to higher dimensions

- Ignores interdimensional relationships
- Implies flattening tensor results in no information loss

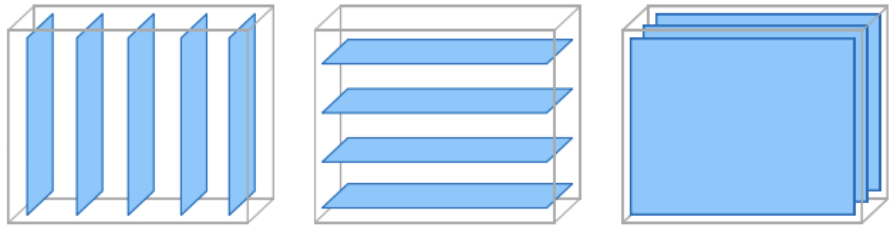


Figure: Order-3 tensor $\mathcal{T}_3 \in \mathbb{R}^{I_1 \times I_2 \times I_3}$

- Functionally, similar to a matrix transformation

# Tensor Size

Elements defined by the tensor product $\pi : U \times V \mapsto U \underset{\mathcal{T}}{\otimes} V$

- $\pi(u, v) = u \underset{\mathcal{T}}{\otimes} v$
- $(\mathcal{T}_n)_{i_1,\ldots,i_n} := v_{i_1} v_{i_2} \cdots v_{i_n}$

Can further describe the size of a tensor by their rank $R$ and order $n$

$$\mathcal{T}_n = \sum_{r=1}^{R} v_1^{(r)} \underset{\mathcal{T}}{\otimes} \cdots \underset{\mathcal{T}}{\otimes} v_n^{(r)}$$

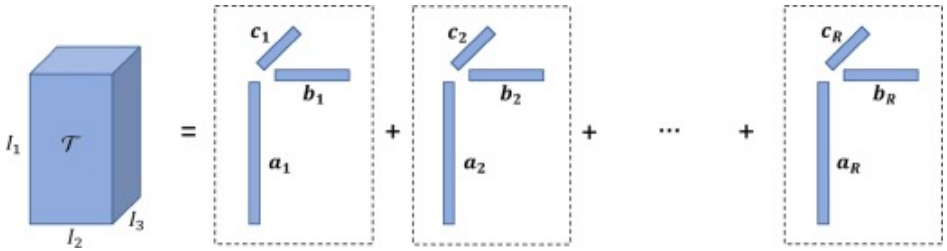



Figure: Rank Decomposition of Order-3 Tensor

# Motivating Example

Why matrix multiplication?

Expensive computations in physics and computer science rely on high-order matrix operations for regression.

Examples:

1. Graphics rendering,
2. Learning algorithms,
3. Cryptography,
4. Simulations, etc.

all perform matrix multiplication in high frequency (trillions)

# Efficient Matrix Operations

1969: Strassen's Algorithm reduces cost of square matrix multiplication.

- Partitions into 7 subproblems

$$
\begin{aligned}
I &= (A_{11} + A_{22})(B_{11} + B_{22}), \\
II &= (A_{21} + A_{22})B_{11}, \\
III &= A_{11}(B_{12} - B_{22}), \\
IV &= A_{22}(-B_{11} + B_{21}),
\end{aligned}
\qquad
\begin{aligned}
V &= (A_{11} + A_{12})B_{22}, \\
VI &= (-A_{11} + A_{21})(B_{11} + B_{12}), \\
VII &= (A_{12} - A_{22})(B_{21} + B_{22})
\end{aligned}
$$

$$
AB = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} (I + IV - V + VII) & (II + IV) \\ (III + IV) & (I + III - II + VI) \end{bmatrix}
$$

- Only 7 multiplications to compute $2 \times 2$ matrix product.

# Time Complexity

Using sub-matrix multiplication, we apply algorithm recursively to compute order-$n$ products, where $n = 2^k$ for some $k \in \mathbb{N}$.

$\implies$ Classify as divide-and-conquer recursive algorithm

$\implies$ Master Theorem: total complexity $T(n)$ for order-$n$ matrices,

$$T(n) = O(n^{\log_b a}) = O(n^{\log_2 7}),$$

where $a$ denotes number of subproblems, $b$ is factor by which the problem is reduced.

## Winograd's Lower Bound

"Order-2 matrix product requires at least 7 multiplications"

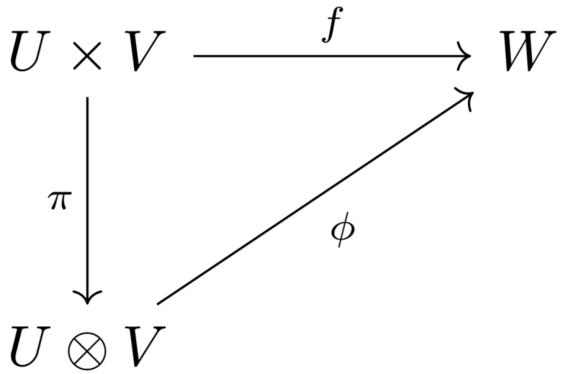
# Tensor Properties

Recall: $\forall\ U \underset{\mathcal{T}}{\otimes} V,\ \exists \pi : U \times V \mapsto U \underset{\mathcal{T}}{\otimes} V$

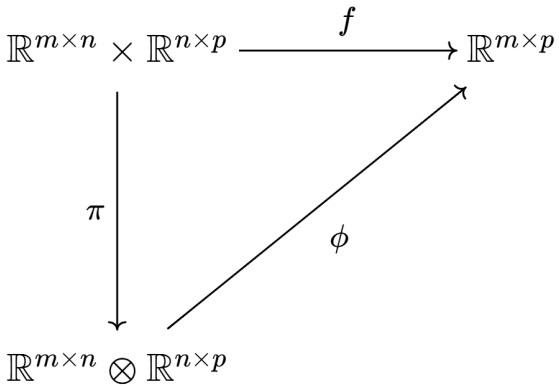
## Theorem (Universal Property of Tensor Products)

*For any pair* $(U \underset{\mathcal{T}}{\otimes} V, \pi)$, *if* $\exists f : U \times V \mapsto W$ *bilinear, then*
$\exists\ \phi : U \underset{\mathcal{T}}{\otimes} V \mapsto W$ *that extends* $\pi : U \times V \mapsto U \underset{\mathcal{T}}{\otimes} V$ *such that*

1. $\phi$ *is unique,*
2. $\phi \circ \pi = f$, *and the following diagram commutes:*



Note: Matrix multiplication is a bilinear map $\implies \exists\ \phi : \mathbb{R}^{m\times n\times p} \mapsto \mathbb{R}^{m\times p}$.




Then every algorithm can be represented by some order-3 tensor $\mathcal{T}_{m,n,p} \in U \underset{\mathcal{T}}{\otimes} V$ such that $\phi \circ \pi(A, B) = \phi(\mathcal{T}_{m,n,p}) = C$, where

$$\mathcal{T}_{m,n,p} = \sum_{r=1}^{R} \mathbf{u}^{(r)} \otimes \mathbf{v}^{(r)} \otimes \mathbf{w}^{(r)}$$

and $R$ denotes $rank(\mathcal{T}_{m,n,p})$ and number of multiplications.

## Example: Decomposition of Strassen's Algorithm

$$
\begin{aligned}
I &= (a_1 + a_4)(b_1 + b_4) \\
II &= (a_3 + a_4)b_1 \\
III &= a_1(b_2 - b_4) \\
IV &= a_4(b_3 - b_1) \\
V &= (a_1 + a_2)b_4 \\
VI &= (a_3 - a_1)(b_1 + b_2) \\
VII &= (a_2 - a_4)(b_3 + b_4) \\
C_{11} &= I + IV - V + VII \\
C_{12} &= I + III \\
C_{21} &= II + IV \\
C_{22} &= I - II + III + VI
\end{aligned}
$$

$$
A = \begin{bmatrix}
1 & 0 & 1 & 0 & 1 & -1 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 & 1 & 0 \\
1 & 1 & 0 & 1 & 0 & 0 & -1
\end{bmatrix},
$$

$$
B = \begin{bmatrix}
1 & 1 & 0 & -1 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 1 \\
1 & 0 & -1 & 0 & 1 & 0 & 1
\end{bmatrix},
$$

$$
C = \begin{bmatrix}
1 & 0 & 0 & 1 & -1 & 0 & 1 \\
0 & 0 & 1 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 \\
1 & -1 & 1 & 0 & 0 & 1 & 0
\end{bmatrix}.
$$

Let $a^{(r)}, b^{(r)}, c^{(r)}$ denote the $r^{th}$ columns of $A, B, C$ respectively

$$
\mathcal{T}_2 = \sum_{r=1}^{7} \mathbf{a}^{(r)} \otimes \mathbf{b}^{(r)} \otimes \mathbf{c}^{(r)}
$$

# Results

Goal: Use reinforcement learning to discover new algorithms that challenge limits of efficiency

Procedure: Begin with order-$n$ iteration tensor $\mathcal{S}_n$, subtract tensor product of $n$ vectors until we reach the 0-vector after $R$ subtractions.

$$\underset{\{v_1^{(r)},\ldots,v_n^{(r)}\}_{r=1}^{R}}{argmin} \quad (R), \text{ subject to : } \mathcal{S}_n - \sum_{r=1}^{R} \mathbf{v}_1^{(r)} \otimes \cdots \otimes \mathbf{v}_n^{(r)} = 0$$

Interpret: $R$ determines the rank of $\mathcal{T}_n = \sum_{r=1}^{R} \mathbf{v}_1^{(r)} \otimes \cdots \otimes \mathbf{v}_n^{(r)}$, number multiplications for order-$n$ matrix product

Results: Order-4 matrix product requires only 47 multiplications

- $T(n) = O(n^{\log_4 47}) < O(n^{\log_2 7})$
- Winograd's lower bound doesn't scale to $7^{n/2}$ for order-$n$ matrices

# Summary

Post-presentation notes:

Identifying matrices as bilinear maps, we can leverage the properties of a tensor to algorithmically search for more efficient matrix multiplication algorithms.

In this case, we find vector triples whose tensor product completely deconstruct an initial tensor such that we can interpret their sum to be the tensor factor representation of some efficient algorithm.

Considering other bilinear maps, it would be interesting to see how we could use the same property to make similar advancements in the field of numerical linear algebra.