Matrix-Mimetic Tensor Algebra: A Unified Framework for Optimal Decomposition and Equivariant Learning

Haim Avron[4] Ken Clarskon[1] Lior Horesh[1] Ben Huh[1] Vasileios Kalantzis[1]

Misha Kilmer[2] Osman Malik[5] Elizabeth Newman[2] Paulina Hoyos Restrepo[6]

Fan Tian[3] Shashanka Ubaru[1]

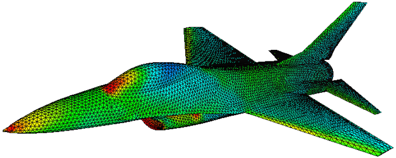
[1]IBM Research [2]Tufts University [3]ICERM

[4]Tel-Aviv University [5]Encube Technologies [6]UT Austin

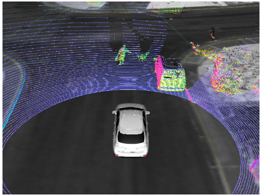
Introduction

- Much of real-world **data** is inherently **multidimensional**

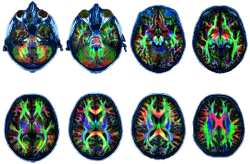
- Many **operators** and **models** are natively **multi-way**



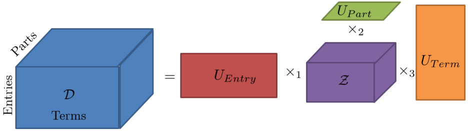
Tensor Applications



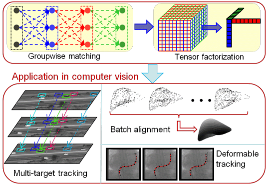
Machine vision



Medical imaging



Latent semantic tensor indexing



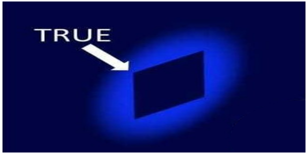
Video surveillance, streaming

Ivanov, Mathies, Vasilescu, **Tensor subspace analysis for viewpoint recognition**, ICCV, 2009

Shi, Ling, Hu, Yuan, Xing, **Multi-target tracking with motion context in tensor power iteration**, CVPR, 2014

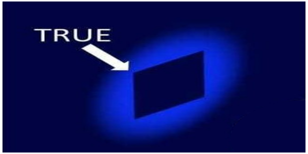
Where the Truth Lies ?

- If this is true

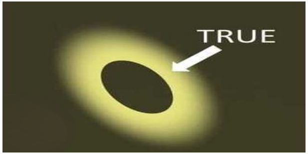


Where the Truth Lies ?

- If this is true

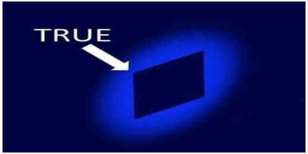


- But also this is true...

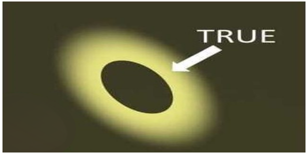


Where the Truth Lies ?

- If this is true



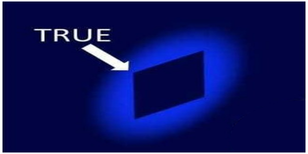
- But also this is true...



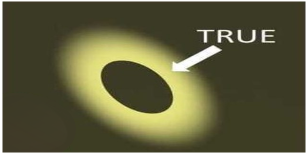
- Where the truth lies ?

Where the Truth Lies ?

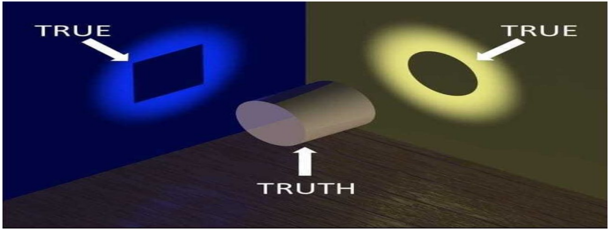
- If this is true



- But also this is true...

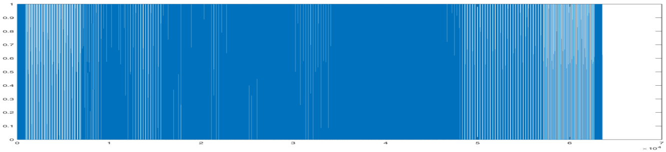


- Where the truth lies ?



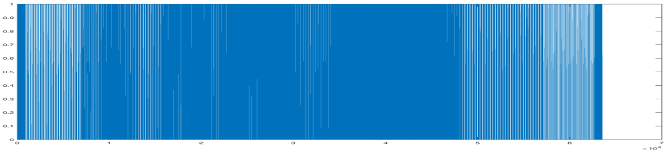
The Power of Representation

- Traditional **matrix-based** methods assuming data **vectorization** are generally **agnostic** to possible **high dimensional correlations**
- What is that ?



The Power of Representation

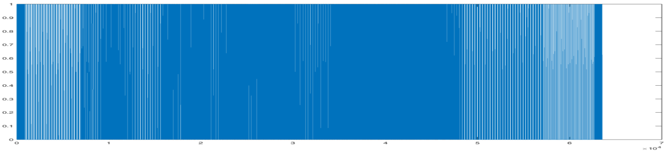
- Traditional **matrix-based** methods assuming data **vectorization** are generally **agnostic** to possible **high dimensional correlations**
- What is that ?



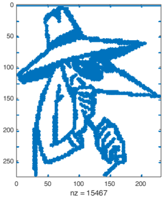
- Observe the same data but in a different (matrix rather than vector) representation

The Power of Representation

- Traditional **matrix-based** methods assuming data **vectorization** are generally **agnostic** to possible **high dimensional correlations**
- What is that ?



- Observe the same data but in a different (matrix rather than vector) representation



- **Representation matters!** some correlations can only be realized in appropriate representation

Background and Notation

- **Notation** : $\mathcal{A}^{n_1 \times n_2 \ldots, \times n_d}$ - d^{th} order tensor
  - 0^{th} order tensor - scalar

Background and Notation

- **Notation** : $\mathcal{A}^{n_1 \times n_2 \ldots, \times n_d}$ - d^{th} order tensor
  - 0^{th} order tensor - scalar

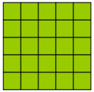
  - 1^{st} order tensor - vector

Background and Notation

- **Notation** : $\mathcal{A}^{n_1 \times n_2 \ldots, \times n_d}$ - d^{th} order tensor
  - 0^{th} order tensor - scalar

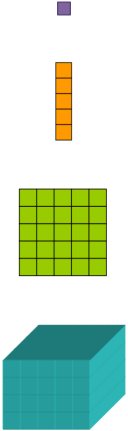
  - 1^{st} order tensor - vector

  - 2^{nd} order tensor - matrix



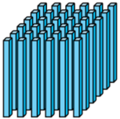
Background and Notation

- **Notation** : $\mathcal{A}^{n_1 \times n_2 \ldots, \times n_d}$ - d^{th} order tensor
  - 0^{th} order tensor - scalar
  - 1^{st} order tensor - vector
  - 2^{nd} order tensor - matrix
  - 3^{rd} order tensor ...

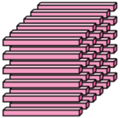


Inside the Box

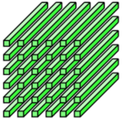
- *Tube (Fiber)* - a vector defined by fixing all but one index while varying the rest



mode-1
$\mathcal{A}_{:,j,k}$

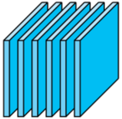


mode-2
$\mathcal{A}_{i,:,k}$

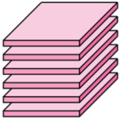


mode-3
$\mathcal{A}_{i,j,:}$, $\boldsymbol{a}_{ij}$

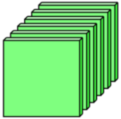
- *Slice* - a matrix defined by fixing all but two indices while varying the rest



lateral
$\mathcal{A}_{:,j,:}$, $\vec{\mathcal{A}}_j$



horizontal
$\mathcal{A}_{i,:,:}$



frontal
$\mathcal{A}_{:,:,k}$, $\boldsymbol{A}^{(k)}$

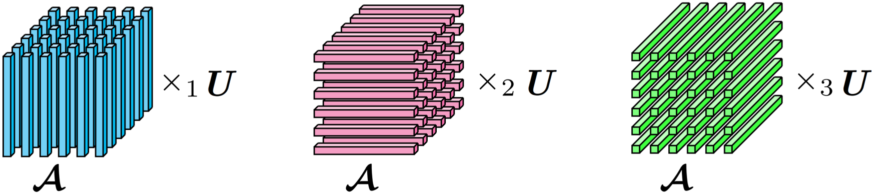
Tensor Multiplication

Definition

- The k - mode multiplication of a tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$ with a matrix $U \in \mathbb{R}^{j \times n_k}$ is denoted by $\mathcal{A} \times_k U$ and is of size $n_1 \times \cdots \times n_{k-1} \times j \times n_{k+1} \times \cdots \times n_d$
- Element-wise

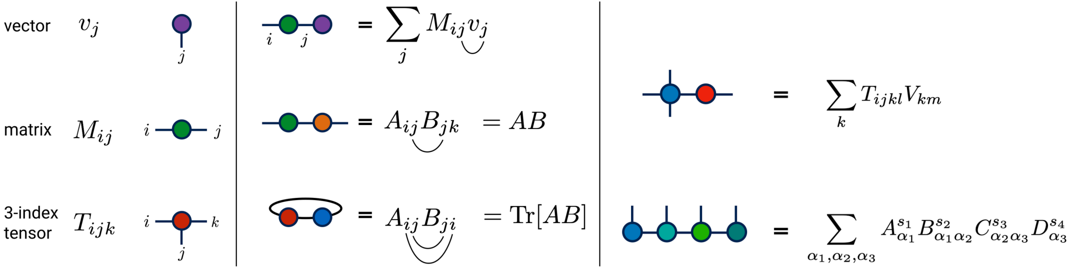
$$(\mathcal{A} \times_k U)_{i_1 \cdots i_{k-1} j i_{k+1} \cdots i_d} = \sum_{i_k=1}^{n_k} a_{i_1 i_2 \cdots i_d} u_{j i_k}$$

- k-mode multiplication



Tensor Network / Diagram Notation

- Tensor networks invariants / isomorphism offers means to analyze and identify (space and time complexity) structure in high dimensional computation



- **Tensors** are notated by **nodes**, while **indices** are represented by **edges**
- **Connecting** index **edges** implies **contraction** / summation over connected indices

How Powerful are Tensor Networks ?

MENU

nature

Article | Published: 23 October 2019

Quantum supremacy using a programmable superconducting processor

Frank Arute, Kunal Arya, [...] John M. Martinis

Nature **574**, 505–510(2019) | Cite this article

Abstract

The promise of quantum computers is that certain computational tasks might be executed exponentially faster on a quantum processor than on a classical processor[1]. A fundamental challenge is to build a high-fidelity processor capable of running quantum algorithms in an exponentially large computational space. Here we report the use of a processor with programmable superconducting qubits[2,3,4,5,6,7] to create quantum states on 53 qubits, corresponding to a computational state-space of dimension 2^{53} (about 10^{16}). Measurements from repeated experiments sample the resulting probability distribution, which we verify using classical simulations. Our Sycamore processor takes about 200 seconds to sample one instance of a quantum circuit a million times—our benchmarks currently indicate that the equivalent task for a state-of-the-art classical supercomputer would take approximately 10,000 years. This dramatic increase in speed compared to all known classical algorithms is an experimental realization of quantum supremacy[8,9,10,11,12,13,14] for this specific computational task, heralding a much-anticipated computing paradigm.

Science | Contents | News | Careers | Journals

SHARE

Google researchers in Santa Barbara, California, say their advance may lead to near-term applications of quantum computers. ISTOCK.COM/JHVEPHOTO

IBM casts doubt on Google's claims of quantum supremacy

By Adrian Cho | Oct. 23, 2019 , 5:40 AM

Pareto-Efficient Quantum Circuit Simulation Using Tensor Contraction Deferral*

Edwin Pednault[†1], John A. Gunnels[‡1], Giacomo Nannicini[‡1], Lior Horesh[1], Thomas Magerlein[2], Edgar Solomonik[3], Erik W. Draeger[4], Eric T. Holland[4], and Robert Wisnieff[1]

[1]IBM T.J. Watson Research Center, Yorktown Heights, NY
[2]Tufts University, Medford, MA
[3]Dept. of Computer Science, University of Illinois at Urbana-Champaign, Champaign, IL
[4]Lawrence Livermore National Laboratory, Livermore, CA

Low Rank Structure

Low Rank Property

- Express a d^{th}-order tensor $\mathcal{A}$ as the sum of rank-1 tensors,

$$\mathcal{A} \approx \sum_{i=1}^{r} \sigma_i \cdot u_i^{(1)} \circ u_i^{(2)} \circ \ldots \circ u_i^{(d)}$$

- We seek a d^{th}-order tensor $\mathcal{B}$ of rank $k \leq r$ to optimize:

$$\begin{aligned} \underset{\mathcal{B}}{\text{argmin}} \quad & \|\mathcal{A} - \mathcal{B}\|_F \\ \text{s.t.} \quad & \mathcal{B} \text{ has rank } k \leq r \end{aligned}$$

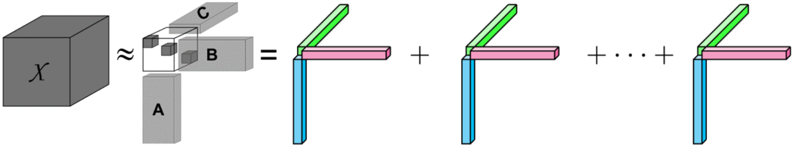
- When $d = 2$, $\mathcal{B}$ is the matrix SVD truncated to k terms (Eckart-Young)
- Is there is a similar theoretical result for higher order tensors ?

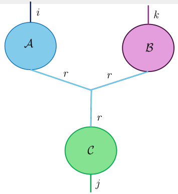
Theorem (Schmidt 1907; Eckart & Young 1936; Mirsky 1960)

Let $\mathbf{A} \in \mathbb{C}^{n_1 \times n_2}$ be a matrix with rank$(\mathbf{A}) = r$. The truncated Singular Value Decomposition (SVD) yields the **best low-rank approximation***; i.e., for $k \leq r$,*

$$\mathbf{A}_k = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^H = \underset{rank(\mathbf{B}) \leq k}{\text{argmin}} \|\mathbf{A} - \mathbf{B}\|_F$$

Tensor Decompositions - CP (CANDECOMP-PARAFAC) [2]





- Find the best **tensor rank-r** fit[1]:

$$\min_{a_i, b_i, c_i} \|\mathcal{X} - \sum_{i=1}^{r} \sigma_i \cdot a_i \circ b_i \circ c_i\|_F$$

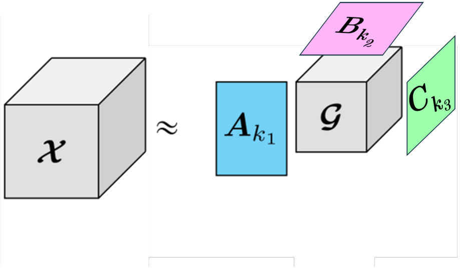
  - Extension of matrix rank
  - Interpretable
  - Summing k factors is sub-optimal
  - Determining rank is NP-hard

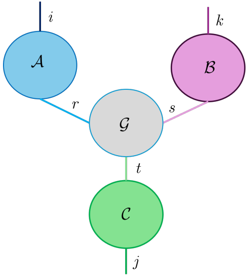
- The set of tensors of a given size that **do not** have a best rank-k approximation has **positive volume** (i.e., positive Lebesgue measure) for at least some values of k, implying that **lack of best approximation** is rather common

[1] de Silva, Lim, Tensor rank and ill-posedness of the best low-rank approximation problem, 2008

[2] Hitchcock, J Math Phys, 1927; Harshman, 1970; Carroll, Chang, 1970

Tensor Decompositions - Tucker [4]





- Find the best **multi-linear rank-**(k_1, k_2 ,k_3) fit[3]:

$$\min_{\mathbf{A}_{k_1},\mathbf{B}_{k_2},\mathbf{C}_{k_3}} \|\mathcal{X} - \mathcal{G} \times_1 \mathbf{A}_{k_1} \times_2 \mathbf{B}_{k_2} \times_3 \mathbf{C}_{k_3}\|_F$$

  - Higher-order PCA
  - Compressible
  - Truncation of full orth. sub-optimal
  - Hard to interpret

[3] De Lathauwer, De Moor, Vandewalle, HOSVD, 2000; Cichocki, Zdunek, Phan, Amari, Nonnegative T&M Factorizations, 2009

[4] Tucker, Problems in Measuring Change, 1963

Tensor Train Decomposition

Tensor Train format of a tensor $\boldsymbol{\mathcal{A}}$

$$\boldsymbol{\mathcal{A}}(i_1,\ldots,i_d) = \sum_{\alpha_0,\ldots,\alpha_d} \boldsymbol{\mathcal{G}}_1(\alpha_0, i_1, \alpha_1)\boldsymbol{\mathcal{G}}_2(\alpha_1, i_2, \alpha_2),\ldots,\boldsymbol{\mathcal{G}}_d(\alpha_{d-1}, i_d, \alpha_d)$$

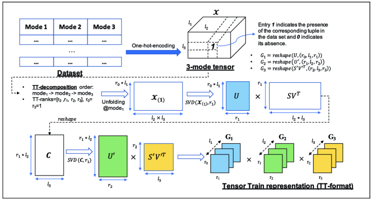
Can be represented compactly as a matrix product:

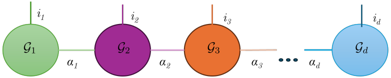
$$\boldsymbol{\mathcal{A}}(i_1,\ldots,i_d) = \underbrace{\boldsymbol{\mathcal{G}}_1[i_1]}_{1\times r_1}\underbrace{\boldsymbol{\mathcal{G}}_2[i_2]}_{r_1\times r_2}\ldots\underbrace{\boldsymbol{\mathcal{G}}_d[i_d]}_{r_{d-1}\times 1}$$

- $\boldsymbol{\mathcal{G}}_i$: TT-cores (collections of matrices)
- r_i: TT-ranks
- $r = \max r_i$: the maximal TT-rank

TT uses $\mathcal{O}(dnr^2)$ memory to store $\mathcal{O}(nd)$ elements

Efficient **only** when all **ranks are small**





Oseledets, Tensor-train decomposition, 2011

Tensor Algebra Desired Properties

Compressible	**Efficient**	**Optimal**	**Mimetic**
Powerful representation	Simple implementation	Provable superiority	Matrix property preservation

Groups and Representations

A **group** is a set G with a binary operation $\circ : G \times G \to G$ satisfying the following axioms:

Group Axioms

- **Closure**: For all $a, b \in G$, $a \circ b \in G$
- **Associativity**: For all $a, b, c \in G$, $(a \circ b) \circ c = a \circ (b \circ c)$
- **Identity**: There exists $e \in G$ such that for all $a \in G$, $e \circ a = a \circ e = a$
- **Inverses**: For each $a \in G$, there exists $a^{-1} \in G$ such that $a \circ a^{-1} = a^{-1} \circ a = e$

Groups and Representations

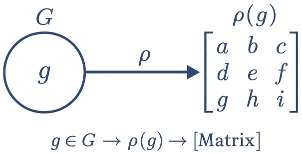
A **group** is a set G with a binary operation $\circ : G \times G \to G$ satisfying the following axioms:

Group Axioms

- **Closure**: For all $a, b \in G$, $a \circ b \in G$
- **Associativity**: For all $a, b, c \in G$, $(a \circ b) \circ c = a \circ (b \circ c)$
- **Identity**: There exists $e \in G$ such that for all $a \in G$, $e \circ a = a \circ e = a$
- **Inverses**: For each $a \in G$, there exists $a^{-1} \in G$ such that $a \circ a^{-1} = a^{-1} \circ a = e$

A group **representation** is a homomorphism $\varrho : G \to \mathrm{GL}(V)$ mapping a group G to the general linear group of a vector space V, preserving the group operation: i.e.

$$\varrho(g_1 \circ g_2) = \varrho(g_1)\varrho(g_2), \quad \forall g_1, g_2 \in G$$



Rings

A **ring** is a set R with two binary operations $+ : R \times R \to R$ and $\cdot : R \times R \to R$ satisfying:

Ring Axioms

- **Abelian Group under** $+$: $(R, +)$ forms an abelian group with identity 0
- **Associative Multiplication**: For all $a, b, c \in R$, $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
- **Distributivity**: For all $a, b, c \in R$:
  - $a \cdot (b + c) = a \cdot b + a \cdot c$ (left distributivity)
  - $(a + b) \cdot c = a \cdot c + b \cdot c$ (right distributivity)

Rings

A **ring** is a set R with two binary operations $+ : R \times R \to R$ and $\cdot : R \times R \to R$ satisfying:

Ring Axioms

- **Abelian Group under** $+$: $(R, +)$ forms an abelian group with identity 0
- **Associative Multiplication**: For all $a, b, c \in R$, $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
- **Distributivity**: For all $a, b, c \in R$:
  - $a \cdot (b + c) = a \cdot b + a \cdot c$ (left distributivity)
  - $(a + b) \cdot c = a \cdot c + b \cdot c$ (right distributivity)

Additional properties may include:

- **Unity**: Multiplicative identity $1 \in R$ with $1 \cdot a = a \cdot 1 = a$ for all $a \in R$
- **Commutativity**: $a \cdot b = b \cdot a$ for all $a, b \in R$

Rings

A **ring** is a set R with two binary operations $+ : R \times R \to R$ and $\cdot : R \times R \to R$ satisfying:

Ring Axioms

- **Abelian Group under** $+$: $(R, +)$ forms an abelian group with identity 0
- **Associative Multiplication**: For all $a, b, c \in R$, $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
- **Distributivity**: For all $a, b, c \in R$:
  - $a \cdot (b + c) = a \cdot b + a \cdot c$ (left distributivity)
  - $(a + b) \cdot c = a \cdot c + b \cdot c$ (right distributivity)

Additional properties may include:

- **Unity**: Multiplicative identity $1 \in R$ with $1 \cdot a = a \cdot 1 = a$ for all $a \in R$
- **Commutativity**: $a \cdot b = b \cdot a$ for all $a, b \in R$

Examples: $\mathbb{Z}$, $\mathbb{Q}[x]$, $M_n(\mathbb{R})$, $\mathbb{Z}/n\mathbb{Z}$

Modules

Let $\mathcal{R}$ be a ring. A **left $\mathcal{R}$-module** is an abelian group $(M, +)$ with scalar multiplication $\cdot : \mathcal{R} \times M \to M$ satisfying:

Module Axioms

- **Distributivity over module addition**: $r \cdot (m + n) = r \cdot m + r \cdot n$
- **Distributivity over ring addition**: $(r + s) \cdot m = r \cdot m + s \cdot m$
- **Associativity**: $(rs) \cdot m = r \cdot (s \cdot m)$
- **Unity** (if R has 1): $1 \cdot m = m$

for all $r, s \in \mathcal{R}$ and $m, n \in M$.

Modules

Let $\mathcal{R}$ be a ring. A **left $\mathcal{R}$-module** is an abelian group $(M, +)$ with scalar multiplication $\cdot : \mathcal{R} \times M \to M$ satisfying:

Module Axioms

- **Distributivity over module addition**: $r \cdot (m + n) = r \cdot m + r \cdot n$
- **Distributivity over ring addition**: $(r + s) \cdot m = r \cdot m + s \cdot m$
- **Associativity**: $(rs) \cdot m = r \cdot (s \cdot m)$
- **Unity** (if R has 1): $1 \cdot m = m$

for all $r, s \in \mathcal{R}$ and $m, n \in M$.

When $\mathcal{R}$ is a **field**, an $\mathcal{R}$-module is precisely a **vector space** over $\mathcal{R}$.

Modules

Let $\mathcal{R}$ be a ring. A **left $\mathcal{R}$-module** is an abelian group $(M, +)$ with scalar multiplication $\cdot : \mathcal{R} \times M \to M$ satisfying:

Module Axioms

- **Distributivity over module addition**: $r \cdot (m + n) = r \cdot m + r \cdot n$
- **Distributivity over ring addition**: $(r + s) \cdot m = r \cdot m + s \cdot m$
- **Associativity**: $(rs) \cdot m = r \cdot (s \cdot m)$
- **Unity** (if R has 1): $1 \cdot m = m$

for all $r, s \in \mathcal{R}$ and $m, n \in M$.

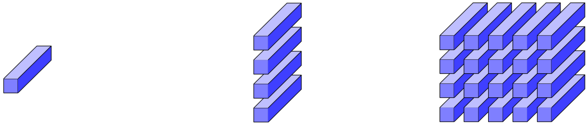
When $\mathcal{R}$ is a **field**, an $\mathcal{R}$-module is precisely a **vector space** over $\mathcal{R}$.

Examples: Abelian groups (as $\mathbb{Z}$-modules), $\mathbb{R}^n$ (as $\mathbb{R}$-module), $\mathbb{Z}/n\mathbb{Z}$ (as $\mathbb{Z}$-module)

Algebra over Tubes

Definition

A tensor $\mathcal{X} \in \mathbb{F}^{m \times p \times n}$ is an $m \times p$ matrix of tubes - $\mathbb{F}_n$ elements



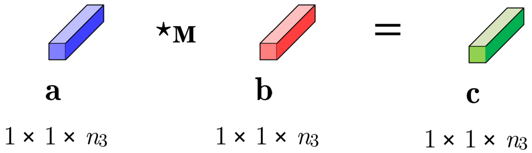
$x \in \mathbb{F}_n$ - tube $\quad X \in \mathbb{F}_n^p$ - slice $\quad \mathcal{X} \in \mathbb{F}_n^{m \times p}$ - tensor

where $\mathbb{F}_n^p$ is a free-module over the ring $\mathbb{F}_n$

Tubal Operation

- Let $\mathbf{M} \in \mathbb{F}^{n \times n}$ be invertible. $\mathbb{F}_n$ denotes the ring $(\mathbb{F}^{1\times 1\times n}, +, \star_M)$.

$$\mathbf{a} \star_M \mathbf{b} = \mathbf{M}^{-1}(\mathbf{M}\ \mathbf{a} \odot \mathbf{M}\ \mathbf{b}) = \mathbf{M}^{-1}\,\mathrm{diag}(\mathbf{M}\ \mathbf{a})\mathbf{M}\ \mathbf{b}$$



- Tube fiber interpretation:

$$\begin{aligned} \mathbf{c} &= \texttt{fold}\left((\mathbf{M}^{-1}\mathrm{diag}(\mathbf{a})\mathbf{M})\texttt{vec}(\mathbf{b})\right) \\ &= \texttt{fold}\left((\mathbf{M}^{-1}\mathrm{diag}(\mathbf{b})\mathbf{M})\texttt{vec}(\mathbf{a})\right) \end{aligned}$$

- Commutativity, and characterization using set of diagonal matrices diagonalized by $\mathbf{M}$ and its inverse

---

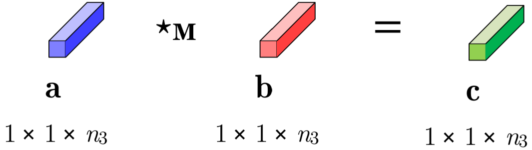
Kernfeld, Kilmer, Aeron, **Linear Algebra and its Applications**, NLA 2015

Avron, Mor, Demystifying Tubal Tensor Algebra. arXiv:2506.03311, 2025

Tubal Operation

- Let $\mathbf{M} \in \mathbb{F}^{n\times n}$ be invertible. $\mathbb{F}_n$ denotes the ring $(\mathbb{F}^{1\times 1\times n}, +, \star_M)$.

$$\mathbf{a} \star_M \mathbf{b} = \mathbf{M}^{-1}(\mathbf{M}\ \mathbf{a} \odot \mathbf{M}\ \mathbf{b}) = \mathbf{M}^{-1}\,\mathrm{diag}(\mathbf{M}\ \mathbf{a})\mathbf{M}\ \mathbf{b}$$



- $T : \mathbb{F}_n^p \to \mathbb{F}_n^m$ is a module homomorphism $(T(\mathbf{a} \star_M \mathbf{b}) = \mathbf{b} \star_M T(\mathbf{a}))$ if and only if $T\mathcal{B} = \mathcal{A} \star_M \mathcal{B}$ for some $\mathcal{A} \in \mathbb{F}_n^{m\times p}$

Kernfeld, Kilmer, Aeron, **Linear Algebra and its Applications**, NLA 2015

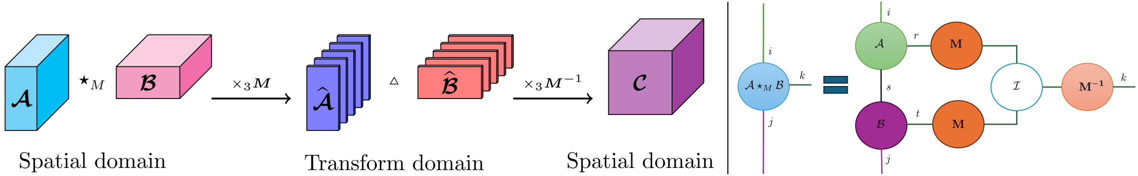
Avron, Mor, Demystifying Tubal Tensor Algebra. arXiv:2506.03311, 2025

The $\star_M$-Product Algebra

Given $\mathcal{A} \in \mathbb{R}^{\ell \times p \times n}$, $\mathcal{B} \in \mathbb{R}^{p \times m \times n}$, and an invertible $n \times n$ matrix M, then

$$\mathcal{C} = \mathcal{A} \star_M \mathcal{B} = \left(\hat{\mathcal{A}} \vartriangle \hat{\mathcal{B}}\right) \times_3 M^{-1}$$

where $\mathcal{C} \in \mathbb{R}^{\ell \times m \times n}$, $\hat{\mathcal{A}} = \mathcal{A} \times_3 M$, and $\vartriangle$ multiplies the frontal slices in parallel



Useful properties: tensor conjugate transpose, unitarity invariance, identity tensor, connection to Fourier transform, circulant shifts invariance, ...

$\star_M$ SVD and Truncation Optimality

Theorem (Kilmer, Horesh, Avron, Newman, 2021)

Let the t-SVD of $\mathcal{A} \in \mathbb{R}^{\ell \times m \times n}$ be given by $\mathcal{A} = \mathcal{U} \star_M \mathcal{S} \star_M \mathcal{V}^\top$, with $\ell \times \ell \times n$ orthogonal tensor $\mathcal{U}$, $m \times m \times n$ orthogonal tensor $\mathcal{V}$, and $\ell \times m \times n$ f-diagonal tensor $\mathcal{S}$

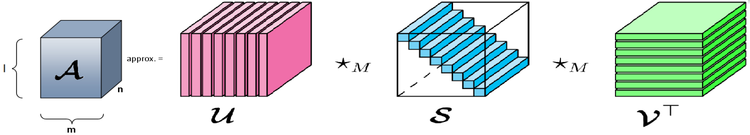
- *For $k < \min(l, m)$, define*

$$\mathcal{A}_k = \mathcal{U}(:, 1:k, :) \star_M \mathcal{S}(1:k, 1:k, :) \star_M \mathcal{V}^\top(:, 1:k, :) = \sum_{i=1}^{k} \mathcal{U}(:, i, :) \star_M \mathcal{S}(i, i, :) \star_M \mathcal{V}(:, i, :)^\top$$

- *Then*

$$\mathcal{A}_k = \operatorname*{argmin}_{\hat{\mathcal{A}} \in \mathcal{M}} \|\mathcal{A} - \hat{\mathcal{A}}\|$$

where $\mathcal{M} = \{\mathcal{C} = \mathcal{X} \star_M \mathcal{Y} \mid \mathcal{X} \in \mathbb{R}^{\ell \times k \times n}, \mathcal{Y} \in \mathbb{R}^{k \times m \times n}\}$



Kilmer, Horesh, Avron, Newman, **Tensor-tensor products for optimal representation and compression**, PNAS 2021

Tensor Representation Superiority

- Why might we expect the **tensor** representation to be more **informative** than matrix representation?

Kilmer, Horesh, Avron, Newman, **Tensor-tensor products for optimal representation and compression**, PNAS 2021

Tensor Representation Superiority

- Why might we expect the **tensor** representation to be more **informative** than matrix representation?

Theorem (Kilmer, Horesh, Avron, Newman, 2021)

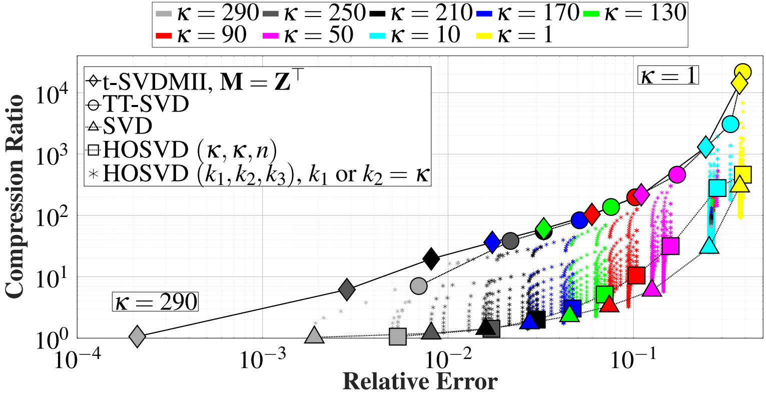
Let $X(:, i) = unfold(\mathcal{A}(:, i, :))$. *Let* $\mathcal{A}_k$ *denote the optimal* k*-term truncated* $\star_M$ *tensor-SVD approximation to* $\mathcal{A}$, *and let* A_k *denote the optimal* k*-term (i.e. rank-*k*) matrix SVD approximation to* A. *Then*

$$\|\mathcal{A} - \mathcal{A}_k\|_F \leq \|A - A_k\|_F$$

Kilmer, Horesh, Avron, Newman, **Tensor-tensor products for optimal representation and compression**, PNAS 2021

How it Compares to Other Tensorial Frameworks?

- **Theoretically**, similar superiority results are proved over HOSVD and TT-SVD
- Possible to **interpret** decompositions of the new tensor in **CP** form



De Lathauwer, De Moor, J. Vandewalle, **A multilinear singular value decomposition**, SIMAX 2000

Oseledets, **Tensor-train decomposition**, SISC 2011

Kilmer, Horesh, Avron, Newman, **Tensor-tensor products for optimal representation and compression**, PNAS 2021

Tensor-Tensor Applications - Proper Orthogonal Decomposition

Proper Orthogonal Decomposition (POD)

Motivation: Solving large-scale dynamical systems is computationally expensive

- Dynamic system: $\frac{\partial \bar{a}(t)}{\partial t} = L\bar{a}(t) + f(\bar{a}(t)) + \bar{q}(t)$
- For 2D grid $n_x \times n_y$: state size is $n_x n_y \times 1$, operator L is $n_x n_y \times n_x n_y$

Matrix-based POD:

- Collect snapshots $\{\bar{a}^1, \ldots, \bar{a}^{\mu_s}\}$ into matrix $A \in \mathbb{R}^{n_x n_y \times \mu_s}$
- Compute SVD $A = USV^\top$
- Use first k left singular vectors as projection basis U_k
- Galerkin projection: $U_k^\top U_k \frac{\partial \tilde{a}}{\partial t} = \underbrace{U_k^\top L U_k}_{k \times k} \tilde{a} + U_k^\top f(U_k \tilde{a}) + U_k^\top \bar{q}$

Tensor Proper Orthogonal Decomposition (t-POD)

Key Idea: Preserve spatial structure using tensors

- Form snapshot **tensor** $\mathcal{A} \in \mathbb{R}^{n_x \times \mu_s \times n_y}$ instead of matrix
- Compute k-term truncated t-SVD: $\mathcal{A}_k = \mathcal{U}_k \star_M \mathcal{S}_k \star_M \mathcal{V}_k^\top$
- Use $\mathcal{U}_k \in \mathbb{R}^{n_x \times k \times n_y}$ for projection

Advantages:

- t-POD basis better captures spatial structure of solutions
- Significantly lower relative error vs. matrix POD for same number of snapshots
- Orders of magnitude error reduction observed in diffusion equation experiments

Zhang, Kilmer, Horesh, Avron (2021); Zhang, Tufts PhD thesis, 2017

Tensor Proper Orthogonal Decomposition - Example

Diffusion Equation: $\frac{\partial a(\mathbf{r},t)}{\partial t} - \nabla \cdot \kappa \nabla a(\mathbf{r},t) = 0$

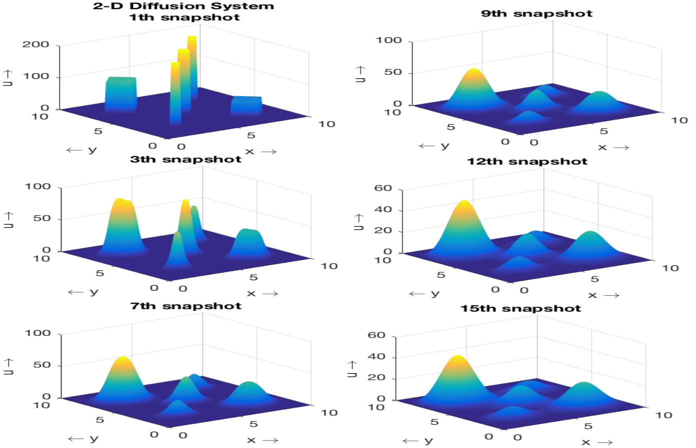


Figure: Sample snapshots of solution $\bar{\mathbf{a}}^j$, $j = 1$, 3, 7, 9, 12, 15

Better Basis? - Numerical Support

Diffusion Equation: $\frac{\partial a(\mathbf{r},t)}{\partial t} - \nabla \cdot \kappa \nabla a(\mathbf{r},t) = 0$

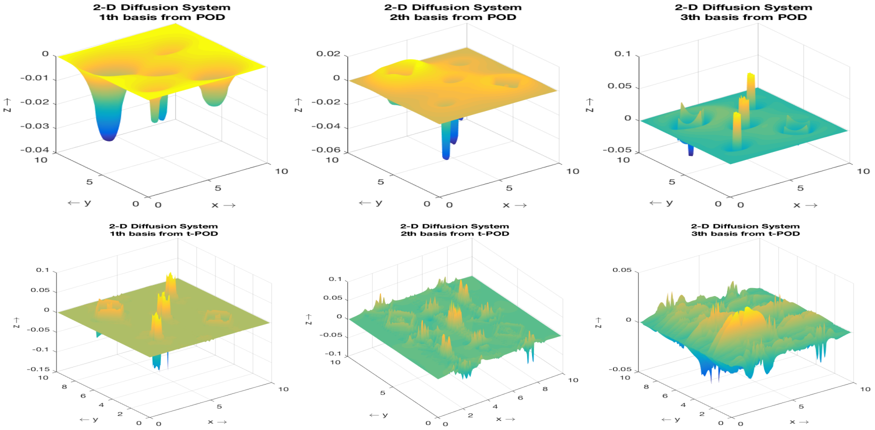
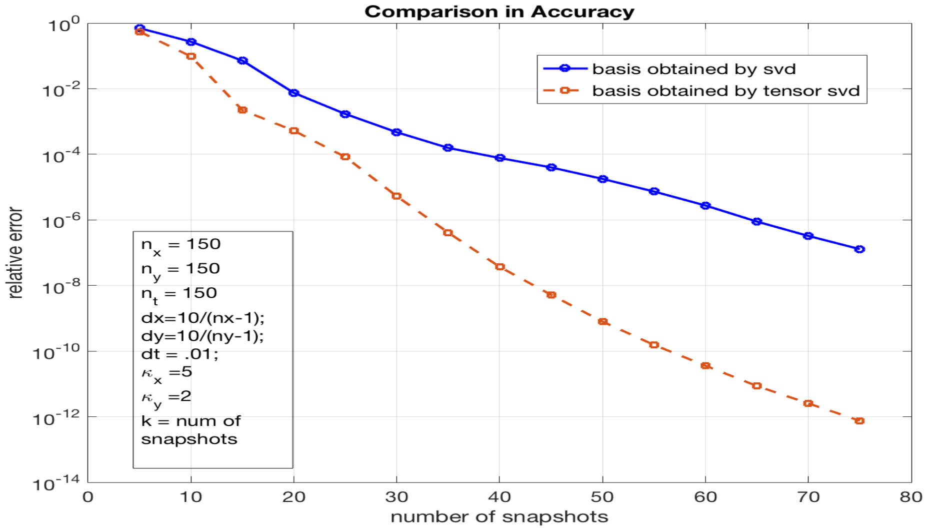


Figure: First 3 basis slices from **matrix POD** (top) and **t-POD** (bottom)

Tensor Proper Orthogonal Decomposition - Error vs. Snapshots Count



Tensor-Tensor Neural Networks

Tensor Neural Networks: Motivation

Key Idea: Replace matrix operations with tensor operations in neural networks

Standard Neural Network (Matrix):

$$\mathbf{a}_{j+1} = \sigma(W_j \cdot \mathbf{a}_j + \mathbf{b}_j) \quad \text{for } j = 0, \ldots, N-1$$

Tensor Neural Network:

$$\vec{\mathcal{A}}_{j+1} = \sigma(\mathcal{W}_j \star_M \vec{\mathcal{A}}_j + \vec{\mathcal{B}}_j)$$

Benefits of Tensor Formulation:

- **Reduced parameters:** $n^4 + n^2 \to n^3 + n^2$ for $n \times n$ images
- Preserves multi-dimensional structure of data
- Mimetic structure: tensors are M-linear operators (analogous to matrices)

Stable Architectures via Hamiltonian Framework

Dynamic Perspective: Network layers = discrete time steps

$$\frac{d}{dt}\mathbf{a}(t) = \sigma(W(t)\ \mathbf{a}(t) + \mathbf{b}(t)) \quad \text{for } t \in [0, T]$$

Well-Posed Learning requires stability conditions on eigenvalues of Jacobian

Hamiltonian-Inspired System: Antisymmetric $\Rightarrow$ inherently stable

$$\frac{d}{dt}\begin{bmatrix}\vec{\mathcal{A}}(t)\\ \vec{\mathcal{Z}}(t)\end{bmatrix} = \sigma\left(\begin{bmatrix}0 & \mathcal{W}(t)\\ -\mathcal{W}(t)^\top & 0\end{bmatrix} \star_M \begin{bmatrix}\vec{\mathcal{A}}(t)\\ \vec{\mathcal{Z}}(t)\end{bmatrix} + \begin{bmatrix}-\vec{\mathcal{B}}(t)\\ \vec{\mathcal{B}}(t)\end{bmatrix}\right)$$

Leapfrog Integration: Stable for purely imaginary eigenvalues

$$\vec{\mathcal{Z}}_{j+\frac{1}{2}} = \vec{\mathcal{Z}}_{j-\frac{1}{2}} - h\,\sigma(\mathcal{W}_j^\top \star_M \vec{\mathcal{A}}_j + \vec{\mathcal{B}}_j), \quad \vec{\mathcal{A}}_{j+1} = \vec{\mathcal{A}}_j + h\,\sigma(\mathcal{W}_j \star_M \vec{\mathcal{Z}}_{j+\frac{1}{2}} + \vec{\mathcal{B}}_j)$$

Tensor Neural Network Training

Forward Propagation: $\vec{\mathcal{A}}_{j+1} = \sigma(\mathcal{W}_j \star_M \vec{\mathcal{A}}_j + \vec{\mathcal{B}}_j)$

Objective Function:

$$E = \frac{1}{2}\|W_N \cdot \mathrm{unfold}(\vec{\mathcal{A}}_N) - \mathbf{c}\|_F^2$$

Backward Propagation:

$$\delta\vec{\mathcal{A}}_j = \mathcal{W}_j^\top \star_M (\delta\vec{\mathcal{A}}_{j+1} \odot \sigma'(\vec{\mathcal{Z}}_{j+1}))$$

Parameter Updates (Gradient Descent):

$$\delta\mathcal{W}_j = (\delta\vec{\mathcal{A}}_{j+1} \odot \sigma'(\vec{\mathcal{Z}}_{j+1})) \star_M \vec{\mathcal{A}}_j^\top$$
$$\delta\vec{\mathcal{B}}_j = \delta\vec{\mathcal{A}}_{j+1} \odot \sigma'(\vec{\mathcal{Z}}_{j+1})$$

where $\vec{\mathcal{Z}}_{j+1} = \mathcal{W}_j \star_M \vec{\mathcal{A}}_j + \vec{\mathcal{B}}_j$ and $\odot$ is pointwise product

Mimetic Structure

- **Update relations** are **analogous** to their matrix counterparts by **no coincidence**
- In the **M-product** framework, tensors are **M-linear** operators just as **matrices** are **linear** operators

Experimental Results

MNIST (28×28 grayscale, 60K train / 10K test)

- Tensor networks achieve **comparable accuracy** with **28**$\times$ **fewer parameters**
- Parameters: Matrix $28^4 N$ vs. Tensor $28^3 N$

CIFAR-10 ($32 \times 32 \times 3$ RGB, 50K train / 10K test)

- Tensor networks **outperform** matrix networks at same depth
- Better accuracy with **32**$\times$ **fewer parameters**

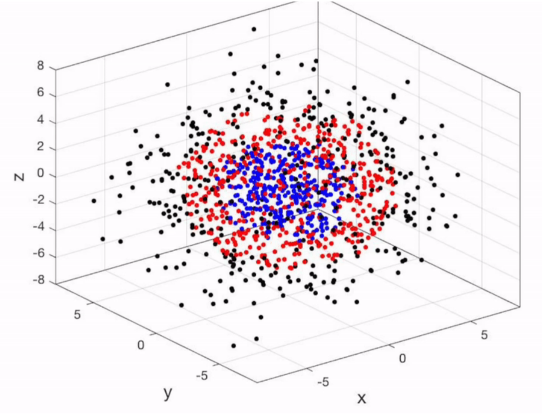
Key Findings:

- Hamiltonian + Leapfrog: stable even with large step sizes ($h = 1$)
- Standard ResNet: unstable, requires small h (≤ 0.25)
- Tensor formulation preserves spatial structure $\Rightarrow$ better generalization

Newman, Horesh, Avron, Kilmer, **Stable tensor neural networks for rapid deep learning**, Frontiers in Big Data, 2024

Stability in Motion

Data: 1200 train, concentric spheres (417 black, 466 red, 317 blue)
Parameters: $\alpha = 0.01$, $\sigma = tanh$, 50 epochs, batch size = 10, $N = 32$
Regularized: $\frac{1}{2h}||\boldsymbol{w}_{j+1} - \boldsymbol{w}_j||_F^2$ for $1 \times 1 \times 3$ tubes $\boldsymbol{w}_j$

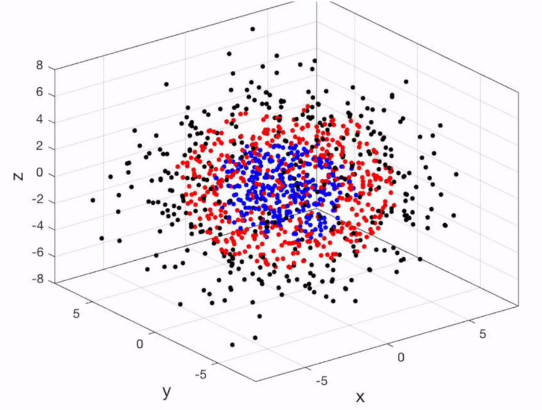


Resnet with $h = 0.5$

$$\boldsymbol{a}_{j+1} = \boldsymbol{a}_j + h\,\sigma(\boldsymbol{w}_j * \boldsymbol{a}_j + \boldsymbol{b}_j)$$

Stability in Motion

Data: 1200 train, concentric spheres (417 black, 466 red, 317 blue)
Parameters: $\alpha = 0.01$, $\sigma = tanh$, 50 epochs, batch size = 10, $N = 32$
Regularized: $\frac{1}{2h}||\boldsymbol{w}_{j+1} - \boldsymbol{w}_j||_F^2$ for $1 \times 1 \times 3$ tubes $\boldsymbol{w}_j$

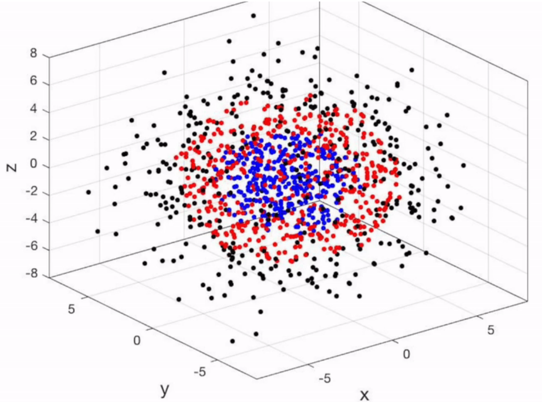


Resnet with $h = 0.25$

$$\boldsymbol{a}_{j+1} = \boldsymbol{a}_j + h\,\sigma(\boldsymbol{w}_j * \boldsymbol{a}_j + \boldsymbol{b}_j)$$

Stability in Motion

Data: 1200 train, concentric spheres (417 black, 466 red, 317 blue)
Parameters: $\alpha = 0.01$, $\sigma = tanh$, 50 epochs, batch size = 10, $N = 32$
Regularized: $\frac{1}{2h}||\boldsymbol{w}_{j+1} - \boldsymbol{w}_j||_F^2$ for $1 \times 1 \times 3$ tubes $\boldsymbol{w}_j$



Leapfrog with $h = 1$

$$\begin{cases} \boldsymbol{z}_{j+\frac{1}{2}} = \boldsymbol{z}_{j-\frac{1}{2}} - h\,\sigma(\boldsymbol{w}_j^\top * \boldsymbol{a}_j + \boldsymbol{b}_j) \\ \boldsymbol{a}_{j+1} = \boldsymbol{a}_j + h\,\sigma(\boldsymbol{w}_j * \boldsymbol{z}_{j+\frac{1}{2}} + \boldsymbol{b}_j) \end{cases}$$

Tensor-Tensor Graph Convolutional Neural Networks

TensorGCN: Motivation and Method

Motivation

- Graphs – popular data structures to effectively represent interactions
- Real world applications involve time evolving graphs; learning representations of dynamic graphs essential

Proposed Approach

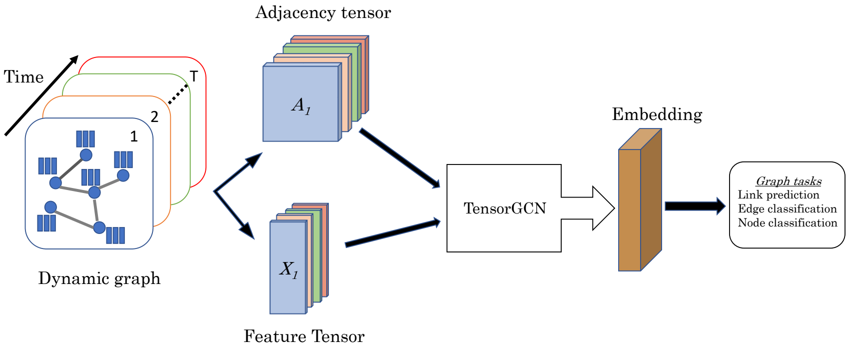
- Novel *tensor variant* of graph convolutional network (GCN)
- Captures correlation over time via tensor M-product framework

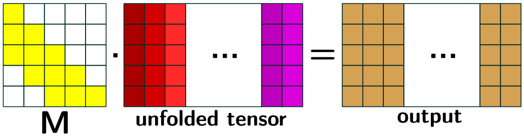
Two-Layer TensorGCN Model:

$$\mathcal{Z} = \text{softmax}\left(\mathcal{L} \star_M \sigma\left(\mathcal{L} \star_M \mathcal{X} \star_M \mathcal{W}^{(0)}\right) \star_M \mathcal{W}^{(1)}\right)$$

Graph Tasks: Link prediction, Edge classification, Node classification

TM-GCN


Adjacency tensor
Time
T
2
1
Dynamic graph
A_1
X_1
Feature Tensor
TensorGCN
Embedding
Graph tasks
Link prediction
Edge classification
Node classification


...
...
M
unfolded tensor
output

TensorGCN: Experimental Results (Edge Classification)

Table 1: Results without symmetrizing adjacency matrices (higher is better)

Method	Bitcoin OTC	Bitcoin Alpha	Reddit	Chess
WD-GCN	0.2062	0.1920	**0.2337**	0.4311
EvolveGCN	0.3284	0.1609	0.2012	0.4351
GCN	0.3317	0.2100	0.1805	0.4342
TensorGCN (Ours)	**0.3529**	**0.2331**	0.2028	**0.4708**

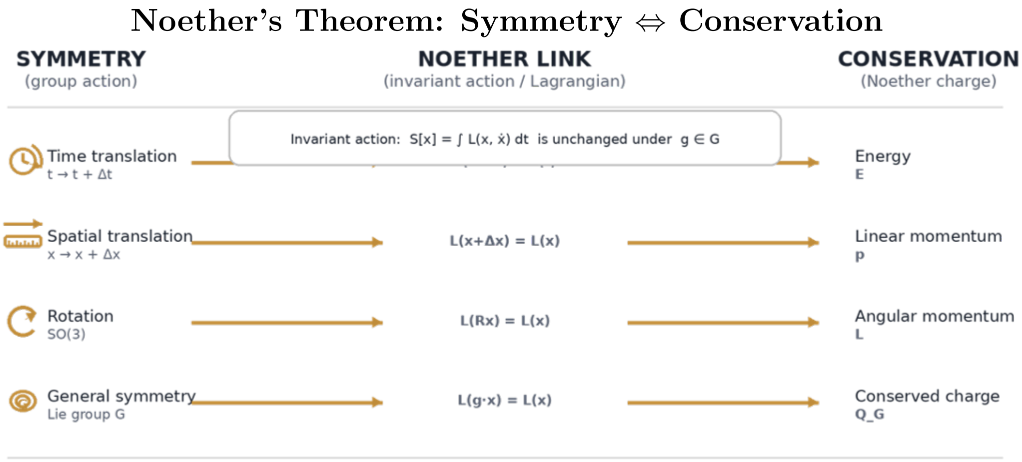
Table 2: Results with symmetrized adjacency matrices (higher is better)

Method	Bitcoin OTC	Bitcoin Alpha	Reddit	Chess
WD-GCN	0.1009	0.1319	**0.2173**	0.4321
EvolveGCN	0.0913	**0.2273**	0.1942	0.4091
GCN	0.0769	0.1538	0.1966	0.4369
TensorGCN (Ours)	**0.3103**	0.2207	0.2071	**0.4713**

Malik, Ubaru, Horesh, Kilmer, Avron, **Dynamic graph convolutional networks using the tensor M-Product**, SIAM SDM, 2021

Optimal Symmetry-Aware Compression of Multiway Data

The Universal Importance of Symmetry



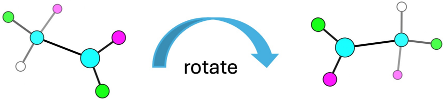
Implication for ML

Models that respect symmetry inherit conservation properties automatically

Symmetries in Data

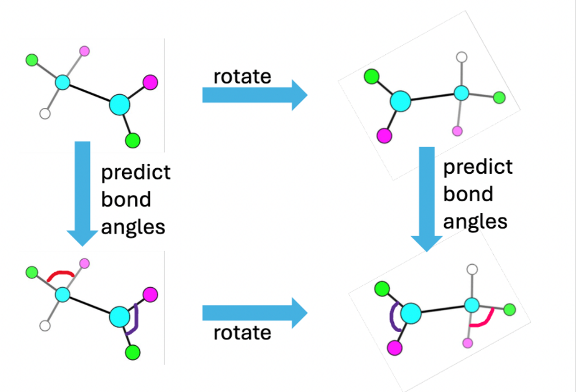
- Real-world data exhibit fundamental symmetries:
  - **Graphs:** Permutation symmetry
  - **Molecules:** 3D rotational symmetry
  - **Images/Videos:** Translational symmetry
  - **Time series:** Temporal invariance
  - **Crystals/Fingerprints:** Reflection symmetry
  - **Financial/Medical data:** Scale invariance

- **Groups** mathematically model data symmetries: A set closed under an associative operation, with identity and inverse elements



Equivariant Models

- **Equivariant models** respect data symmetries
- **Definition:** A function f is equivariant to group G if:
$$f(g \cdot x) = g \cdot f(x) \quad \forall g \in G$$
- If we transform the input, the output transforms *consistently*



Translating Physical Symmetries into Machine Learning Language

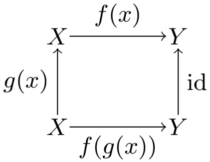
We express symmetry in models through two primary properties: Invariance and Equivariance

Invariance (The prediction doesn't change)

A function f is **invariant** to a transformation g if the output remains identical:

$$f(g(x)) = f(x)$$

Example: Classifying a 3D object. The label ("airplane") should not change if the object is rotated

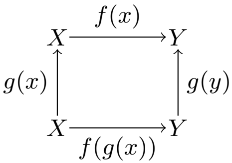


Equivariance (The prediction transforms with the input)

A function f is **equivariant** if transforming the input and then applying the function is the same as applying the function and then transforming the output:

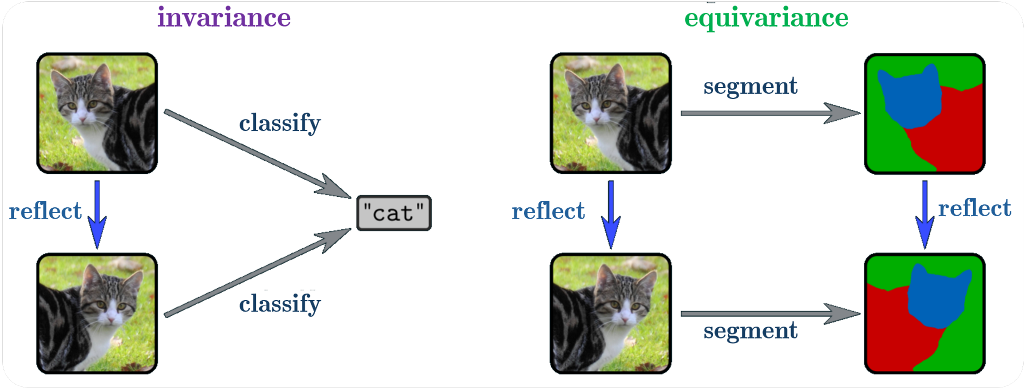
$$f(g(x)) = g(f(x))$$

Example: Predicting a force field on a molecule. If the molecule rotates, the force vectors must rotate with it



Translating Physical Symmetries into Machine Learning Language

We express symmetry in models through two primary properties: Invariance and Equivariance



Credits: Maurice Weiler

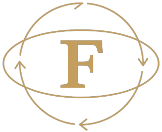
The Brute-Force Approach vs. Principled Design

How can a model learn to respect symmetries?

Learning by Rote (Data Augmentation)

Present the model with the same data in many different transformed states (e.g., rotated, translated)

- **Computationally Expensive**: Requires massive datasets and long training times.
- **Incomplete**: Can never cover the full continuum of transformations (e.g., all possible rotations).
- **Not Robust**: Only achieves approximate invariance/equivariance

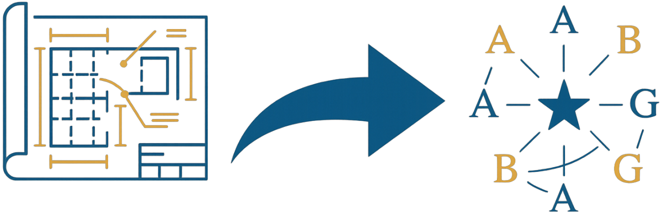


Generalizing by Design (Equivariant Architecture)

Constrain the model's hypothesis space to functions that are *guaranteed* to be equivariant

- The model does not need to *learn* the symmetry; it is an inherent property of its operations
- This 'bakes in' the equivariance, leading to automatic generalization across all transformations within the group

A Paradigm Shift: From Architectural Constraint to Intrinsic Property



The Old Question: "How can we design a *network architecture* that is equivariant to group G?"

$\Longrightarrow$

The New Question: "How can we define an *algebra* where group symmetry is an inherent property of multiplication itself?"

ENNs: Symmetry by Design

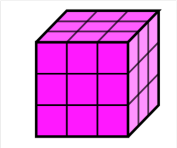
$\star_G$ **Algebra: Symmetry by Definition**

Two Paths to Symmetry: A Comparison

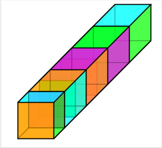
Feature	**Equivariant Neural Networks (ENNs)**	$\star_G$ **Algebra**
Symmetry Handling	An **architectural constraint**, *enforced* layer by layer	An **algebraic property**, *intrinsic* to the features
Flexibility	Architectures are often **group-specific**	**Universal**. Any finite group G can define an algebra
Composition	**Complex**; may require framework redesign	**Seamless** via direct product groups $(G_1 \times G_2 \times \cdots \times G_m)$
Underlying Math	**Constrained function approximation**	**Group representation theory**
Analogy	Building a house from a specific blueprint	Writing sentences in a universal language

Structure-Aware Tensor Compression

- We look for tensor approximation that preserves
  1. Symmetries in data
  2. Number of modes of the original tensor
- **Why symmetries matter:** Enable simpler models, better sample efficiency, and improved generalization
- **Why preserve modes:** Maintain higher-order correlations lost in the matricization / vectorization process



order 3 tensor



Higher-order tensors

Definition: Convolution Tensor

Definition (Convolution Tensor)

Given a finite group G of order n, its corresponding **convolution tensor** $\mathcal{T} \in \mathbb{R}^{n \times n \times n}$ is defined by

$$\mathcal{T}(a, b, c) = \begin{cases} 1 & \text{if } ab = c \\ 0 & \text{otherwise} \end{cases}$$

for all $a, b, c \in G$

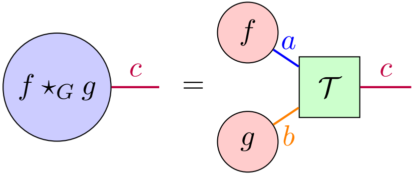
Key Property: For each $a \in G$, $\mathcal{T}(a, :, :)^\top = \rho(a)$
The horizontal slices of $\mathcal{T}$ give the regular representation of group G

Convolution as Tensor Contraction

The convolution of functions $f, g\colon G \to \mathbb{R}$ satisfies

$$(f \star_G g)(c) = \sum_{a\in G}\sum_{b\in G} f(a)g(b)\mathcal{T}(a,b,c)$$

for all $c \in G$

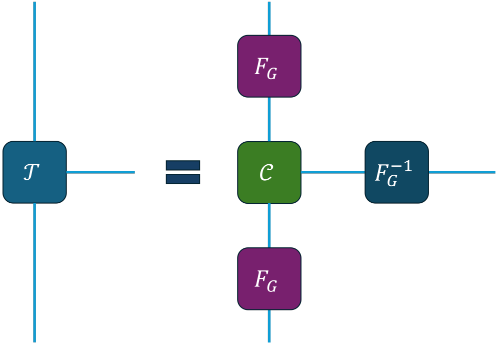


Proof idea: The condition $ab = c$ is equivalent to $b = a^{-1}c$, so $\mathcal{T}(a,b,c) = \delta_{b,a^{-1}c}$

- Reduces the double sum to the standard convolution definition
- The convolution tensor $\mathcal{T}$ acts as a **multiplication table** in tensor form, enabling efficient computation of group convolutions through tensor contractions

Decomposing the Engine with the Peter-Weyl Theorem

The Peter-Weyl theorem allows us to decompose the convolution tensor $\mathcal{T}$ into its fundamental algebraic components:



- **Factor Matrices (F_G):** Generalized Fourier Transforms that map data from the group domain into a new 'symmetry basis'. For any finite group, $F(a,:) = \text{rvec}(\rho(a))$
- **Core Tensor ($\mathcal{C}$):** The **Peter-Weyl Tensor**: The algebraic "heart" of the group convolution tensor

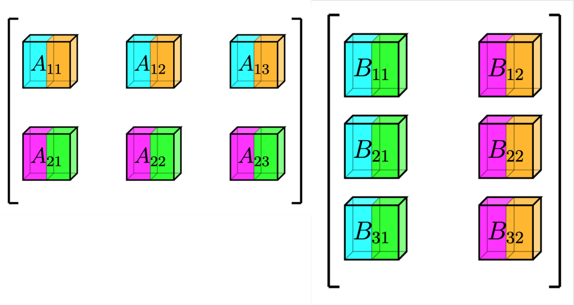
New Equivariant Tensor Algebra

- View order p tensors as matrices whose entries are order $d = p - 2$ tensors with entry-wise addition and multiplication given by convolution along each mode:

$$(\mathcal{A}_{ik} \star_G \mathcal{B}_{kj})(c_1, \ldots, c_d) = \sum_{(a_1 \ldots, a_d) \in G} \mathcal{A}_{ik}(a_1, \ldots, a_d) \mathcal{B}_{kj}(a_1^{-1} c_1, \ldots, a_d^{-1} c_d),$$

- ***Equivariant*** product of two of these matrices is defined by:

$$(\mathcal{A} \star_G \mathcal{B})_{ij} = \sum_{k=1}^{m} \mathcal{A}_{ik} \star_G \mathcal{B}_{kj}$$



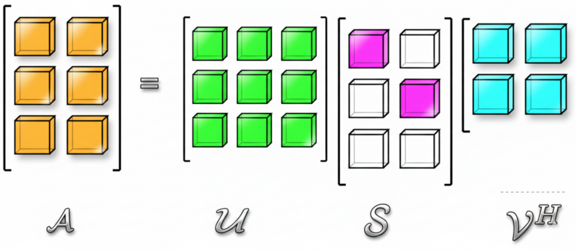
Useful properties: Notions of conjugate transpose, identity tensor, unitary tensor.
Connection to Fourier analysis

Tensor $\star_G$-SVD

- The SVD decomposition of a tensor $\mathcal{A}$ is

$$\mathcal{A} = \mathcal{U} \star_G \mathcal{S} \star_G \mathcal{V}^H$$
$$= \sum_{i=1}^{r} \mathcal{U}(:, i, :) \star_G \mathcal{S}(i, i, :) \star_G \mathcal{V}^H(:, i, :)$$

with $\mathcal{U}, \mathcal{V}$ unitary and $\mathcal{S}$ f-diagonal.



- The t-rank k approximation is

$$\mathcal{A}_k = \mathcal{U}(:, 1:k, :) \star_G \mathcal{S}(1:k, 1:k, :) \star_G \mathcal{V}^H(:, 1:k, :)$$

Optimal Compression Guarantee

- Eckart-Young-like equivariant result for tensors:

Theorem (Hoyos, Ubaru, Huh, Clarkson, Kilmer, Horesh (2025))

Given a finite group G, let $\mathcal{A} = \mathcal{U} \star_G \mathcal{S} \star_G \mathcal{V}^H$ be an SVD decomposition for the tensor $\mathcal{A} \in \mathbb{K}_G^{\ell \times m}$. Define $\mathcal{A}_k = \mathcal{U}(:,1:k,:) \star_G \mathcal{S}(1:k,1:k,:) \star_G \mathcal{V}^H(:,1:k,:)$. Then $\mathcal{A}_k$ is the best t-rank k approximation of the tensor $\mathcal{A}$. The squared error is

$$\|\mathcal{A} - \mathcal{A}_k\|_F^2 = \sum_{i=k+1}^{r} \|\mathbf{s}_i\|_F^2,$$

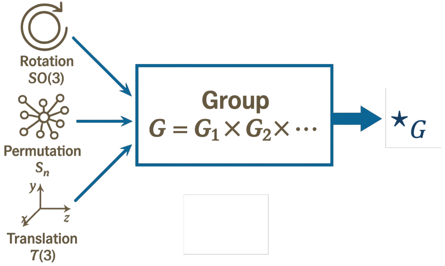
where r is the t-rank of $\mathcal{A}$.

- For any t-rank k tensor $\mathcal{B}$, we have that

$$\|\mathcal{A} - \mathcal{A}_k\|_F^2 \leq \|\mathcal{A} - \mathcal{B}\|_F^2$$

Benefit 1: Seamless Composition of Symmetries

- The $\star_G$ algebra naturally handles **direct product groups**: $G = G_1 \times G_2 \times \ldots \times G_d$
- Allows the creation of a **single algebraic system** that is simultaneously equivariant to **multiple**, distinct symmetry groups



Example: An algebra defined by the group $G = SO(3) \times S_n$ can natively handle data that is both rotatable ($SO(3)$) and contains permutable parts (S_n). This is achieved without re-engineering network layers

Rotation $SO(3)$

Permutation S_n

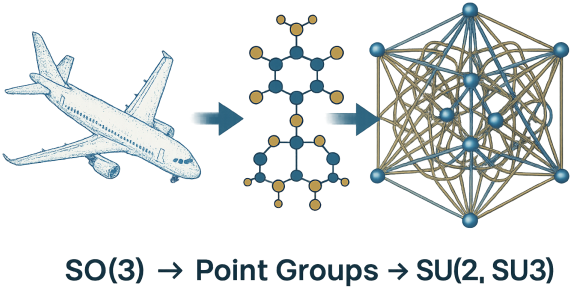
Translation $T(3)$

$\Downarrow$

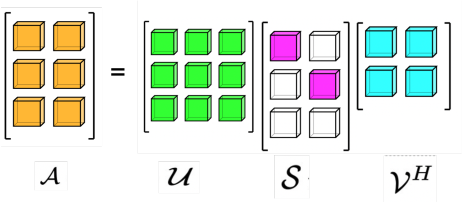
Group
$G = G_1 \times G_2 \times \ldots$

Benefit 2: The Native Language of High-Order Symmetries

- Built upon **group representation theory**, the mathematical language of quantum mechanics and field theory
- Can **natively** represent symmetries beyond simple geometric transformations, such as the $SU(N)$ gauge symmetries crucial in Lattice QCD
- Provides a **principled**, first-principles way to incorporate deep **physical/geometrical priors** directly into machine learning models



Benefit 3: Optimal Equivariant Compression



- The $\star_G$ algebra offers a Singular Value Decomposition, and an Eckart-Young-like theorem holds:

$$A = U \star_G S \star_G V^H$$

where U and V are $\star_G$-unitary

- **The truncated $\star_G$-SVD provides the provably best low-rank approximation** to a tensor within the space of G-equivariant structures
- This is not possible with standard SVD or Tucker decompositions, which are blind to symmetry

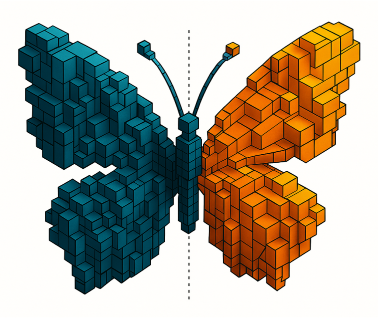
Summary

$\star_M$ **Algebra:**

- Elementary units are *tubes*, not scalars
- Matrix-mimetic with optimal representations
- Preserves operator/data structure duality
- Scalable algorithms with seamless retrofit

$\star_G$ **Algebra:**

- Elementary units are $d = p - 2$ tensors
- Extends to general group symmetries
- Reveals hidden equivariants in high-dimensional data

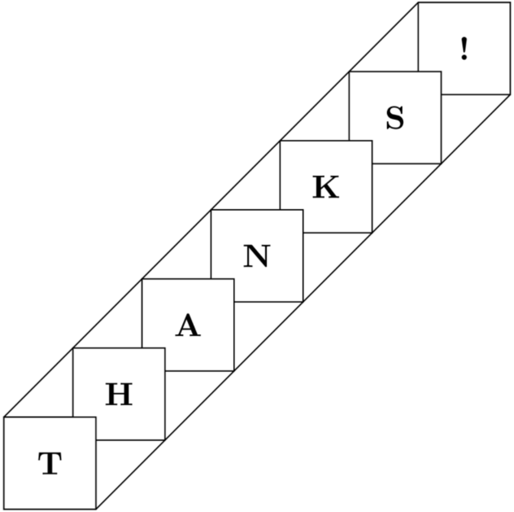


Take Away Message

Instead of forcing our world into a simple math,
we can adapt our math to the shape of the world

References

- D. Huh, **Discovering group structures via unitary representation learning**, The International Conference on Learning Representations (ICLR) 2025
- L. Newman, L. Horesh, H. Avron, M. Kilmer, **Stable tensor neural networks for rapid deep learning**, Frontiers in Big Data, 2024
- O. Malik, S. Ubaru, L. Horesh, M. Kilmer, H. Avron, **Dynamic graph convolutional networks using the tensor M-product**, SIAM International Conference on Data Mining, 2021
- S. Ubaru, L. Horesh, G. Cohen, **Dynamic graph and polynomial chaos based models for contact tracing data analysis and optimal testing prescription**, Journal of Biomedical Informatics, 2021
- M. Kilmer, L. Horesh, H. Avron, L. Newman, **Tensor-tensor products for optimal representation and compression**, PNAS 2021
- V. Chakaravarthy, S. Pandian, S. Raje, Y. Sabharwal, T. Suzumura, S. Ubaru, **Efficient scaling of dynamic graph neural networks**, Supercomputing, 2021
- TM-GCN Pytorch code - `https://github.com/IBM/TM-GCN`
- Covid transmission Matlab simulation : `https://github.com/Shashankaubaru/GraphSEIR_aPCE`


T
H
A
N
K
S
!