Guaranteed Learning of Latent Variable Models through Tensor Methods

Furong Huang

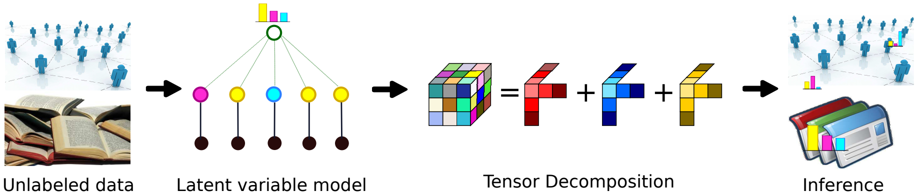
University of Maryland

furongh@cs.umd.edu

ACM SIGMETRICS Tutorial 2018

Tutorial Topic

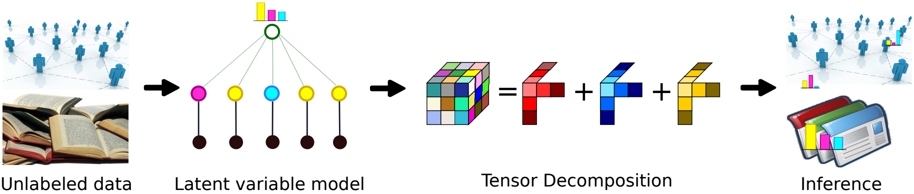
Learning algorithms
for **latent variable models**
based on **decompositions of moment tensors**.



"**Method-of-moments**" (Pearson, 1894)

Tutorial Topic

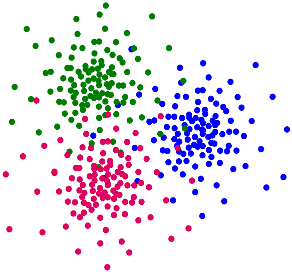
Learning algorithms (**parameter estimation**)
for **latent variable models**
based on **decompositions of moment tensors**.



"**Method-of-moments**" (Pearson, 1894)

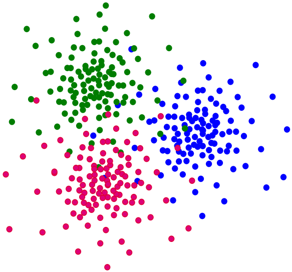
Application 1: Clustering

- Basic operation of grouping data points.
- Hypothesis: each data point belongs to an unknown group.



Application 1: Clustering

- Basic operation of grouping data points.
- Hypothesis: each data point belongs to an unknown group.



Probabilistic/latent variable viewpoint

- The groups represent different distributions. (e.g. Gaussian).
- Each data point is drawn from one of the given distributions. (e.g. Gaussian mixtures).

Application 2: Topic Modeling

Bleary-Eyed Syrian Troops Fight a Building at a Time

The New York Times

The News-Review

MONDAY, NOVEMBER 12, 2012

Callers to 211 seek help with food, rent

211: Douglas County trails others in info line usage

Corrections

Hippo eats dwarf

Document modeling

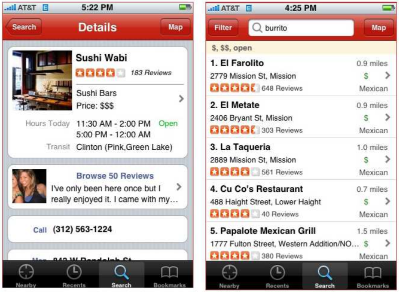
- Observed: words in document corpus.
- Hidden: topics.
- Goal: carry out document summarization.

Application 3: Understanding Human Communities

Social Networks

- Observed: network of social ties, e.g. friendships, co-authorships
- Hidden: groups/communities of social actors.

Application 4: Recommender Systems



Recommender System

- Observed: Ratings of users for various products, e.g. yelp reviews.
- Goal: Predict new recommendations.
- Modeling: Find groups/communities of users and products.

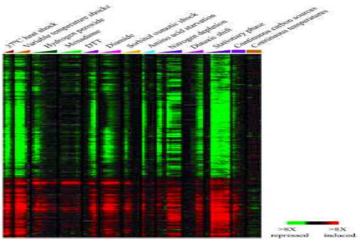
Application 5: Feature Learning

Label	Features			
0	2.1	5.2	0	0
1	0	0	2	1
1	1.1	0	0	0
0	0	0	7	0

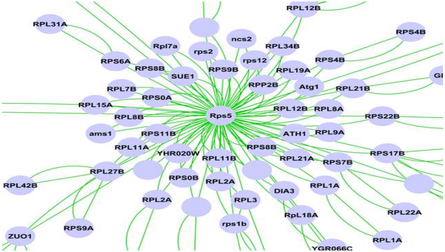
Feature Engineering

- Learn good features/representations for classification tasks, e.g. image and speech recognition.
- Sparse representations, low dimensional hidden structures.

Application 6: Computational Biology



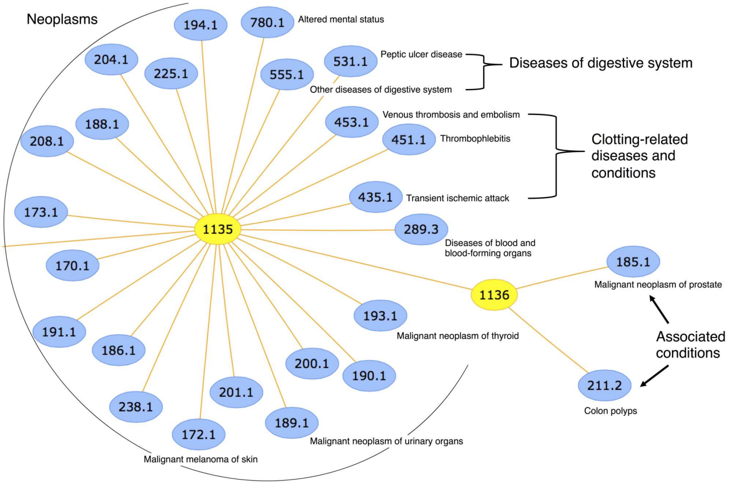
Gasch *et al.* *Mol Biol Cell* 2000.



- Observed: gene expression levels
- Goal: discover gene groups
- Hidden variables: regulators controlling gene groups

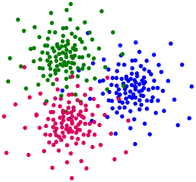
Application 7: Human Disease Hierarchy Discovery

CMS: 1.6 million patients, 168 million diagnostic events, 11 k diseases.



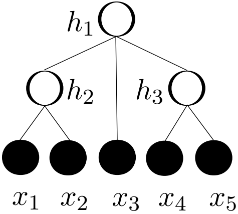
" Scalable Latent TreeModel and its Application to Health Analytics " by F. Huang, N. U.Niranjan, I. Perros, R. Chen, J. Sun, A. Anandkumar, NIPS 2015 MLHC workshop.

How to model hidden effects?



Basic Approach: mixtures/clusters

- Hidden variable h is categorical.



Advanced: Probabilistic models

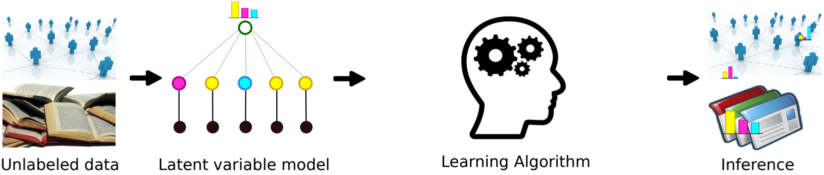
- Hidden variable h has more general distributions.
- Can model mixed memberships.

This talk: basic mixture model and some advanced models.

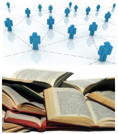
Challenges in Learning

Basic goal in all mentioned applications

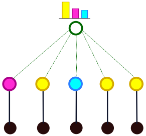
Discover hidden structure in data: unsupervised learning.



Challenges in Learning – find hidden structure in data

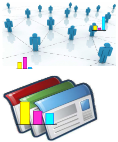


Unlabeled data



Latent variable model

Learning Algorithm

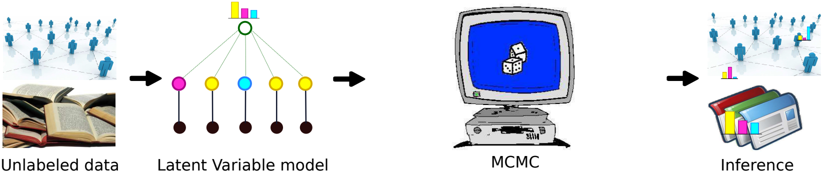


Inference

Challenge: Conditions for Identifiability

- Whether can model be identified given infinite computation and data?
- Are there tractable algorithms under identifiability?

Challenges in Learning – find hidden structure in data



Challenge: Conditions for Identifiability

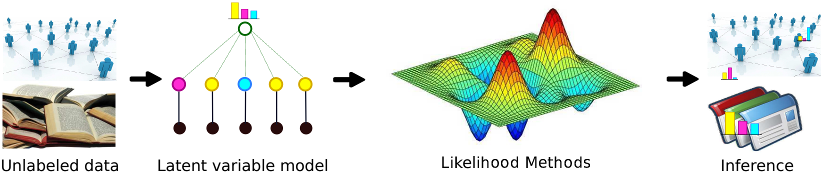
- Whether can model be identified given infinite computation and data?
- Are there tractable algorithms under identifiability?

Challenge: Efficient Learning of Latent Variable Models

- MCMC: random sampling, slow

  Exponential mixing time

Challenges in Learning – find hidden structure in data



Challenge: Conditions for Identifiability

- Whether can model be identified given infinite computation and data?
- Are there tractable algorithms under identifiability?

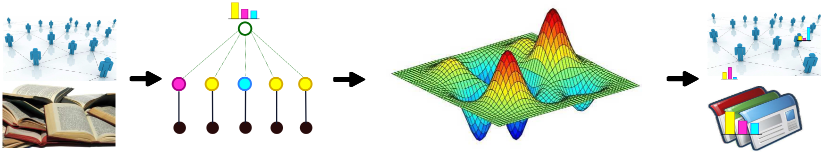
Challenge: Efficient Learning of Latent Variable Models

- MCMC: random sampling, slow

  Exponential mixing time
- Likelihood: non-convex, not scalable

  Exponential critical points

Challenges in Learning – find hidden structure in data



Unlabeled data — Latent variable model — Likelihood Methods — Inference

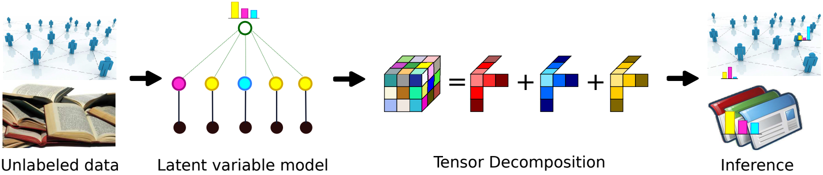
Challenge: Conditions for Identifiability

- Whether can model be identified given infinite computation and data?
- Are there tractable algorithms under identifiability?

Challenge: Efficient Learning of Latent Variable Models

- MCMC: random sampling, slow
  Exponential mixing time
- Likelihood: non-convex, not scalable
  Exponential critical points
- Efficient computational and sample complexities?

Challenges in Learning – find hidden structure in data



Challenge: Conditions for Identifiability

- Whether can model be identified given infinite computation and data?
- Are there tractable algorithms under identifiability?

Challenge: Efficient Learning of Latent Variable Models

- MCMC: random sampling, slow

  Exponential mixing time
- Likelihood: non-convex, not scalable

  Exponential critical points
- Efficient computational and sample complexities?

Guaranteed and efficient learning through spectral methods

What this tutorial will cover

Outline

1. Introduction
2. Motivation: Challenges of MLE for Gaussian Mixtures

What this tutorial will cover

Outline

1. Introduction
2. Motivation: Challenges of MLE for Gaussian Mixtures
3. Introduction of Method of Moments and Tensor Notations

What this tutorial will cover

Outline

1. Introduction
2. Motivation: Challenges of MLE for Gaussian Mixtures
3. Introduction of Method of Moments and Tensor Notations
4. Topic Model for Single-topic Documents

What this tutorial will cover

Outline

1. Introduction
2. Motivation: Challenges of MLE for Gaussian Mixtures
3. Introduction of Method of Moments and Tensor Notations
4. Topic Model for Single-topic Documents
    - Identifiability
    - Parameter recovery via decomposition of exact moments

What this tutorial will cover

Outline

1. Introduction
2. Motivation: Challenges of MLE for Gaussian Mixtures
3. Introduction of Method of Moments and Tensor Notations
4. Topic Model for Single-topic Documents
    - Identifiability
    - Parameter recovery via decomposition of exact moments
5. Error-tolerant Algorithms for Tensor Decompositions

What this tutorial will cover

Outline

1. Introduction
2. Motivation: Challenges of MLE for Gaussian Mixtures
3. Introduction of Method of Moments and Tensor Notations
4. Topic Model for Single-topic Documents
   - Identifiability
   - Parameter recovery via decomposition of exact moments
5. Error-tolerant Algorithms for Tensor Decompositions
   - Decomposition for tensors with linearly independent components
   - Decomposition for tensors with orthogonal components

What this tutorial will cover

Outline

1. Introduction
2. Motivation: Challenges of MLE for Gaussian Mixtures
3. Introduction of Method of Moments and Tensor Notations
4. Topic Model for Single-topic Documents
    - Identifiability
    - Parameter recovery via decomposition of exact moments
5. Error-tolerant Algorithms for Tensor Decompositions
    - Decomposition for tensors with linearly independent components
    - Decomposition for tensors with orthogonal components
6. Tensor Decomposition for Neural Network Compression
7. Conclusion

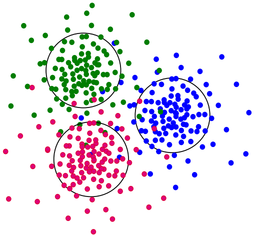
Outline

Gaussian Mixture Model

Generative Model

- Samples are comprised of K different Gaussians according to $\mathsf{Cat}(\pi_1, \pi_2, \ldots, \pi_K)$
- Each sample is from one of the K Gaussians, $\mathcal{N}(\boldsymbol{\mu}_h, \boldsymbol{\Sigma}_h)$, $\forall h \in [K]$

$$
\begin{aligned}
H &\sim \mathsf{Cat}(\pi_1, \pi_2, \ldots, \pi_K) \\
\boldsymbol{X}|_{H=h} &\sim \mathcal{N}(\boldsymbol{\mu}_h, \boldsymbol{\Sigma}_h), \quad \forall h \in [K]
\end{aligned}
$$

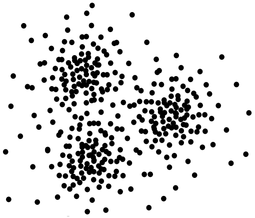


Gaussian Mixture Model

Generative Model

- Samples are comprised of K different Gaussians according to $\mathsf{Cat}(\pi_1, \pi_2, \ldots, \pi_K)$
- Each sample is from one of the K Gaussians, $\mathcal{N}(\boldsymbol{\mu}_h, \boldsymbol{\Sigma}_h)$, $\forall h \in [K]$

$$H \sim \mathsf{Cat}(\pi_1, \pi_2, \ldots, \pi_K)$$
$$\boldsymbol{X}|_{H=h} \sim \mathcal{N}(\boldsymbol{\mu}_h, \boldsymbol{\Sigma}_h), \quad \forall h \in [K]$$



Learning Problem

Estimate **mean vector** $\boldsymbol{\mu}_h$, **covariance matrix** $\boldsymbol{\Sigma}_h$, and **mixing weight** $\mathsf{Cat}(\pi_1, \pi_2, \ldots, \pi_K)$ of each subpopulation from unlabeled data.

Maximum Likelihood Estimator (MLE)

- Data $\{\boldsymbol{x}_i\}_{i=1}^n$
- Likelihood $\Pr_{\boldsymbol{\theta}}(\text{data}) \overset{\text{iid}}{=} \prod_{i=1}^n \Pr_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$
- Model parameter estimation $\widehat{\boldsymbol{\theta}}_{\text{mle}} := \operatorname*{argmax}_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \log \Pr_{\boldsymbol{\theta}}(\text{data})$
- Latent variable models: some variables are hidden
  - No "direct" estimators when some variables are hidden
  - Local optimization via Expectation-Maximization (EM) (Dempster, Laird, & Rubin, 1977)

MLE for Gaussian Mixture Models

Given data $\{\boldsymbol{x}_i\}_{i=1}^n$ and the number of Gaussian components K, the model parameters to be estimated are $\boldsymbol{\theta} = \{(\boldsymbol{\mu}_h, \boldsymbol{\Sigma}_h, \pi_h)\}_{h=1}^K$.

$\widehat{\boldsymbol{\theta}}_{\text{mle}}$ for Gaussian Mixture Models

$$\widehat{\boldsymbol{\theta}}_{\text{mle}} := \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \sum_{i=1}^n \log\left(\sum_{h=1}^K \frac{\pi_h}{\det(\boldsymbol{\Sigma}_h)^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{\mu}_h)^\top \boldsymbol{\Sigma}_h^{-1}(\boldsymbol{x}_i - \boldsymbol{\mu}_h)\right)\right)$$

- Solving MLE estimator is NP-hard (Dasgupta, 2008; Aloise, Deshpande, Hansen, & Popat, 2009; Mahajan, Nimbhorkar, & Varadarajan, 2009; Vattani, 2009; Awasthi, Charikar, Krishnaswamy, & Sinop, 2015).

Consistent Estimator

Definition

Suppose iid samples $\{\boldsymbol{x}_i\}_{i=1}^n$ are generated by distribution $\Pr_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$ where the model parameters $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ are unknown. An estimator $\widehat{\boldsymbol{\theta}}$ is consistent if

$$\mathbb{E}\|\widehat{\boldsymbol{\theta}} - \boldsymbol{\theta}\| \to 0 \quad \text{as} \quad n \to \infty$$

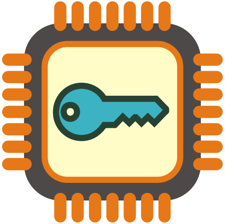
Spherical Gaussian Mixtures $\Sigma_h = I$ (as $n \to \infty$)

- For $K = 2$ and $\pi_h = 1/2$: EM is consistent (Xu, H., & Maleki, 2016; Daskalakis, Tzamos, & Zampetakis, 2016).
- Larger K: easily trapped in local maxima, far from global max (Jin, Zhang, Balakrishnan, Wainwright, & Jordan, 2016).
- Practitioners often use EM with many (random) restarts, but may take a long time to get near the global max.

Hardness of Parameter Estimation

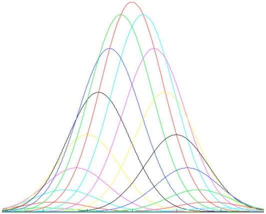
Exponentially difficult computationally or statistically
to learn model parameters, even under the parametric setting.

Cryptographic hardness



E.g., Mossel & Roch, 2006

Information-theoretic hardness



E.g., Moitra & Valiant, 2010

May require $2^{\Omega(K)}$ running time or $2^{\Omega(K)}$ sample size.

Ways Around the Hardness

- Separation conditions.

  E.g., assume $\min_{i \neq j} \frac{\|\boldsymbol{\mu}_i - \boldsymbol{\mu}_j\|^2}{\sigma_i^2 + \sigma_j^2}$ is sufficiently large.

  (Dasgupta, 1999; Arora & Kannan, 2001; Vempala & Wang, 2002; . . .)

- Structural assumptions.

  E.g., assume sparsity, separable (anchor words).

  (Spielman, Wang & Wright, 2012; Arora, Ge & Moitra, 2012; . . .)

- Non-degeneracy conditions.

  E.g., assume $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K$ span a K-dimensional space.

This tutorial: statistically and computationally efficient learning algorithms for non-degenerate instances via method-of-moments.

Outline

Method-of-Moments At A Glance

1. Determine function of model parameters $\boldsymbol{\theta}$ estimatable from observable data:
   - **Moments**

$$\mathbb{E}_{\boldsymbol{\theta}}[f(\boldsymbol{X})]$$

2. Form estimates of moments using data (iid samples $\{\boldsymbol{x}_i\}_{i=1}^n$):
   - **Empirical Moments**

$$\widehat{\mathbb{E}}[f(\boldsymbol{X})]$$

3. Solve the approximate equations for parameters $\boldsymbol{\theta}$:
   - **Moment matching**

$$\mathbb{E}_{\boldsymbol{\theta}}[f(\boldsymbol{X})] \overset{n\to\infty}{=} \widehat{\mathbb{E}}[f(\boldsymbol{X})]$$

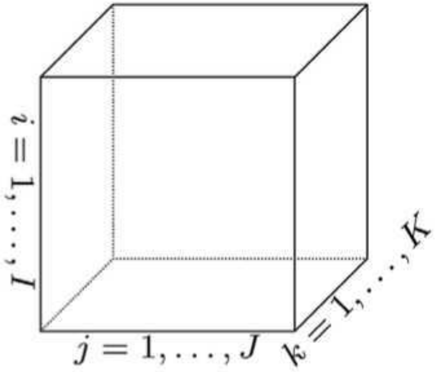
Toy Example

How to estimate Gaussian variable, i.e., $(\boldsymbol{\mu},\boldsymbol{\Sigma})$,
given iid samples $\{\boldsymbol{x}_i\}_{i=1}^n \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}^2)$?

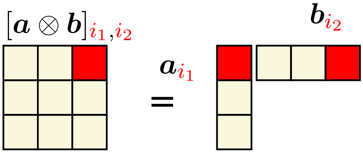
What is a tensor?

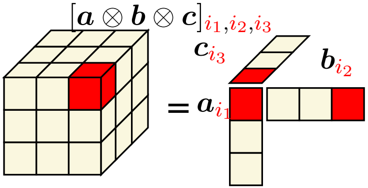
Multi-dimensional Array

- Tensor - Higher order matrix
- The number of dimensions is called tensor order.



Tensor Product

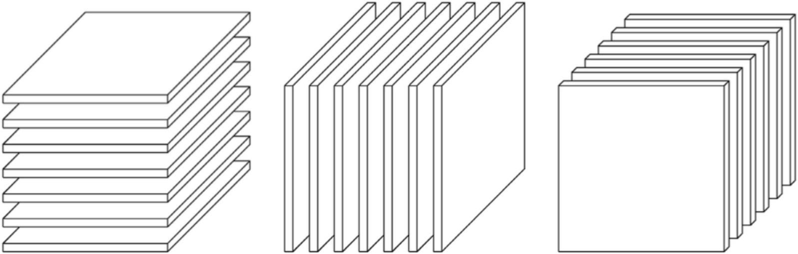




- $[\boldsymbol{a} \otimes \boldsymbol{b}]_{i_1,i_2} = \boldsymbol{a}_{i_1} \boldsymbol{b}_{i_2}$
- Rank-1 matrix

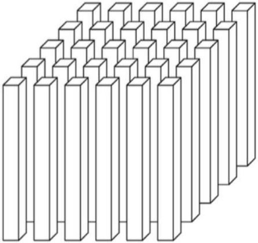
- $[\boldsymbol{a} \otimes \boldsymbol{b} \otimes \boldsymbol{c}]_{i_1,i_2,i_3} = \boldsymbol{a}_{i_1} \boldsymbol{b}_{i_2} \boldsymbol{c}_{i_3}$
- Rank-1 tensor

Slices

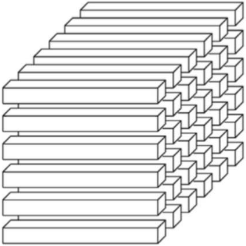


- Horizontal slices
- Lateral slices
- Frontal slices

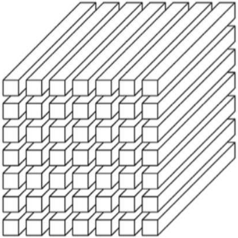
Fiber



- Mode-1 (column) fibers

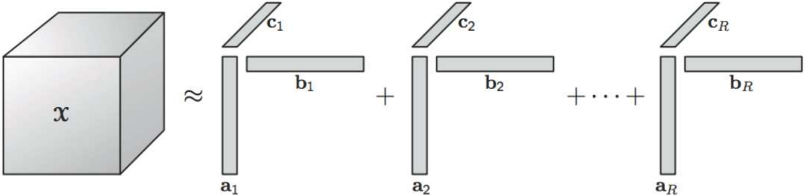


- Mode-2 (row) fibers



- Mode-3 (tube) fibers

CP decomposition



- $\boldsymbol{\mathcal{X}} = \sum_{h=1}^{R} \boldsymbol{a}_h \otimes \boldsymbol{b}_h \otimes \boldsymbol{c}_h$
- Rank: Minimum number of rank-1 tensors whose sum generates the tensor.

Multi-linear Transform

Multi-linear Operation

If $\boldsymbol{\mathcal{T}} = \sum_{h=1}^{R} \boldsymbol{a}_h \otimes \boldsymbol{b}_h \otimes \boldsymbol{c}_h$, a multi-linear operation using matrices $(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z})$ is as follows

$$\boldsymbol{\mathcal{T}}(\boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{Z}) := \sum_{h=1}^{K} (\boldsymbol{X}^\top \boldsymbol{a}_h) \otimes (\boldsymbol{Y}^\top \boldsymbol{b}_h) \otimes (\boldsymbol{Z}^\top \boldsymbol{c}_h).$$

Similarly for a multi-linear operation using vectors $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$

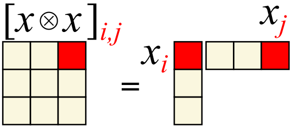
$$\boldsymbol{\mathcal{T}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) := \sum_{h=1}^{K} (\boldsymbol{x}^\top \boldsymbol{a}_h) \otimes (\boldsymbol{y}^\top \boldsymbol{b}_h) \otimes (\boldsymbol{z}^\top \boldsymbol{c}_h).$$

Tensors in Method of Moments

Matrix: Pair-wise relationship

- Signal or data observed $\boldsymbol{x} \in \mathbb{R}^d$
- Rank 1 matrix: $[\boldsymbol{x} \otimes \boldsymbol{x}]_{i,j} = \boldsymbol{x}_i \boldsymbol{x}_j$
- Aggregated pair-wise relationship

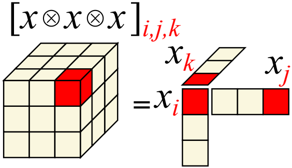
$$\boldsymbol{M}_2 = \mathbb{E}[\boldsymbol{x} \otimes \boldsymbol{x}]$$



Tensor: Triple-wise relationship or higher

- Signal or data observed $\boldsymbol{x} \in \mathbb{R}^d$
- Rank 1 tensor:
  $[\boldsymbol{x} \otimes \boldsymbol{x} \otimes \boldsymbol{x}]_{i,j,k} = \boldsymbol{x}_i \boldsymbol{x}_j \boldsymbol{x}_k$
- Aggregated triple-wise relationship

$$\boldsymbol{\mathcal{M}}_3 = \mathbb{E}[\boldsymbol{x} \otimes \boldsymbol{x} \otimes \boldsymbol{x}] = \mathbb{E}[\boldsymbol{x}\otimes^3]$$

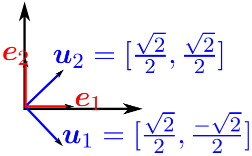


Why are tensors powerful?

Matrix Orthogonal Decomposition

- Not unique without eigenvalue gap

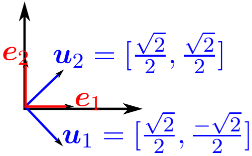
$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \boldsymbol{e}_1\boldsymbol{e}_1^\top + \boldsymbol{e}_2\boldsymbol{e}_2^\top = \boldsymbol{u}_1\boldsymbol{u}_1^\top + \boldsymbol{u}_2\boldsymbol{u}_2^\top$$



Why are tensors powerful?

Matrix Orthogonal Decomposition

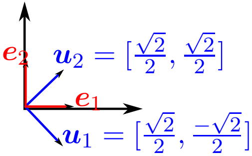
- Not unique without eigenvalue gap
$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \boldsymbol{e}_1\boldsymbol{e}_1^\top + \boldsymbol{e}_2\boldsymbol{e}_2^\top = \boldsymbol{u}_1\boldsymbol{u}_1^\top + \boldsymbol{u}_2\boldsymbol{u}_2^\top$$
- Unique with eigenvalue gap



Why are tensors powerful?

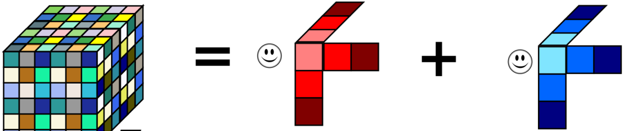
Matrix Orthogonal Decomposition

- Not unique without eigenvalue gap
  $$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = e_1 e_1^\top + e_2 e_2^\top = u_1 u_1^\top + u_2 u_2^\top$$
- Unique with eigenvalue gap



Tensor Orthogonal Decomposition (Harshman, 1970)

- Unique: eigenvalue gap not needed



$$Tensor = u_1 \otimes u_1 \otimes u_1 + u_2 \otimes u_2 \otimes u_2$$

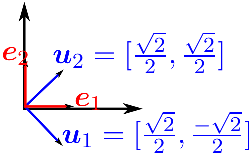
Why are tensors powerful?

Matrix Orthogonal Decomposition

- Not unique without eigenvalue gap

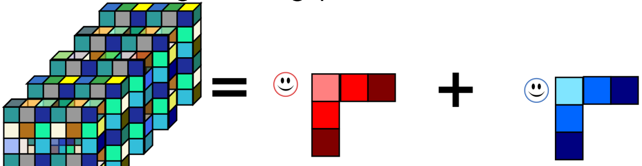
$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \boldsymbol{e}_1\boldsymbol{e}_1^\top + \boldsymbol{e}_2\boldsymbol{e}_2^\top = \boldsymbol{u}_1\boldsymbol{u}_1^\top + \boldsymbol{u}_2\boldsymbol{u}_2^\top$$

- Unique with eigenvalue gap



Tensor Orthogonal Decomposition (Harshman, 1970)

- Unique: eigenvalue gap not needed
- Slice of tensor has eigenvalue gap



$$Slice_i = u_1(i)\, u_1 \otimes u_1 + u_2(i)\, u_2 \otimes u_2$$

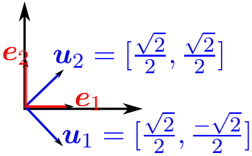
Why are tensors powerful?

Matrix Orthogonal Decomposition

- Not unique without eigenvalue gap

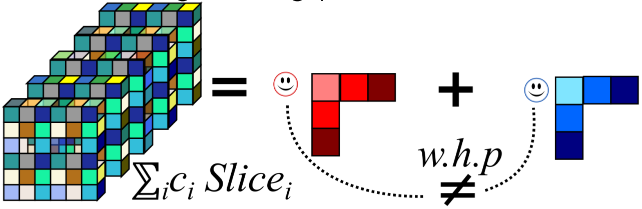
$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = e_1 e_1^\top + e_2 e_2^\top = u_1 u_1^\top + u_2 u_2^\top$$

- Unique with eigenvalue gap



Tensor Orthogonal Decomposition (Harshman, 1970)

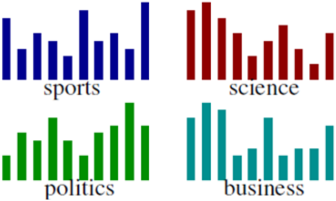
- Unique: eigenvalue gap not needed
- Slice of tensor has eigenvalue gap



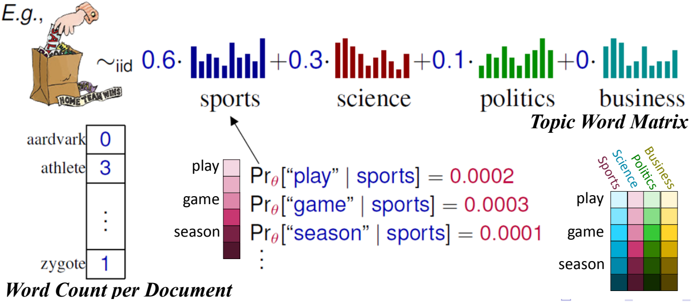
Outline

Topic Modeling

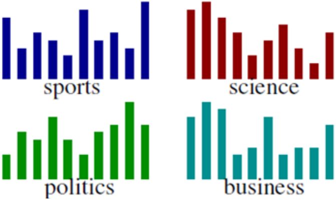
General Topic Model (e.g., Latent Dirichlet Allocation)



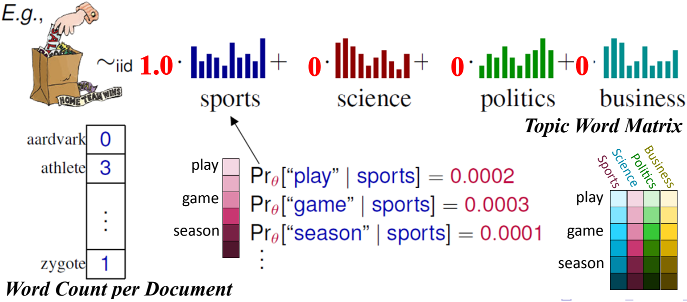
- K topics
  - each associated with a **distribution over vocab words** $\{a_h\}_{h=1}^{K}$
- **Hidden** topic proportion $\boldsymbol{w}$
  - per document i, $\boldsymbol{w}^{(i)} \in \Delta^{K-1}$
- Document $\overset{\text{iid}}{\sim}$ mixture of topics



Topic Model for Single-topic Documents



- K topics
  - each associated with a **distribution over vocab words** $\{a_h\}_{h=1}^K$
- **Hidden** topic proportion w
  - per document i, $w^{(i)} \in \{e_1, \ldots, e_K\}$
- Document $\overset{\text{iid}}{\sim} a_h$



Model Parameters of Topic Model for Single-topic Documents

Estimate Topic Proportion

- Topic proportion $\boldsymbol{w} = [w_1, \ldots, w_K]$

  $\boxed{w_h = \mathbb{P}[\text{topic of word} = h]}$

Estimate Topic Word Matrix

- Topic-word matrix $\boldsymbol{A} = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_K]$

  $\boxed{\boldsymbol{A}_{jh} = \mathbb{P}[\text{word} = \boldsymbol{e}_j | \text{topic} = h]}$

- **Goal**: to estimate model parameters $\{(\boldsymbol{a}_h, w_h)\}_{h=1}^{K}$, given iid samples of n documents (word count $\{\boldsymbol{c}^{(i)}\}_{i=1}^{n}$)
- Frequency vector $\boldsymbol{x}^{(i)} = \frac{\boldsymbol{c}^{(i)}}{L}$, the length of document is $L = \sum_j \boldsymbol{c}_j^{(i)}$

Moment Matching

Nondegenerate model (linearly independent topic-word matrix)

- Generative process:

  Choose $h \sim \mathsf{Cat}(w_1, \ldots, w_K)$
  Generate L words $\sim \boldsymbol{a}_h$

- $\mathbb{E}[\boldsymbol{x}] = \sum_{h=1}^{K} \mathbb{P}[\text{topic} = h]\mathbb{E}[\boldsymbol{x}|\text{topic} = h]$
- $\mathbb{E}[\boldsymbol{x}|\text{topic} = h] = \boxed{\sum_j \mathbb{P}[\text{word} = \boldsymbol{e}_j|\text{topic} = h]\boldsymbol{e}_j} = \boldsymbol{a}_h$

Moment Matching

Nondegenerate model (linearly independent topic-word matrix)

- Generative process:

  Choose $h \sim \mathsf{Cat}(w_1, \ldots, w_K)$
  Generate L words $\sim \boldsymbol{a}_h$

- $\mathbb{E}[\boldsymbol{x}] = \sum_{h=1}^{K} \mathbb{P}[\text{topic} = h]\mathbb{E}[\boldsymbol{x}|\text{topic} = h] \;= \sum_{h=1}^{K} w_h \boldsymbol{a}_h$
- $\mathbb{E}[\boldsymbol{x}|\text{topic} = h] = \boxed{\sum_j \mathbb{P}[\text{word} = \boldsymbol{e}_j|\text{topic} = h]\boldsymbol{e}_j} = \boldsymbol{a}_h$

Identifiability: how long must the documents be?

Nondegenerate model (linearly independent topic-word matrix)

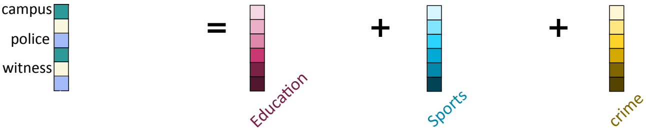
- Generative process:

  Choose $h \sim \mathrm{Cat}(w_1, \ldots, w_K)$
  Generate L words $\sim \boldsymbol{a}_h$

- $\mathbb{E}[\boldsymbol{x}] = \sum_{h=1}^{K} \mathbb{P}[\text{topic} = h]\mathbb{E}[\boldsymbol{x}|\text{topic} = h] \; = \sum_{h=1}^{K} w_h \boldsymbol{a}_h$
- $\mathbb{E}[\boldsymbol{x}|\text{topic} = h] = \boxed{\sum_j \mathbb{P}[\text{word} = \boldsymbol{e}_j|\text{topic} = h]\boldsymbol{e}_j} = \boldsymbol{a}_h$

$\boldsymbol{M}_1$: Distribution of words ($\widehat{\boldsymbol{M}}_1$: Occurrence frequency of words)

$$\boldsymbol{M}_1 = \mathbb{E}[\boldsymbol{x}] = \sum_h w_h \boldsymbol{a}_h; \quad \widehat{\boldsymbol{M}}_1 = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}^{(i)}$$



Identifiability: how long must the documents be?

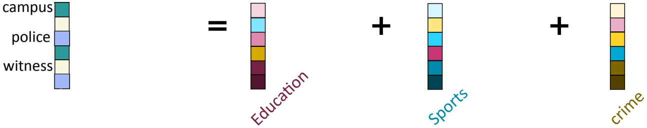
Nondegenerate model (linearly independent topic-word matrix)

- Generative process:

  Choose $h \sim \mathsf{Cat}(w_1, \ldots, w_K)$
  Generate L words $\sim \boldsymbol{a}_h$
- $\mathbb{E}[\boldsymbol{x}] = \sum_{h=1}^{K} \mathbb{P}[\text{topic} = h]\mathbb{E}[\boldsymbol{x}|\text{topic} = h] \;= \sum_{h=1}^{K} w_h \boldsymbol{a}_h$
- $\mathbb{E}[\boldsymbol{x}|\text{topic} = h] = \boxed{\sum_j \mathbb{P}[\text{word} = \boldsymbol{e}_j|\text{topic} = h]\boldsymbol{e}_j} = \boldsymbol{a}_h$

$\boldsymbol{M}_1$: Distribution of words ($\widehat{\boldsymbol{M}}_1$: Occurrence frequency of words)

$$\boldsymbol{M}_1 = \mathbb{E}[\boldsymbol{x}] = \sum_h w_h \boldsymbol{a}_h; \quad \widehat{\boldsymbol{M}}_1 = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}^{(i)}$$



No unique decomposition of vectors

Identifiability: how long must the documents be?

Nondegenerate model (linearly independent topic-word matrix)

- Generative process:

  Choose $h \sim \text{Cat}(w_1, \dots, w_K)$
  Generate L words $\sim \boldsymbol{a}_h$

- $\mathbb{E}[\boldsymbol{x}] = \sum_{h=1}^{K} \mathbb{P}[\text{topic} = h]\mathbb{E}[\boldsymbol{x}|\text{topic} = h] \;\; = \sum_{h=1}^{K} w_h \boldsymbol{a}_h$
- $\mathbb{E}[\boldsymbol{x}|\text{topic} = h] = \boxed{\sum_j \mathbb{P}[\text{word} = \boldsymbol{e}_j|\text{topic} = h]\boldsymbol{e}_j} = \boldsymbol{a}_h$

$\boldsymbol{M}_2$: Distribution of word pairs ($\widehat{\boldsymbol{M}}_2$: Co-occurrence of word pairs)

$$\boldsymbol{M}_2 = \mathbb{E}[\boldsymbol{x} \otimes \boldsymbol{x}] = \sum_h w_h \boldsymbol{a}_h \otimes \boldsymbol{a}_h; \quad \widehat{\boldsymbol{M}}_2 = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}^{(i)} \otimes \boldsymbol{x}^{(i)}$$

Identifiability: how long must the documents be?

Nondegenerate model (linearly independent topic-word matrix)

- Generative process:

  Choose $h \sim \mathrm{Cat}(w_1, \ldots, w_K)$
  Generate L words $\sim \boldsymbol{a}_h$

- $\mathbb{E}[\boldsymbol{x}] = \sum_{h=1}^{K} \mathbb{P}[\text{topic} = h]\mathbb{E}[\boldsymbol{x}|\text{topic} = h] \;\; = \sum_{h=1}^{K} w_h \boldsymbol{a}_h$
- $\mathbb{E}[\boldsymbol{x}|\text{topic} = h] = \boxed{\sum_j \mathbb{P}[\text{word} = \boldsymbol{e}_j|\text{topic} = h]\boldsymbol{e}_j} = \boldsymbol{a}_h$

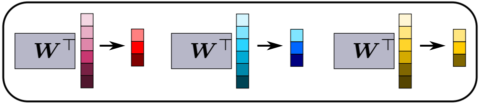
$\boldsymbol{M}_2$: Distribution of word pairs ($\widehat{\boldsymbol{M}}_2$: Co-occurrence of word pairs)

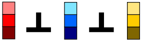
$$\boldsymbol{M}_2 = \mathbb{E}[\boldsymbol{x} \otimes \boldsymbol{x}] = \sum_h w_h \boldsymbol{a}_h \otimes \boldsymbol{a}_h; \quad \widehat{\boldsymbol{M}}_2 = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}^{(i)} \otimes \boldsymbol{x}^{(i)}$$

Matrix decomposition recovers subspace, not actual model

Identifiability: how long must the documents be?

Nondegenerate model (linearly independent topic-word matrix)

Find a W 

such that 

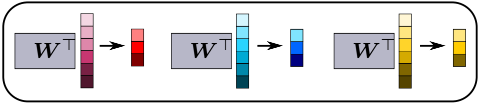
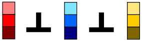
M_2: Distribution of word pairs ($\widehat{M}_2$: Co-occurrence of word pairs)

$$M_2 = \mathbb{E}[x \otimes x] = \sum_h w_h a_h \otimes a_h; \quad \widehat{M}_2 = \frac{1}{n} \sum_{i=1}^{n} x^{(i)} \otimes x^{(i)}$$

Many such W's, find one such that $v_h = W^\top a_h$ orthogonal

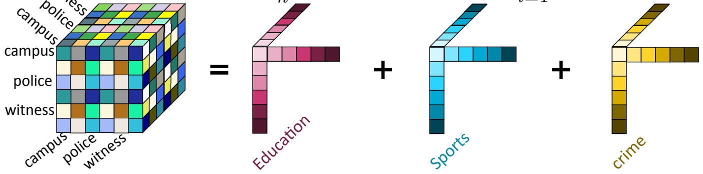
Identifiability: how long must the documents be?

Nondegenerate model (linearly independent topic-word matrix)

Know a $\boldsymbol{W}$  such that 

$\mathcal{M}_3$: Distribution of word triples ($\widehat{\mathcal{M}}_3$: Co-occurrence of word triples)

$$\mathcal{M}_3 = \mathbb{E}[\boldsymbol{x}\otimes^3] = \sum_h w_h \boldsymbol{a}_h \otimes^3; \quad \widehat{\mathcal{M}}_3 = \frac{1}{n}\sum_{i=1}^{n} \boldsymbol{x}^{(i)}\otimes^3$$

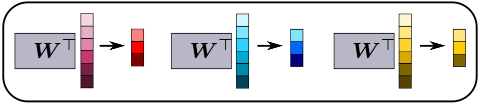


Orthogonalize the tensor, project data with $\boldsymbol{W}$: $\mathcal{M}_3(\boldsymbol{W}, \boldsymbol{W}, \boldsymbol{W})$

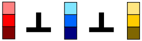
Identifiability: how long must the documents be?

Nondegenerate model (linearly independent topic-word matrix)

Know a $\boldsymbol{W}$

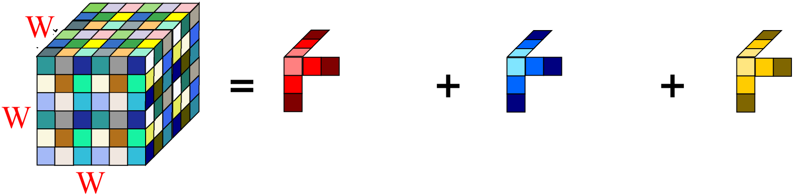


such that



$\boldsymbol{\mathcal{M}}_3$: Distribution of word triples ($\widehat{\boldsymbol{\mathcal{M}}}_3$: Co-occurrence of word triples)

$$\boldsymbol{\mathcal{M}}_3(\boldsymbol{W},\boldsymbol{W},\boldsymbol{W}) = \mathbb{E}[(\boldsymbol{W}^\top \boldsymbol{x})^{\otimes 3}] = \sum_h w_h (\boldsymbol{W}^\top \boldsymbol{a}_h)^{\otimes 3}; \ \widehat{\boldsymbol{\mathcal{M}}}_3(\boldsymbol{W},\boldsymbol{W},\boldsymbol{W}) = \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{W}^\top \boldsymbol{x}^{(i)})^{\otimes 3}$$

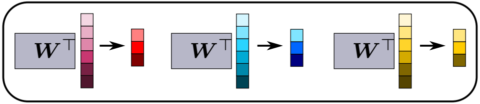


Unique orthogonal tensor decomposition $\{\widehat{\boldsymbol{v}}_h\}_{h=1}^{K}$

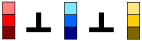
Identifiability: how long must the documents be?

Nondegenerate model (linearly independent topic-word matrix)

Know a $\boldsymbol{W}$

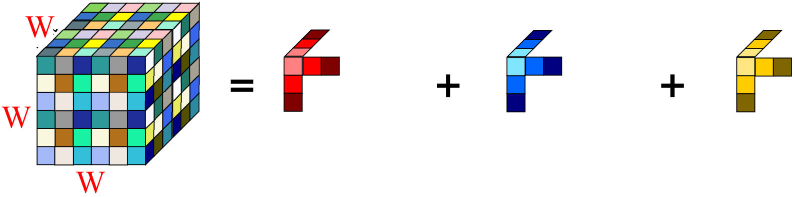


such that



$\mathcal{M}_3$: Distribution of word triples ($\widehat{\mathcal{M}}_3$: Co-occurrence of word triples)

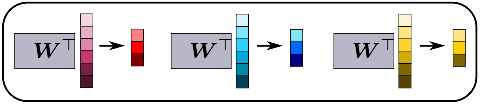
$$\mathcal{M}_3(\boldsymbol{W},\boldsymbol{W},\boldsymbol{W}) = \mathbb{E}[(\boldsymbol{W}^\top \boldsymbol{x})^{\otimes 3}] = \sum_h w_h (\boldsymbol{W}^\top \boldsymbol{a}_h)^{\otimes 3};\ \widehat{\mathcal{M}}_3(\boldsymbol{W},\boldsymbol{W},\boldsymbol{W}) = \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{W}^\top \boldsymbol{x}^{(i)})^{\otimes 3}$$

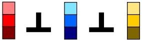


$$\boxed{\text{Model parameter estimation: } \widehat{\boldsymbol{a}}_h = (W^\top)^\dagger \widehat{\boldsymbol{v}}_h}$$

Identifiability: how long must the documents be?

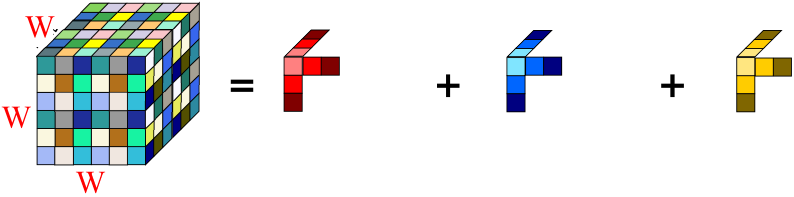
Nondegenerate model (linearly independent topic-word matrix)

Know a $\boldsymbol{W}$ 

such that 

$\boldsymbol{\mathcal{M}}_3$: Distribution of word triples ($\widehat{\boldsymbol{\mathcal{M}}}_3$: Co-occurrence of word triples)

$$\boldsymbol{\mathcal{M}}_3(\boldsymbol{W},\boldsymbol{W},\boldsymbol{W}) = \mathbb{E}[(\boldsymbol{W}^\top \boldsymbol{x})^{\otimes 3}] = \sum_h w_h (\boldsymbol{W}^\top \boldsymbol{a}_h)^{\otimes 3}; \ \widehat{\boldsymbol{\mathcal{M}}}_3(\boldsymbol{W},\boldsymbol{W},\boldsymbol{W}) = \frac{1}{n}\sum_{i=1}^{n} (\boldsymbol{W}^\top \boldsymbol{x}^{(i)})^{\otimes 3}$$



$L \geq 3$: Learning Topic Models through Matrix/Tensor Decomposition

Take Away Message

- Consider topic models satisfying linear independent word distributions under different topics.
- Parameters of topic model for single-topic documents can be efficiently recovered from distribution of three-word documents.
  - Distribution of three-word documents (word triples)

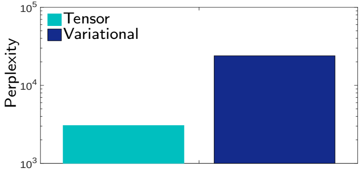
$$M_3 = \mathbb{E}[\boldsymbol{x} \otimes \boldsymbol{x} \otimes \boldsymbol{x}] = \sum_h w_h \boldsymbol{a}_h \otimes \boldsymbol{a}_h \otimes \boldsymbol{a}_h$$

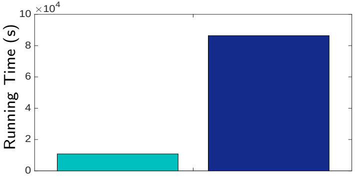
  - $\widehat{M}_3$: Co-occurrence of word triples

- Two-word documents are not sufficient for identifiability.

Tensor Methods Compared with Variational Inference

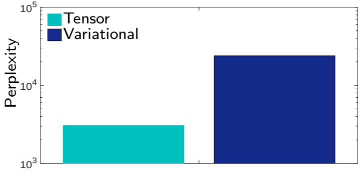
Learning Topics from PubMed on Spark: 8 million docs

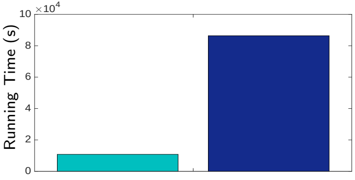




Tensor Methods Compared with Variational Inference

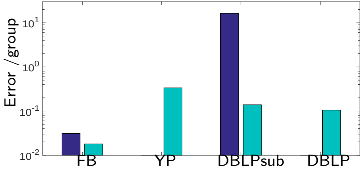
Learning Topics from PubMed on Spark: 8 million docs

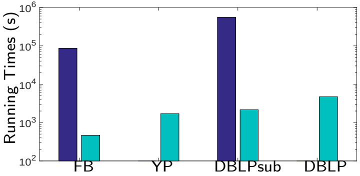




Learning Communities from Graph Connectivity

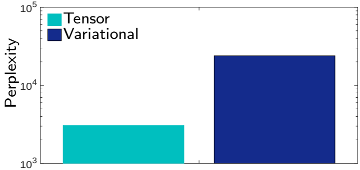
Facebook: $n \sim 20k$ Yelp: $n \sim 40k$ DBLPsub: $n \sim 0.1m$ DBLP: $n \sim 1m$

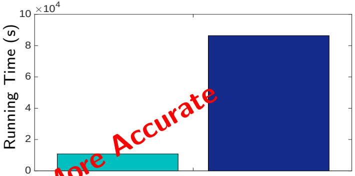




Tensor Methods Compared with Variational Inference

Learning Topics from PubMed on Spark: 8 million docs

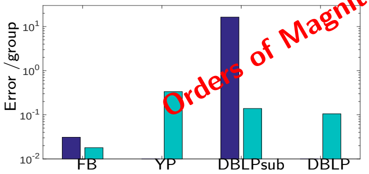


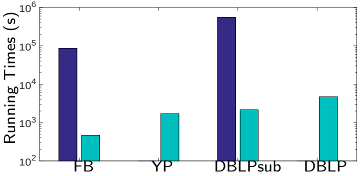


Orders of Magnitude Faster & More Accurate

Learning Communities from Graph Connectivity

Facebook: $n \sim 20k$ Yelp: $n \sim 40k$ DBLPsub: $n \sim 0.1m$ DBLP: $n \sim 1m$



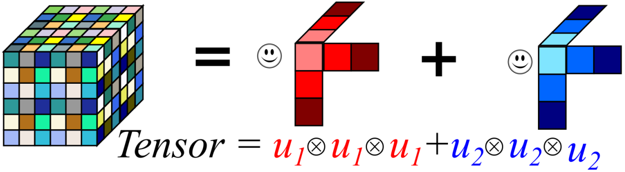


“Online Tensor Methods for Learning Latent Variable Models”, F. Huang, U. Niranjan, M. Hakeem, A. Anandkumar, JMLR14.
“Tensor Methods on Apache Spark”, by F. Huang, A. Anandkumar, Oct. 2015.

Outline

Jennrich's Algorithm (Simplified)

Task: Given tensor $\mathcal{T} = \sum_{h=1}^{K} \boldsymbol{\mu}_h {\otimes}^3$ with linearly independent components $\{\boldsymbol{\mu}_h\}_{h=1}^{K}$, find the components (up to scaling).

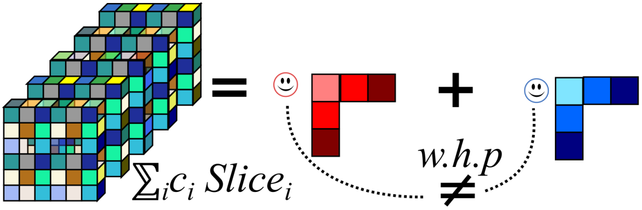


Jennrich's Algorithm (Simplified)

Task: Given tensor $\mathcal{T} = \sum_{h=1}^{K} \boldsymbol{\mu}_h \otimes^3$ with linearly independent components $\{\boldsymbol{\mu}_h\}_{h=1}^{K}$, find the components (up to scaling).

Properties of Tensor Slices

- Linear combination of slices $\mathcal{T}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c}) = \sum_h < \boldsymbol{\mu}_h, \boldsymbol{c} > \boldsymbol{\mu}_h \otimes \boldsymbol{\mu}_h$

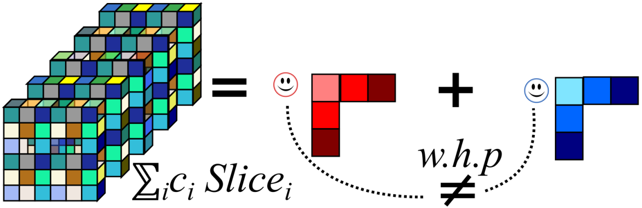


Jennrich's Algorithm (Simplified)

Task: Given tensor $\mathcal{T} = \sum_{h=1}^{K} \boldsymbol{\mu}_h^{\otimes 3}$ with linearly independent components $\{\boldsymbol{\mu}_h\}_{h=1}^{K}$, find the components (up to scaling).

Properties of Tensor Slices

- Linear combination of slices $\mathcal{T}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c}) = \sum_h < \boldsymbol{\mu}_h, \boldsymbol{c} > \boldsymbol{\mu}_h \otimes \boldsymbol{\mu}_h$



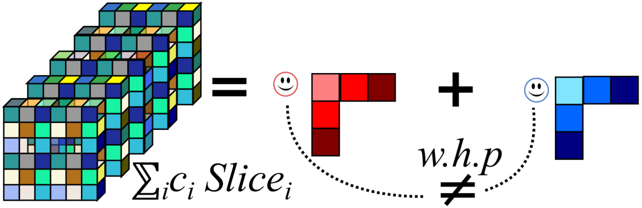
Intuitions for Jennrich's Algorithm

Jennrich's Algorithm (Simplified)

Task: Given tensor $\mathcal{T} = \sum_{h=1}^{K} \boldsymbol{\mu}_h \otimes^3$ with linearly independent components $\{\boldsymbol{\mu}_h\}_{h=1}^{K}$, find the components (up to scaling).

Properties of Tensor Slices

- Linear combination of slices $\mathcal{T}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c}) = \sum_h < \boldsymbol{\mu}_h, \boldsymbol{c} > \boldsymbol{\mu}_h \otimes \boldsymbol{\mu}_h$



Intuitions for Jennrich's Algorithm

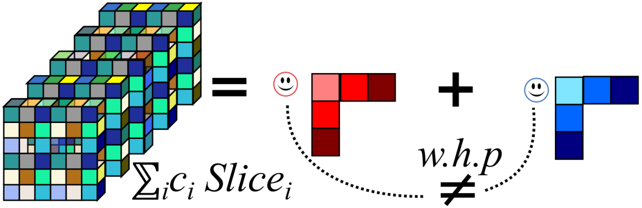
Linear comb. of slices of a tensor share the same set of eigenvectors

Jennrich's Algorithm (Simplified)

Task: Given tensor $\mathcal{T} = \sum_{h=1}^{K} \boldsymbol{\mu}_h {\otimes}^3$ with linearly independent components $\{\boldsymbol{\mu}_h\}_{h=1}^{K}$, find the components (up to scaling).

Properties of Tensor Slices

- Linear combination of slices $\mathcal{T}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c}) = \sum_h < \boldsymbol{\mu}_h, \boldsymbol{c} > \boldsymbol{\mu}_h \otimes \boldsymbol{\mu}_h$



Intuitions for Jennrich's Algorithm

$\boxed{\text{Linear comb. of slices of a tensor share the same set of eigenvectors}}$

$\boxed{\text{The shared eigenvectors are tensor components } \{\boldsymbol{\mu}_h\}_{h=1}^{K}}$

Jennrich's Algorithm (Simplified)

Task: Given tensor $\mathcal{T} = \sum_{h=1}^{K} \boldsymbol{\mu}_h^{\otimes 3}$ with linearly independent components $\{\boldsymbol{\mu}_h\}_{h=1}^{K}$, find the components (up to scaling).

Algorithm Jennrich's Algorithm

Require: Tensor $\mathcal{T} \in \mathbb{R}^{d \times d \times d}$

Ensure: Components $\{\widehat{\boldsymbol{\mu}}_h\}_{h=1}^{K} \overset{\text{a.s.}}{=} \{\boldsymbol{\mu}_h\}_{h=1}^{K}$

1: Sample $\boldsymbol{c}$ and $\boldsymbol{c}'$ independently & uniformly at random from S^{d-1}

2: Return $\{\widehat{\boldsymbol{\mu}}_h\}_{h=1}^{K} \leftarrow$ eigenvectors of $\left(\mathcal{T}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c})\mathcal{T}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c}')^{\dagger}\right)$

Jennrich's Algorithm (Simplified)

Task: Given tensor $\mathcal{T} = \sum_{h=1}^{K} \mu_h^{\otimes 3}$ with linearly independent components $\{\mu_h\}_{h=1}^{K}$, find the components (up to scaling).

Algorithm Jennrich's Algorithm

Require: Tensor $\mathcal{T} \in \mathbb{R}^{d \times d \times d}$

Ensure: Components $\{\widehat{\mu}_h\}_{h=1}^{K} \overset{\text{a.s.}}{=} \{\mu_h\}_{h=1}^{K}$

1: Sample c and c' independently & uniformly at random from S^{d-1}

2: Return $\{\widehat{\mu}_h\}_{h=1}^{K} \leftarrow$ eigenvectors of $\left(\mathcal{T}(I, I, c)\mathcal{T}(I, I, c')^{\dagger}\right)$

Consistency of Jennrich's Algorithm?

Estimators $\{\widehat{\mu}_h\}_{h=1}^{K} \equiv$ unknown components $\{\mu_h\}_{h=1}^{K}$ (up to scaling)?

Analysis of Consistency of Jennrich's algorithm

Recall: Linear comb. of slices share eigenvectors $\{\boldsymbol{\mu}_h\}_{h=1}^{K}$, i.e.,

$$\boldsymbol{\mathcal{T}}(\boldsymbol{I},\boldsymbol{I},\boldsymbol{c})\boldsymbol{\mathcal{T}}(\boldsymbol{I},\boldsymbol{I},\boldsymbol{c}')^{\dagger} \overset{\text{a.s.}}{=} \boldsymbol{U}\boldsymbol{D}_{\boldsymbol{c}}\boldsymbol{U}^{\top}(\boldsymbol{U}^{\top})^{\dagger}\boldsymbol{D}_{\boldsymbol{c}'}^{-1}\boldsymbol{U}^{\dagger} \overset{\text{a.s.}}{=} \boldsymbol{U}(\boldsymbol{D}_{\boldsymbol{c}}\boldsymbol{D}_{\boldsymbol{c}'}^{-1})\boldsymbol{U}^{\dagger},$$

where $\boldsymbol{U} = [\boldsymbol{\mu}_1|\ldots|\boldsymbol{\mu}_K]$ are the linearly independent tensor components and $\boldsymbol{D}_{\boldsymbol{c}} = \mathrm{Diag}\Big(< \boldsymbol{\mu}_1, \boldsymbol{c} >, \ldots, < \boldsymbol{\mu}_K, \boldsymbol{c} > \Big)$ is diagonal.

Analysis of Consistency of Jennrich's algorithm

Recall: Linear comb. of slices share eigenvectors $\{\boldsymbol{\mu}_h\}_{h=1}^K$, i.e.,

$$\boldsymbol{\mathcal{T}}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c})\boldsymbol{\mathcal{T}}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c}')^\dagger \stackrel{\text{a.s.}}{=} \boldsymbol{U}\boldsymbol{D}_{\boldsymbol{c}}\boldsymbol{U}^\top(\boldsymbol{U}^\top)^\dagger \boldsymbol{D}_{\boldsymbol{c}'}^{-1}\boldsymbol{U}^\dagger \stackrel{\text{a.s.}}{=} \boldsymbol{U}(\boldsymbol{D}_{\boldsymbol{c}}\boldsymbol{D}_{\boldsymbol{c}'}^{-1})\boldsymbol{U}^\dagger,$$

where $\boldsymbol{U} = [\boldsymbol{\mu}_1|\ldots|\boldsymbol{\mu}_K]$ are the linearly independent tensor components and $\boldsymbol{D}_{\boldsymbol{c}} = \text{Diag}\Big(<\boldsymbol{\mu}_1, \boldsymbol{c}>, \ldots, <\boldsymbol{\mu}_K, \boldsymbol{c}> \Big)$ is diagonal.

By linear independence of $\{\boldsymbol{\mu}_i\}_{i=1}^K$ and random choice of $\boldsymbol{c}$ and $\boldsymbol{c}'$:

1. $\boldsymbol{U}$ has rank K;

Analysis of Consistency of Jennrich's algorithm

Recall: Linear comb. of slices share eigenvectors $\{\boldsymbol{\mu}_h\}_{h=1}^K$, i.e.,

$$\mathcal{T}(\boldsymbol{I},\boldsymbol{I},\boldsymbol{c})\mathcal{T}(\boldsymbol{I},\boldsymbol{I},\boldsymbol{c}')^\dagger \overset{\text{a.s.}}{=} \boldsymbol{U}\boldsymbol{D}_{\boldsymbol{c}}\boldsymbol{U}^\top(\boldsymbol{U}^\top)^\dagger\boldsymbol{D}_{\boldsymbol{c}'}^{-1}\boldsymbol{U}^\dagger \overset{\text{a.s.}}{=} \boldsymbol{U}(\boldsymbol{D}_{\boldsymbol{c}}\boldsymbol{D}_{\boldsymbol{c}'}^{-1})\boldsymbol{U}^\dagger,$$

where $\boldsymbol{U} = [\boldsymbol{\mu}_1|\ldots|\boldsymbol{\mu}_K]$ are the linearly independent tensor components and $\boldsymbol{D}_{\boldsymbol{c}} = \text{Diag}\Big(< \boldsymbol{\mu}_1, \boldsymbol{c} >, \ldots, < \boldsymbol{\mu}_K, \boldsymbol{c} > \Big)$ is diagonal.

By linear independence of $\{\boldsymbol{\mu}_i\}_{i=1}^K$ and random choice of $\boldsymbol{c}$ and $\boldsymbol{c}'$:

1. $\boldsymbol{U}$ has rank K;
2. $\boldsymbol{D}_{\boldsymbol{c}}$ and $\boldsymbol{D}_{\boldsymbol{c}'}$ are invertible (a.s.);

Analysis of Consistency of Jennrich's algorithm

Recall: Linear comb. of slices share eigenvectors $\{\boldsymbol{\mu}_h\}_{h=1}^K$, i.e.,

$$\boldsymbol{\mathcal{T}}(\boldsymbol{I},\boldsymbol{I},\boldsymbol{c})\boldsymbol{\mathcal{T}}(\boldsymbol{I},\boldsymbol{I},\boldsymbol{c}')^{\dagger} \overset{\text{a.s.}}{=} \boldsymbol{U}\boldsymbol{D}_{\boldsymbol{c}}\boldsymbol{U}^{\top}(\boldsymbol{U}^{\top})^{\dagger}\boldsymbol{D}_{\boldsymbol{c}'}^{-1}\boldsymbol{U}^{\dagger} \overset{\text{a.s.}}{=} \boldsymbol{U}(\boldsymbol{D}_{\boldsymbol{c}}\boldsymbol{D}_{\boldsymbol{c}'}^{-1})\boldsymbol{U}^{\dagger},$$

where $\boldsymbol{U} = [\boldsymbol{\mu}_1|\ldots|\boldsymbol{\mu}_K]$ are the linearly independent tensor components and $\boldsymbol{D}_{\boldsymbol{c}} = \text{Diag}\Big(< \boldsymbol{\mu}_1, \boldsymbol{c} >, \ldots, < \boldsymbol{\mu}_K, \boldsymbol{c} > \Big)$ is diagonal.

By linear independence of $\{\boldsymbol{\mu}_i\}_{i=1}^K$ and random choice of $\boldsymbol{c}$ and $\boldsymbol{c}'$:

1. $\boldsymbol{U}$ has rank K;
2. $\boldsymbol{D}_{\boldsymbol{c}}$ and $\boldsymbol{D}_{\boldsymbol{c}'}$ are invertible (a.s.);
3. Diagonal entries of $\boldsymbol{D}_{\boldsymbol{c}}\boldsymbol{D}_{\boldsymbol{c}'}^{-1}$ are distinct (a.s.);

Analysis of Consistency of Jennrich's algorithm

Recall: Linear comb. of slices share eigenvectors $\{\boldsymbol{\mu}_h\}_{h=1}^K$,

i.e.,

$$\mathcal{T}(\boldsymbol{I},\boldsymbol{I},\boldsymbol{c})\mathcal{T}(\boldsymbol{I},\boldsymbol{I},\boldsymbol{c}')^{\dagger} \overset{\text{a.s.}}{=} \boldsymbol{U}\boldsymbol{D_c}\boldsymbol{U}^{\top}(\boldsymbol{U}^{\top})^{\dagger}\boldsymbol{D}_{\boldsymbol{c}'}^{-1}\boldsymbol{U}^{\dagger} \overset{\text{a.s.}}{=} \boldsymbol{U}(\boldsymbol{D_c}\boldsymbol{D}_{\boldsymbol{c}'}^{-1})\boldsymbol{U}^{\dagger},$$

where $\boldsymbol{U} = [\boldsymbol{\mu}_1|\ldots|\boldsymbol{\mu}_K]$ are the linearly independent tensor components and $\boldsymbol{D_c} = \text{Diag}\Big(<\boldsymbol{\mu}_1,\boldsymbol{c}>,\ldots,<\boldsymbol{\mu}_K,\boldsymbol{c}> \Big)$ is diagonal.

By linear independence of $\{\boldsymbol{\mu}_i\}_{i=1}^K$ and random choice of $\boldsymbol{c}$ and $\boldsymbol{c}'$:

1. $\boldsymbol{U}$ has rank K;
2. $\boldsymbol{D_c}$ and $\boldsymbol{D}_{\boldsymbol{c}'}$ are invertible (a.s.);
3. Diagonal entries of $\boldsymbol{D_c}\boldsymbol{D}_{\boldsymbol{c}'}^{-1}$ are distinct (a.s.);

So $\{\boldsymbol{\mu}_i\}_{i=1}^K$ are the eigenvectors of $\mathcal{T}(\boldsymbol{I},\boldsymbol{I},\boldsymbol{c})\mathcal{T}(\boldsymbol{I},\boldsymbol{I},\boldsymbol{c})^{\dagger}$ with distinct non-zero eigenvalues.

$\boxed{\text{Jennrich's algorithm is consistent}}$

Error-tolerant algorithms for tensor decompositions

Moment Estimator: Empirical Moments

Moment Estimator: Empirical Moments

- Moments $\mathbb{E}_{\boldsymbol{\theta}}[f(\boldsymbol{X})]$ are functions of model parameters $\boldsymbol{\theta}$
- Empirical Moments $\widehat{\mathbb{E}}[f(\boldsymbol{X})]$ are computed using iid samples $\{\boldsymbol{x}_i\}_{i=1}^n$ only

Moment Estimator: Empirical Moments

- Moments $\mathbb{E}_{\boldsymbol{\theta}}[f(\boldsymbol{X})]$ are functions of model parameters $\boldsymbol{\theta}$
- Empirical Moments $\widehat{\mathbb{E}}[f(\boldsymbol{X})]$ are computed using iid samples $\{\boldsymbol{x}_i\}_{i=1}^n$ only

Example

- Third Order Moment: distribution of word triples

$$\mathbb{E}[\boldsymbol{x} \otimes \boldsymbol{x} \otimes \boldsymbol{x}] = \sum_h w_h \boldsymbol{a}_h \otimes \boldsymbol{a}_h \otimes \boldsymbol{a}_h$$

- Empirical Third Order Moment: co-occurrence frequency of word triples

$$\widehat{\mathbb{E}}[\boldsymbol{x} \otimes \boldsymbol{x} \otimes \boldsymbol{x}] = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i \otimes \boldsymbol{x}_i \otimes \boldsymbol{x}_i$$

Moment Estimator: Empirical Moments

- Moments $\mathbb{E}_{\boldsymbol{\theta}}[f(\boldsymbol{X})]$ are functions of model parameters $\boldsymbol{\theta}$
- Empirical Moments $\widehat{\mathbb{E}}[f(\boldsymbol{X})]$ are computed using iid samples $\{\boldsymbol{x}_i\}_{i=1}^n$ only

Example

- Third Order Moment: distribution of word triples

$$\mathbb{E}[\boldsymbol{x} \otimes \boldsymbol{x} \otimes \boldsymbol{x}] = \sum_h w_h \boldsymbol{a}_h \otimes \boldsymbol{a}_h \otimes \boldsymbol{a}_h$$

- Empirical Third Order Moment: co-occurrence frequency of word triples

$$\widehat{\mathbb{E}}[\boldsymbol{x} \otimes \boldsymbol{x} \otimes \boldsymbol{x}] = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i \otimes \boldsymbol{x}_i \otimes \boldsymbol{x}_i$$

- Inevitably expect error of order $n^{-\frac{1}{2}}$ in some norm, e.g.,

Operator norm: $\|\mathbb{E}[\boldsymbol{x} \otimes \boldsymbol{x} \otimes \boldsymbol{x}] - \widehat{\mathbb{E}}[\boldsymbol{x} \otimes \boldsymbol{x} \otimes \boldsymbol{x}]\| \lessapprox n^{-\frac{1}{2}}$
where $\|\boldsymbol{\mathcal{T}}\| := \sup_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z} \in S^{d-1}} \boldsymbol{\mathcal{T}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$

Frobenius norm: $\|\mathbb{E}[\boldsymbol{x} \otimes \boldsymbol{x} \otimes \boldsymbol{x}] - \widehat{\mathbb{E}}[\boldsymbol{x} \otimes \boldsymbol{x} \otimes \boldsymbol{x}]\|_F \lessapprox n^{-\frac{1}{2}}$
where $\|\boldsymbol{\mathcal{T}}\|_F := \sqrt{\sum_{i,j,k} T_{i,j,k}^2}$

Stability of Jennrich's Algorithm

Recall Jennrich's algorithm

Given tensor $\boldsymbol{\mathcal{T}} = \sum_{h=1}^{K} \boldsymbol{\mu}_h {\otimes}^3$ with linearly independent components $\{\boldsymbol{\mu}_h\}_{h=1}^{K}$, find the components (up to scaling).

Algorithm Jennrich's Algorithm

Require: Tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{d \times d \times d}$

Ensure: Components $\{\widehat{\boldsymbol{\mu}}_h\}_{h=1}^{K} \overset{\text{a.s.}}{=} \{\boldsymbol{\mu}_h\}_{h=1}^{K}$

1: Sample $\boldsymbol{c}$ and $\boldsymbol{c}'$ independently & uniformly at random from S^{d-1}

2: Return $\{\widehat{\boldsymbol{\mu}}_h\}_{h=1}^{K} \leftarrow$ eigenvectors of $\left(\boldsymbol{\mathcal{T}}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c})\boldsymbol{\mathcal{T}}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c}')^{\dagger}\right)$

Stability of Jennrich's Algorithm

Recall Jennrich's algorithm

Given tensor $\mathcal{T} = \sum_{h=1}^{K} \boldsymbol{\mu}_h{\otimes}^3$ with linearly independent components $\{\boldsymbol{\mu}_h\}_{h=1}^{K}$, find the components (up to scaling).

Algorithm Jennrich's Algorithm

Require: Tensor $\mathcal{T} \in \mathbb{R}^{d\times d\times d}$

Ensure: Components $\{\widehat{\boldsymbol{\mu}}_h\}_{h=1}^{K} \overset{\text{a.s.}}{=} \{\boldsymbol{\mu}_h\}_{h=1}^{K}$

1: Sample $\boldsymbol{c}$ and $\boldsymbol{c}'$ independently & uniformly at random from S^{d-1}

2: Return $\{\widehat{\boldsymbol{\mu}}_h\}_{h=1}^{K} \leftarrow$ eigenvectors of $\left(\mathcal{T}(\boldsymbol{I},\boldsymbol{I},\boldsymbol{c})\mathcal{T}(\boldsymbol{I},\boldsymbol{I},\boldsymbol{c}')^{\dagger}\right)$

Challenge: Only have access to $\widehat{\mathcal{T}}$ such that $\|\widehat{\mathcal{T}} - \mathcal{T}\| \lesssim n^{-\frac{1}{2}}$

Stability of Jennrich's Algorithm

Recall Jennrich's algorithm

Given tensor $\boldsymbol{\mathcal{T}} = \sum_{h=1}^{K} \boldsymbol{\mu}_h{\otimes}^3$ with linearly independent components $\{\boldsymbol{\mu}_h\}_{h=1}^{K}$, find the components (up to scaling).

Algorithm Jennrich's Algorithm

Require: Tensor $\widehat{\boldsymbol{\mathcal{T}}} \in \mathbb{R}^{d\times d\times d}$

Ensure: Components $\{\widehat{\boldsymbol{\mu}}_h\}_{h=1}^{K} \overset{\text{a.s.}}{=} \{\boldsymbol{\mu}_h\}_{h=1}^{K}$?

1: Sample $\boldsymbol{c}$ and $\boldsymbol{c}'$ independently & uniformly at random from S^{d-1}

2: Return $\{\widehat{\boldsymbol{\mu}}_h\}_{h=1}^{K} \leftarrow$ eigenvectors of $\left(\widehat{\boldsymbol{\mathcal{T}}}(\boldsymbol{I},\boldsymbol{I},\boldsymbol{c})\widehat{\boldsymbol{\mathcal{T}}}(\boldsymbol{I},\boldsymbol{I},\boldsymbol{c}')^{\dagger}\right)$

Challenge: Only have access to $\widehat{\boldsymbol{\mathcal{T}}}$ such that $\|\widehat{\boldsymbol{\mathcal{T}}} - \boldsymbol{\mathcal{T}}\| \lessapprox n^{-\frac{1}{2}}$

Stability of Jennrich's Algorithm

Recall Jennrich's algorithm

Given tensor $\boldsymbol{\mathcal{T}} = \sum_{h=1}^{K} \boldsymbol{\mu}_h{\otimes}^3$ with linearly independent components $\{\boldsymbol{\mu}_h\}_{h=1}^{K}$, find the components (up to scaling).

Algorithm Jennrich's Algorithm

Require: Tensor $\widehat{\boldsymbol{\mathcal{T}}} \in \mathbb{R}^{d \times d \times d}$

Ensure: Components $\{\widehat{\boldsymbol{\mu}}_h\}_{h=1}^{K} \overset{\text{a.s.}}{=} \{\boldsymbol{\mu}_h\}_{h=1}^{K}$?

1: Sample $\boldsymbol{c}$ and $\boldsymbol{c}'$ independently & uniformly at random from S^{d-1}

2: Return $\{\widehat{\boldsymbol{\mu}}_h\}_{h=1}^{K} \leftarrow$ eigenvectors of $\left(\widehat{\boldsymbol{\mathcal{T}}}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c})\widehat{\boldsymbol{\mathcal{T}}}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c}')^{\dagger}\right)$

Stability of eigenvectors requires eigenvalue gaps

Stability of Jennrich's Algorithm

Recall Jennrich's algorithm

Given tensor $\mathcal{T} = \sum_{h=1}^{K} \mu_h {\otimes}^3$ with linearly independent components $\{\mu_h\}_{h=1}^{K}$, find the components (up to scaling).

Algorithm Jennrich's Algorithm

Require: Tensor $\widehat{\mathcal{T}} \in \mathbb{R}^{d \times d \times d}$

Ensure: Components $\{\widehat{\mu}_h\}_{h=1}^{K} \overset{\text{a.s.}}{=} \{\mu_h\}_{h=1}^{K}$?

1: Sample c and c' independently & uniformly at random from S^{d-1}

2: Return $\{\widehat{\mu}_h\}_{h=1}^{K} \leftarrow$ eigenvectors of $\left(\widehat{\mathcal{T}}(I, I, c)\widehat{\mathcal{T}}(I, I, c')^\dagger\right)$

Stability of eigenvectors requires eigenvalue gaps

- To ensure eigenvalue gaps for $\widehat{\mathcal{T}}(\cdot, \cdot, c)\widehat{\mathcal{T}}(\cdot, \cdot, c)^\dagger$, $\|\widehat{\mathcal{T}}(\cdot, \cdot, c)\widehat{\mathcal{T}}(\cdot, \cdot, c)^\dagger - \mathcal{T}(\cdot, \cdot, c)\mathcal{T}(\cdot, \cdot, c)^\dagger\| \ll \Delta$ is needed.

Stability of Jennrich's Algorithm

Recall Jennrich's algorithm

Given tensor $\boldsymbol{\mathcal{T}} = \sum_{h=1}^{K} \boldsymbol{\mu}_h {\otimes}^3$ with linearly independent components $\{\boldsymbol{\mu}_h\}_{h=1}^{K}$, find the components (up to scaling).

Algorithm Jennrich's Algorithm

Require: Tensor $\widehat{\boldsymbol{\mathcal{T}}} \in \mathbb{R}^{d \times d \times d}$

Ensure: Components $\{\widehat{\boldsymbol{\mu}}_h\}_{h=1}^{K} \overset{\text{a.s.}}{=} \{\boldsymbol{\mu}_h\}_{h=1}^{K}$?

1: Sample $\boldsymbol{c}$ and $\boldsymbol{c}'$ independently & uniformly at random from S^{d-1}

2: Return $\{\widehat{\boldsymbol{\mu}}_h\}_{h=1}^{K} \leftarrow$ eigenvectors of $\left(\widehat{\boldsymbol{\mathcal{T}}}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c}) \widehat{\boldsymbol{\mathcal{T}}}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c}')^{\dagger}\right)$

Stability of eigenvectors requires eigenvalue gaps

- To ensure eigenvalue gaps for $\widehat{\boldsymbol{\mathcal{T}}}(\cdot, \cdot, \boldsymbol{c}) \widehat{\boldsymbol{\mathcal{T}}}(\cdot, \cdot, \boldsymbol{c})^{\dagger}$, $\|\widehat{\boldsymbol{\mathcal{T}}}(\cdot, \cdot, \boldsymbol{c}) \widehat{\boldsymbol{\mathcal{T}}}(\cdot, \cdot, \boldsymbol{c})^{\dagger} - \boldsymbol{\mathcal{T}}(\cdot, \cdot, \boldsymbol{c}) \boldsymbol{\mathcal{T}}(\cdot, \cdot, \boldsymbol{c})^{\dagger}\| \ll \Delta$ is needed.
- Ultimately, $\|\widehat{\boldsymbol{\mathcal{T}}} - \boldsymbol{\mathcal{T}}\|_F \ll \frac{1}{\operatorname{poly} d}$ is required.

Stability of Jennrich's Algorithm

Recall Jennrich's algorithm

Given tensor $\boldsymbol{\mathcal{T}} = \sum_{h=1}^{K} \boldsymbol{\mu}_h{}^{\otimes 3}$ with linearly independent components $\{\boldsymbol{\mu}_h\}_{h=1}^{K}$, find the components (up to scaling).

Algorithm Jennrich's Algorithm

Require: Tensor $\widehat{\boldsymbol{\mathcal{T}}} \in \mathbb{R}^{d \times d \times d}$

Ensure: Components $\{\widehat{\boldsymbol{\mu}}_h\}_{h=1}^{K} \overset{\text{a.s.}}{=} \{\boldsymbol{\mu}_h\}_{h=1}^{K}$?

1: Sample $\boldsymbol{c}$ and $\boldsymbol{c}'$ independently & uniformly at random from S^{d-1}

2: Return $\{\widehat{\boldsymbol{\mu}}_h\}_{h=1}^{K} \leftarrow$ eigenvectors of $\left(\widehat{\boldsymbol{\mathcal{T}}}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c}) \widehat{\boldsymbol{\mathcal{T}}}(\boldsymbol{I}, \boldsymbol{I}, \boldsymbol{c}')^{\dagger}\right)$

Stability of eigenvectors requires eigenvalue gaps

- To ensure eigenvalue gaps for $\widehat{\boldsymbol{\mathcal{T}}}(\cdot, \cdot, \boldsymbol{c}) \widehat{\boldsymbol{\mathcal{T}}}(\cdot, \cdot, \boldsymbol{c})^{\dagger}$, $\|\widehat{\boldsymbol{\mathcal{T}}}(\cdot, \cdot, \boldsymbol{c}) \widehat{\boldsymbol{\mathcal{T}}}(\cdot, \cdot, \boldsymbol{c})^{\dagger} - \boldsymbol{\mathcal{T}}(\cdot, \cdot, \boldsymbol{c}) \boldsymbol{\mathcal{T}}(\cdot, \cdot, \boldsymbol{c})^{\dagger}\| \ll \Delta$ is needed.
- Ultimately, $\|\widehat{\boldsymbol{\mathcal{T}}} - \boldsymbol{\mathcal{T}}\|_F \ll \frac{1}{\operatorname{poly} d}$ is required. A different approach?

Initial Ideas

In many applications, we estimate moments of the form

$$\mathcal{M}_3 = \sum_{h=1}^{K} w_h \boldsymbol{a}_h \otimes^3,$$

where $\{\boldsymbol{a}_h\}_{h=1}^K$ are assumed to be linearly independent.

What if $\{\boldsymbol{a}_h\}_{h=1}^K$ has orthonormal columns?

Initial Ideas

In many applications, we estimate moments of the form

$$\boldsymbol{\mathcal{M}}_3 = \sum_{h=1}^{K} w_h \boldsymbol{a}_h \otimes^3,$$

where $\{\boldsymbol{a}_h\}_{h=1}^{K}$ are assumed to be linearly independent.

What if $\{\boldsymbol{a}_h\}_{h=1}^{K}$ has orthonormal columns?

$$\boxed{\boldsymbol{\mathcal{M}}_3(I, \boldsymbol{a}_i, \boldsymbol{a}_i) = \sum_h w_h \langle \boldsymbol{a}_h, \boldsymbol{a}_i \rangle^2 \boldsymbol{a}_h = w_i \boldsymbol{a}_i, \ \forall i.}$$

Initial Ideas

In many applications, we estimate moments of the form

$$\mathcal{M}_3 = \sum_{h=1}^{K} w_h \boldsymbol{a}_h {\otimes}^3,$$

where $\{\boldsymbol{a}_h\}_{h=1}^K$ are assumed to be linearly independent.

What if $\{\boldsymbol{a}_h\}_{h=1}^K$ has orthonormal columns?

$$\boxed{\mathcal{M}_3(I, \boldsymbol{a}_i, \boldsymbol{a}_i) = \sum_h w_h \langle \boldsymbol{a}_h, \boldsymbol{a}_i \rangle^2 \boldsymbol{a}_h = w_i \boldsymbol{a}_i, \ \forall i.}$$

- Analogous to matrix eigenvectors: $\boldsymbol{M}\boldsymbol{v} = \boldsymbol{M}(\boldsymbol{I}, \boldsymbol{v}) = \lambda \boldsymbol{v}$.

Initial Ideas

In many applications, we estimate moments of the form

$$\mathcal{M}_3 = \sum_{h=1}^{K} w_h \boldsymbol{a}_h {\otimes}^3,$$

where $\{\boldsymbol{a}_h\}_{h=1}^{K}$ are assumed to be linearly independent.

What if $\{\boldsymbol{a}_h\}_{h=1}^{K}$ has orthonormal columns?

$$\boxed{\mathcal{M}_3(I, \boldsymbol{a}_i, \boldsymbol{a}_i) = \sum_h w_h \langle \boldsymbol{a}_h, \boldsymbol{a}_i \rangle^2 \boldsymbol{a}_h = w_i \boldsymbol{a}_i, \ \forall i.}$$

- Analogous to matrix eigenvectors: $\boldsymbol{M}\boldsymbol{v} = \boldsymbol{M}(\boldsymbol{I}, \boldsymbol{v}) = \lambda \boldsymbol{v}$.
- **Define** orthonormal $\{\boldsymbol{a}_h\}_{h=1}^{K}$ as eigenvectors of tensor $\mathcal{M}_3$.

Initial Ideas

In many applications, we estimate moments of the form

$$\mathcal{M}_3 = \sum_{h=1}^{K} w_h \boldsymbol{a}_h {\otimes}^3,$$

where $\{\boldsymbol{a}_h\}_{h=1}^K$ are assumed to be linearly independent.

What if $\{\boldsymbol{a}_h\}_{h=1}^K$ has orthonormal columns?

$$\boxed{\mathcal{M}_3(I, \boldsymbol{a}_i, \boldsymbol{a}_i) = \sum_h w_h \langle \boldsymbol{a}_h, \boldsymbol{a}_i \rangle^2 \boldsymbol{a}_h = w_i \boldsymbol{a}_i, \ \forall i.}$$

- Analogous to matrix eigenvectors: $\boldsymbol{M}\boldsymbol{v} = \boldsymbol{M}(\boldsymbol{I}, \boldsymbol{v}) = \lambda \boldsymbol{v}$.
- **Define** orthonormal $\{\boldsymbol{a}_h\}_{h=1}^K$ as eigenvectors of tensor $\mathcal{M}_3$.

Two Problems

- $\{\boldsymbol{a}_h\}_{h=1}^K$ is not orthogonal in general.
- How to find eigenvectors of a tensor?

Initial Ideas

In many applications, we estimate moments of the form

$$\boldsymbol{\mathcal{M}}_3 = \sum_{h=1}^{K} w_h \boldsymbol{a}_h {\otimes}^3,$$

where $\{\boldsymbol{a}_h\}_{h=1}^K$ are assumed to be linearly independent.

What if $\{\boldsymbol{a}_h\}_{h=1}^K$ has orthonormal columns?

$$\boxed{\boldsymbol{\mathcal{M}}_3(I, \boldsymbol{a}_i, \boldsymbol{a}_i) = \textstyle\sum_h w_h \langle \boldsymbol{a}_h, \boldsymbol{a}_i \rangle^2 \boldsymbol{a}_h = w_i \boldsymbol{a}_i, \ \forall i.}$$

- Analogous to matrix eigenvectors: $\boldsymbol{M}\boldsymbol{v} = \boldsymbol{M}(\boldsymbol{I}, \boldsymbol{v}) = \lambda \boldsymbol{v}$.
- **Define** orthonormal $\{\boldsymbol{a}_h\}_{h=1}^K$ as eigenvectors of tensor $\boldsymbol{\mathcal{M}}_3$.

Two Problems

- $\{\boldsymbol{a}_h\}_{h=1}^K$ **is not orthogonal in general.**
- How to find eigenvectors of a tensor?

Whitening is the process of finding a whitening matrix $\boldsymbol{W}$ such that multi-linear operation (using $\boldsymbol{W}$) on $\boldsymbol{\mathcal{M}}_3$ orthogonalize its components:

$$\begin{aligned}\boldsymbol{\mathcal{M}}_3(\boldsymbol{W},\boldsymbol{W},\boldsymbol{W}) &= \sum_h w_h(\boldsymbol{W}^\top \boldsymbol{a}_h)^{\otimes 3} \\ &= \sum_h w_h \boldsymbol{v}_h^{\otimes 3}, \quad \boldsymbol{v}_h \perp \boldsymbol{v}_{h'},\ \forall\, h \neq h'\end{aligned}$$

Whitening

Given

$$\mathcal{M}_3 = \sum_h w_h \boldsymbol{a}_h {\otimes}^3, \quad \boldsymbol{M}_2 = \sum_h w_h \boldsymbol{a}_h \otimes \boldsymbol{a}_h,$$

Whitening

Given

$$\mathcal{M}_3 = \sum_h w_h \boldsymbol{a}_h^{\otimes 3}, \quad \boldsymbol{M}_2 = \sum_h w_h \boldsymbol{a}_h \otimes \boldsymbol{a}_h,$$

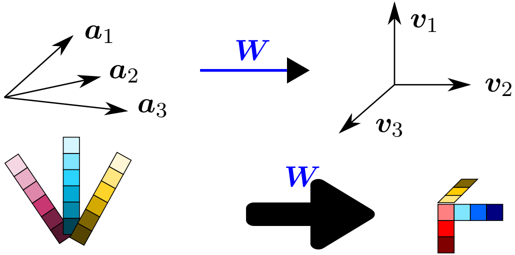
- Find whitening matrix $\boldsymbol{W}$ s.t. $\boldsymbol{W}^\top \boldsymbol{a}_h = \boldsymbol{v}_h$ are orthogonal.

Whitening

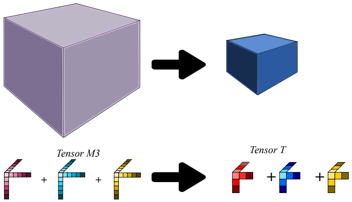
Given

$$\mathcal{M}_3 = \sum_h w_h \boldsymbol{a}_h^{\otimes 3}, \quad \boldsymbol{M}_2 = \sum_h w_h \boldsymbol{a}_h \otimes \boldsymbol{a}_h,$$

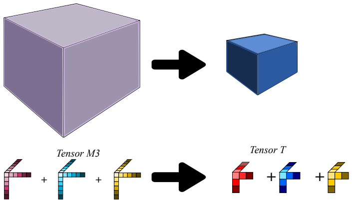
- Find whitening matrix $\boldsymbol{W}$ s.t. $\boldsymbol{W}^\top \boldsymbol{a}_h = \boldsymbol{v}_h$ are orthogonal.
- When $\{\boldsymbol{a}_h\}_{h=1}^K \in \mathbb{R}^{d \times K}$ has full column rank, it is an invertible transformation.



Using Whitening to Obtain Orthogonal Tensor



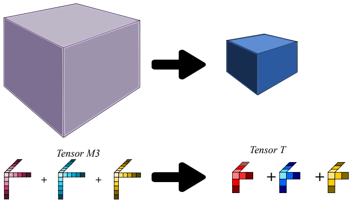
Using Whitening to Obtain Orthogonal Tensor



Multi-linear transform

- $\mathcal{T} = \mathcal{M}_3(\boldsymbol{W}, \boldsymbol{W}, \boldsymbol{W}) = \sum_h w_h (\boldsymbol{W}^\top \boldsymbol{a}_h)^{\otimes 3}$.

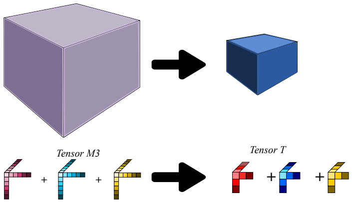
Using Whitening to Obtain Orthogonal Tensor



Multi-linear transform

- $\mathcal{T} = \mathcal{M}_3(\boldsymbol{W}, \boldsymbol{W}, \boldsymbol{W}) = \sum_h w_h (\boldsymbol{W}^\top \boldsymbol{a}_h)^{\otimes 3}$.
- $\mathcal{T} = \sum_{h \in [K]} w_h \cdot \boldsymbol{v}_h {\otimes}^3$ has orthogonal components.

Using Whitening to Obtain Orthogonal Tensor



Multi-linear transform

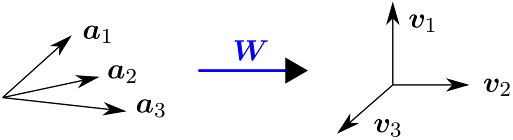
- $\mathcal{T} = \mathcal{M}_3(\boldsymbol{W}, \boldsymbol{W}, \boldsymbol{W}) = \sum_h w_h (\boldsymbol{W}^\top \boldsymbol{a}_h)^{\otimes 3}$.
- $\mathcal{T} = \sum_{h \in [K]} w_h \cdot \boldsymbol{v}_h {\otimes}^3$ has orthogonal components.
- Dimensionality reduction when $K \ll d$, as $\mathcal{M}_3 \in \mathbb{R}^{d \times d \times d}$ and $\mathcal{T} \in \mathbb{R}^{K \times K \times K}$.

How to Find Whitening Matrix?

Given

$$\mathcal{M}_3 = \sum_h w_h \boldsymbol{a}_h^{\otimes 3}, \quad \boldsymbol{M}_2 = \sum_h w_h \boldsymbol{a}_h \otimes \boldsymbol{a}_h,$$

Goal: $\boldsymbol{W}$ such that

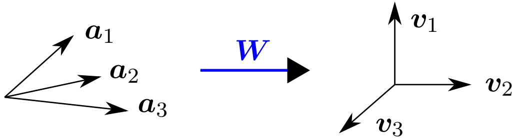


How to Find Whitening Matrix?

Given

$$\mathcal{M}_3 = \sum_h w_h \boldsymbol{a}_h {\otimes}^3, \quad \boldsymbol{M}_2 = \sum_h w_h \boldsymbol{a}_h \otimes \boldsymbol{a}_h,$$

Goal: $\boldsymbol{W}$ such that



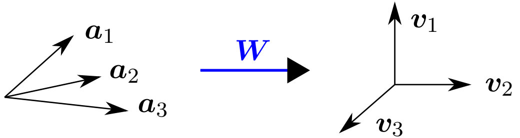
- Use pairwise moments $\boldsymbol{M}_2$ to find $\boldsymbol{W}$ s.t. $\boldsymbol{W}^\top \boldsymbol{M}_2 \boldsymbol{W} = \boldsymbol{I}$.

How to Find Whitening Matrix?

Given

$$\mathcal{M}_3 = \sum_h w_h \boldsymbol{a}_h \otimes^3, \quad \boldsymbol{M}_2 = \sum_h w_h \boldsymbol{a}_h \otimes \boldsymbol{a}_h,$$

Goal: $\boldsymbol{W}$ such that



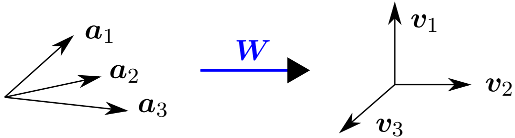
- Use pairwise moments $\boldsymbol{M}_2$ to find $\boldsymbol{W}$ s.t. $\boldsymbol{W}^\top \boldsymbol{M}_2 \boldsymbol{W} = \boldsymbol{I}$.
- $\boldsymbol{W} = \boldsymbol{U}\text{Diag}(\tilde{\boldsymbol{\lambda}}^{-1/2})$, where Eigen-decomposition $\boldsymbol{M}_2 = \boldsymbol{U}\text{Diag}(\tilde{\boldsymbol{\lambda}})\boldsymbol{U}^\top$.

How to Find Whitening Matrix?

Given

$$\mathcal{M}_3 = \sum_h w_h \boldsymbol{a}_h {\otimes}^3, \quad \boldsymbol{M}_2 = \sum_h w_h \boldsymbol{a}_h \otimes \boldsymbol{a}_h,$$

Goal: $\boldsymbol{W}$ such that



- Use pairwise moments $\boldsymbol{M}_2$ to find $\boldsymbol{W}$ s.t. $\boldsymbol{W}^\top \boldsymbol{M}_2 \boldsymbol{W} = \boldsymbol{I}$.
- $\boldsymbol{W} = \boldsymbol{U}\mathrm{Diag}(\tilde{\boldsymbol{\lambda}}^{-1/2})$, where Eigen-decomposition $\boldsymbol{M}_2 = \boldsymbol{U}\mathrm{Diag}(\tilde{\boldsymbol{\lambda}})\boldsymbol{U}^\top$.
- $\boldsymbol{V} := \boldsymbol{W}^\top \boldsymbol{A}\mathrm{Diag}(\boldsymbol{w})^{1/2}$ is an orthogonal matrix.

$$\begin{aligned}\mathcal{T} = \mathcal{M}_3(\boldsymbol{W}, \boldsymbol{W}, \boldsymbol{W}) &= \sum_h w_h^{-1/2} (\boldsymbol{W}^\top \boldsymbol{a}_h \sqrt{w_h})^{\otimes 3} \\ &= \sum_h \lambda_h \boldsymbol{v}_h {\otimes}^3, \quad \lambda_h := w_h^{-1/2}.\end{aligned}$$

$\mathcal{T}$ is an orthogonal tensor.

Initial Ideas

In many applications, we estimate moments of the form

$$\mathcal{M}_3 = \sum_h w_h \boldsymbol{a}_h {\otimes}^3,$$

where $\{\boldsymbol{a}_h\}_{h=1}^K$ are assumed to be linearly independent.

What if $\{\boldsymbol{a}_h\}_{h=1}^K$ has orthonormal columns?

$$\boxed{\mathcal{M}_3(I, \boldsymbol{a}_i, \boldsymbol{a}_i) = \sum_h w_h \langle \boldsymbol{a}_h, \boldsymbol{a}_i \rangle^2 \boldsymbol{a}_h = w_i \boldsymbol{a}_i, \ \forall i.}$$

- Analogous to matrix eigenvectors: $\boldsymbol{M}\boldsymbol{v} = \boldsymbol{M}(\boldsymbol{I}, \boldsymbol{v}) = \lambda \boldsymbol{v}$.
- Define orthonormal $\{\boldsymbol{a}_h\}_{h=1}^K$ as eigenvectors of tensor $\mathcal{M}_3$.

Two Problems

- $\{\boldsymbol{a}_h\}_{h=1}^K$ is not orthogonal in general.
- **How to find eigenvectors of a tensor?**

Review: Orthogonal Matrix Eigen Decomposition

Task: Given matrix $\boldsymbol{M} = \sum_{h=1}^{K} \lambda_h \boldsymbol{v}_h \otimes \boldsymbol{v}_h$ with orthonormal components $\{\boldsymbol{v}_h\}_{h=1}^{K}$ ($\boldsymbol{v}_h \perp \boldsymbol{v}_{h'}$, $\forall\, h \neq h'$), find the components/eigenvectors.

Review: Orthogonal Matrix Eigen Decomposition

Task: Given matrix $\boldsymbol{M} = \sum_{h=1}^{K} \lambda_h \boldsymbol{v}_h \otimes \boldsymbol{v}_h$ with orthonormal components $\{\boldsymbol{v}_h\}_{h=1}^{K}$ ($\boldsymbol{v}_h \perp\!\!\!\perp \boldsymbol{v}_{h'}$, $\forall\, h \neq h'$), find the components/eigenvectors.

Properties of Matrix Eigenvectors

- Fixed point: linear transform $\boldsymbol{M}(\boldsymbol{I}, \boldsymbol{v}_i) = \sum_h \lambda_h \langle \boldsymbol{v}_i, \boldsymbol{v}_h \rangle \boldsymbol{v}_h = \lambda_i \boldsymbol{v}_i$

Review: Orthogonal Matrix Eigen Decomposition

Task: Given matrix $\boldsymbol{M} = \sum_{h=1}^{K} \lambda_h \boldsymbol{v}_h \otimes \boldsymbol{v}_h$ with orthonormal components $\{\boldsymbol{v}_h\}_{h=1}^{K}$ ($\boldsymbol{v}_h \perp\!\!\!\perp \boldsymbol{v}_{h'}$, $\forall\, h \neq h'$), find the components/eigenvectors.

Properties of Matrix Eigenvectors

- Fixed point: linear transform $\boldsymbol{M}(\boldsymbol{I}, \boldsymbol{v}_i) = \sum_h \lambda_h \langle \boldsymbol{v}_i, \boldsymbol{v}_h \rangle \boldsymbol{v}_h = \lambda_i \boldsymbol{v}_i$

Intuitions for Matrix Power Method

Review: Orthogonal Matrix Eigen Decomposition

Task: Given matrix $\boldsymbol{M} = \sum_{h=1}^{K} \lambda_h \boldsymbol{v}_h \otimes \boldsymbol{v}_h$ with orthonormal components $\{\boldsymbol{v}_h\}_{h=1}^{K}$ ($\boldsymbol{v}_h \perp\!\!\!\perp \boldsymbol{v}_{h'}, \forall\, h \neq h'$), find the components/eigenvectors.

Properties of Matrix Eigenvectors

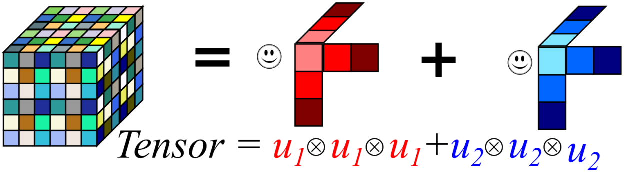
- Fixed point: linear transform $\boldsymbol{M}(\boldsymbol{I}, \boldsymbol{v}_i) = \sum_h \lambda_h \langle \boldsymbol{v}_i, \boldsymbol{v}_h \rangle \boldsymbol{v}_h = \lambda_i \boldsymbol{v}_i$

Intuitions for Matrix Power Method

Linear transform on eigenvectors $\{\boldsymbol{v}_h\}_{h=1}^{K}$ preserve direction

Orthogonal Tensor Eigen Decomposition

Task: Given tensor $\mathcal{T} = \sum_{h=1}^{K} \lambda_h \boldsymbol{v}_h^{\otimes 3}$ with orthonormal components $\{\boldsymbol{v}_h\}_{h=1}^{K}$ ($\boldsymbol{v}_h \perp\!\!\!\perp \boldsymbol{v}_{h'}, \forall\, h \neq h'$), find the components/eigenvectors.

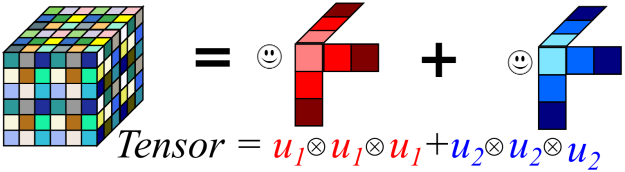


Orthogonal Tensor Eigen Decomposition

Task: Given tensor $\mathcal{T} = \sum_{h=1}^{K} \lambda_h \boldsymbol{v}_h^{\otimes 3}$ with orthonormal components $\{\boldsymbol{v}_h\}_{h=1}^{K}$ ($\boldsymbol{v}_h \perp\!\!\!\perp \boldsymbol{v}_{h'}, \forall\, h \neq h'$), find the components/eigenvectors.

Properties of Tensor Eigenvectors

- Fixed point: bi-linear transform
  $\mathcal{T}(\boldsymbol{I}, \boldsymbol{v}_i, \boldsymbol{v}_i) = \sum_h \lambda_h \langle \boldsymbol{v}_i, \boldsymbol{v}_h \rangle^2 \boldsymbol{v}_h = \lambda_i \boldsymbol{v}_i$

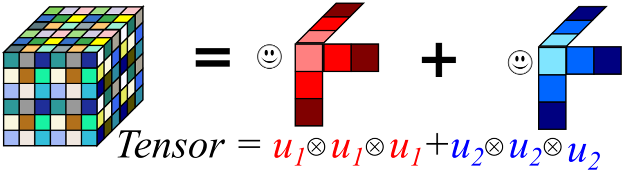


Orthogonal Tensor Eigen Decomposition

Task: Given tensor $\mathcal{T} = \sum_{h=1}^{K} \lambda_h \boldsymbol{v}_h {\otimes}^3$ with orthonormal components $\{\boldsymbol{v}_h\}_{h=1}^{K}$ ($\boldsymbol{v}_h \perp\!\!\!\perp \boldsymbol{v}_{h'}$, $\forall\, h \neq h'$), find the components/eigenvectors.

Properties of Tensor Eigenvectors

- Fixed point: bi-linear transform
  $\mathcal{T}(\boldsymbol{I}, \boldsymbol{v}_i, \boldsymbol{v}_i) = \sum_h \lambda_h \langle \boldsymbol{v}_i, \boldsymbol{v}_h \rangle^2 \boldsymbol{v}_h = \lambda_i \boldsymbol{v}_i$



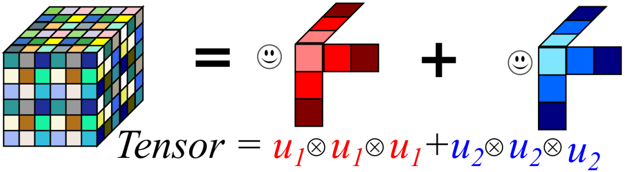
Intuitions for Tensor Power Method

Orthogonal Tensor Eigen Decomposition

Task: Given tensor $\boldsymbol{\mathcal{T}} = \sum_{h=1}^{K} \lambda_h \boldsymbol{v}_h{\otimes}^3$ with orthonormal components $\{\boldsymbol{v}_h\}_{h=1}^{K}$ ($\boldsymbol{v}_h \perp\!\!\!\perp \boldsymbol{v}_{h'}, \forall\, h \neq h'$), find the components/eigenvectors.

Properties of Tensor Eigenvectors

- Fixed point: bi-linear transform
  $\boldsymbol{\mathcal{T}}(\boldsymbol{I}, \boldsymbol{v}_i, \boldsymbol{v}_i) = \sum_h \lambda_h \langle \boldsymbol{v}_i, \boldsymbol{v}_h\rangle^2 \boldsymbol{v}_h = \lambda_i \boldsymbol{v}_i$



Intuitions for Tensor Power Method

Bilinear transform on eigenvectors $\{\boldsymbol{v}_h\}_{h=1}^{K}$ preserve direction

Orthogonal Matrix Eigen Decomposition

Task: Given matrix $\boldsymbol{M} = \sum_{h=1}^{K} \lambda_h \boldsymbol{v}_h {\otimes}^2$ with orthonormal components $\{\boldsymbol{v}_h\}_{h=1}^{K}$ ($\boldsymbol{v}_h \perp \boldsymbol{v}_{h'}$, $\forall\, h \neq h'$), find the components/eigenvectors.

Algorithm Matrix Power Method

Require: Matrix $\boldsymbol{M} \in \mathbb{R}^{K \times K}$

Ensure: Components $\{\widehat{\boldsymbol{v}}_h\}_{h=1}^{K} \overset{\text{w.h.p.}}{=} \{\boldsymbol{v}_h\}_{h=1}^{K}$

1: **for** $h = 1 : K$ **do**
2: Sample $\boldsymbol{u}_0$ uniformly at random from S^{K-1}
3: **for** $i = 1 : T$ **do**
4: $\boldsymbol{u}_i \leftarrow \frac{\boldsymbol{M}(\boldsymbol{I}, \boldsymbol{u}_{i-1})}{\|\boldsymbol{M}(\boldsymbol{I}, \boldsymbol{u}_{i-1})\|}$
5: **end for**
6: $\widehat{\boldsymbol{v}}_h \leftarrow \boldsymbol{u}_T$, $\widehat{\lambda}_h \leftarrow \boldsymbol{M}(\widehat{\boldsymbol{v}}_h, \widehat{\boldsymbol{v}}_h)$
7: Deflate $\boldsymbol{M} \leftarrow \boldsymbol{M} - \widehat{\lambda}_h \widehat{\boldsymbol{v}}_h {\otimes}^2$
8: **end for**

Orthogonal Matrix Eigen Decomposition

Task: Given matrix $\boldsymbol{M} = \sum_{h=1}^{K} \lambda_h \boldsymbol{v}_h \otimes^2$ with orthonormal components $\{\boldsymbol{v}_h\}_{h=1}^{K}$ ($\boldsymbol{v}_h \perp \boldsymbol{v}_{h'}$, $\forall\, h \neq h'$), find the components/eigenvectors.

Algorithm Matrix Power Method

Require: Matrix $\boldsymbol{M} \in \mathbb{R}^{K \times K}$

Ensure: Components $\{\widehat{\boldsymbol{v}}_h\}_{h=1}^{K} \overset{\text{w.h.p.}}{=} \{\boldsymbol{v}_h\}_{h=1}^{K}$

1: **for** $h = 1 : K$ **do**
2: Sample $\boldsymbol{u}_0$ uniformly at random from S^{K-1}
3: **for** $i = 1 : T$ **do**
4: $\boldsymbol{u}_i \leftarrow \frac{\boldsymbol{M}(\boldsymbol{I}, \boldsymbol{u}_{i-1})}{\|\boldsymbol{M}(\boldsymbol{I}, \boldsymbol{u}_{i-1})\|}$
5: **end for**
6: $\widehat{\boldsymbol{v}}_h \leftarrow \boldsymbol{u}_T$, $\widehat{\lambda}_h \leftarrow \boldsymbol{M}(\widehat{\boldsymbol{v}}_h, \widehat{\boldsymbol{v}}_h)$
7: Deflate $\boldsymbol{M} \leftarrow \boldsymbol{M} - \widehat{\lambda}_h \widehat{\boldsymbol{v}}_h \otimes^2$
8: **end for**

Consistency of Matrix Power Method?

Is there convergence? $\{\widehat{\boldsymbol{v}}_h\}_{h=1}^{K} \equiv \{\boldsymbol{v}_h\}_{h=1}^{K}$ w.h.p.?

Orthogonal Matrix Eigen Decomposition

Task: Given matrix $\boldsymbol{M} = \sum_{h=1}^{K} \lambda_h \boldsymbol{v}_h {\otimes}^2$ with orthonormal components $\{\boldsymbol{v}_h\}_{h=1}^{K}$ ($\boldsymbol{v}_h \perp \boldsymbol{v}_{h'}, \forall\, h \neq h'$), find the components/eigenvectors.

Algorithm Matrix Power Method

Require: Matrix $\boldsymbol{M} \in \mathbb{R}^{K \times K}$

Ensure: Components $\{\widehat{\boldsymbol{v}}_h\}_{h=1}^{K} \overset{\text{w.h.p.}}{=} \{\boldsymbol{v}_h\}_{h=1}^{K}$

1: **for** $h = 1 : K$ **do**
2: Sample $\boldsymbol{u}_0$ uniformly at random from S^{K-1}
3: **for** $i = 1 : T$ **do**
4: $\boldsymbol{u}_i \leftarrow \frac{\boldsymbol{M}(\boldsymbol{I}, \boldsymbol{u}_{i-1})}{\|\boldsymbol{M}(\boldsymbol{I}, \boldsymbol{u}_{i-1})\|}$
5: **end for**
6: $\widehat{\boldsymbol{v}}_h \leftarrow \boldsymbol{u}_T$, $\widehat{\lambda}_h \leftarrow \boldsymbol{M}(\widehat{\boldsymbol{v}}_h, \widehat{\boldsymbol{v}}_h)$
7: Deflate $\boldsymbol{M} \leftarrow \boldsymbol{M} - \widehat{\lambda}_h \widehat{\boldsymbol{v}}_h {\otimes}^2$
8: **end for**

Consistency of Matrix Power Method?

Is there convergence? $\{\widehat{\boldsymbol{v}}_h\}_{h=1}^{K} \equiv \{\boldsymbol{v}_h\}_{h=1}^{K}$ w.h.p.?
Does the convergence depend on initialization?

Orthogonal Tensor Eigen Decomposition

Task: Given tensor $\mathcal{T} = \sum_{h=1}^{K} \lambda_h \boldsymbol{v}_h^{\otimes 3}$ with orthonormal components $\{\boldsymbol{v}_h\}_{h=1}^{K}$ ($\boldsymbol{v}_h \perp \boldsymbol{v}_{h'}, \forall\, h \neq h'$), find the components/eigenvectors.

Algorithm Tensor Power Method

Require: Tensor $\mathcal{T} \in \mathbb{R}^{K \times K \times K}$

Ensure: Components $\{\widehat{\boldsymbol{v}}_h\}_{h=1}^{K} \overset{\text{w.h.p.}}{=} \{\boldsymbol{v}_h\}_{h=1}^{K}$

1: **for** $h = 1 : K$ **do**
2: Sample $\boldsymbol{u}_0$ uniformly at random from S^{K-1}
3: **for** $i = 1 : T$ **do**
4: $\boldsymbol{u}_i \leftarrow \frac{\mathcal{T}(\boldsymbol{I}, \boldsymbol{u}_{i-1}, \boldsymbol{u}_{i-1})}{\|\mathcal{T}(\boldsymbol{I}, \boldsymbol{u}_{i-1}, \boldsymbol{u}_{i-1})\|}$
5: **end for**
6: $\widehat{\boldsymbol{v}}_h \leftarrow \boldsymbol{u}_T$, $\widehat{\lambda}_h \leftarrow \mathcal{T}(\widehat{\boldsymbol{v}}_h, \widehat{\boldsymbol{v}}_h, \widehat{\boldsymbol{v}}_h)$
7: Deflate $\mathcal{T} \leftarrow \mathcal{T} - \widehat{\lambda}_h \widehat{\boldsymbol{v}}_h^{\otimes 3}$
8: **end for**

Orthogonal Tensor Eigen Decomposition

Task: Given tensor $\mathcal{T} = \sum_{h=1}^{K} \lambda_h \boldsymbol{v}_h {\otimes}^3$ with orthonormal components $\{\boldsymbol{v}_h\}_{h=1}^{K}$ ($\boldsymbol{v}_h \perp \boldsymbol{v}_{h'}, \forall\, h \neq h'$), find the components/eigenvectors.

Algorithm Tensor Power Method

Require: Tensor $\mathcal{T} \in \mathbb{R}^{K\times K\times K}$

Ensure: Components $\{\widehat{\boldsymbol{v}}_h\}_{h=1}^{K} \overset{\text{w.h.p.}}{=} \{\boldsymbol{v}_h\}_{h=1}^{K}$

1: **for** $h = 1 : K$ **do**
2: Sample $\boldsymbol{u}_0$ uniformly at random from S^{K-1}
3: **for** $i = 1 : T$ **do**
4: $\boldsymbol{u}_i \leftarrow \frac{\mathcal{T}(\boldsymbol{I}, \boldsymbol{u}_{i-1}, \boldsymbol{u}_{i-1})}{\|\mathcal{T}(\boldsymbol{I}, \boldsymbol{u}_{i-1}, \boldsymbol{u}_{i-1})\|}$
5: **end for**
6: $\widehat{\boldsymbol{v}}_h \leftarrow \boldsymbol{u}_T$, $\widehat{\lambda}_h \leftarrow \mathcal{T}(\widehat{\boldsymbol{v}}_h, \widehat{\boldsymbol{v}}_h, \widehat{\boldsymbol{v}}_h)$
7: Deflate $\mathcal{T} \leftarrow \mathcal{T} - \widehat{\lambda}_h \widehat{\boldsymbol{v}}_h {\otimes}^3$
8: **end for**

Consistency of Tensor Power Method?

Is there convergence? $\{\widehat{\boldsymbol{v}}_h\}_{h=1}^{K} \equiv \{\boldsymbol{v}_h\}_{h=1}^{K}$ w.h.p.?
Does the convergence depend on initialization?

Analysis of Consistency of Matrix Power Method

- Order eigenvectors $\{\boldsymbol{v}_h\}_{h=1}^K$ such that corresponding eigenvalues satisfy $\lambda_1 \geq \lambda_2 \ldots \geq \lambda_K$.
- Project initial point $\boldsymbol{u}_0$ onto eigenvectors $\{\boldsymbol{v}_h\}_{h=1}^K$

$$c_h = \langle \boldsymbol{u}_0, \boldsymbol{v}_h \rangle, \ \forall h$$

Convergence properties

- Unique (identifiable) i.f.f. $\{\lambda_h\}_{h=1}^K$ are distinct.
- If gap $\frac{\lambda_2}{\lambda_1} < 1$ and $c_1 \neq 0$, matrix power method converges to $\boldsymbol{v}_1$.
- Converges linearly to $\boldsymbol{v}_1$ assuming gap $\lambda_2/\lambda_1 < 1$.
  - Linear transform permits $\boldsymbol{M}(\boldsymbol{I}, \boldsymbol{u}_0) = \sum_h \lambda_h \left(\boldsymbol{v}_h^\top \boldsymbol{u}_0\right) \boldsymbol{v}_h = \sum_h \lambda_h c_h \boldsymbol{v}_h$, *i.e.*, projection in $\boldsymbol{v}_h$ direction is scaled by λ_h.
  - In t iterations, $\frac{\left(\boldsymbol{v}_1^\top \boldsymbol{v}\right)^2}{\sum_i \left(\boldsymbol{v}_i^\top \boldsymbol{v}\right)^2} \geq 1 - K \left(\frac{\lambda_2}{\lambda_1}\right)^{2t}$.

Analysis of Consistency of Tensor Power Method

- Project initial point $\boldsymbol{u}_0$ onto eigenvectors $c_h = \langle \boldsymbol{u}_0, \boldsymbol{v}_h \rangle, \ \forall h$.
- Order eigenvectors $\{\boldsymbol{v}_h\}_{h=1}^K$ such that

$$\lambda_1 |c_1| > \lambda_2 |c_2| \geq \cdots \geq \lambda_K |c_K|.$$

Convergence properties

- Identifiable i.f.f. $\{\lambda_h |c_h|\}_{h=1}^K$ are distinct. Initialization dependent.
- If $\frac{\lambda_2 |c_2|}{\lambda_1 |c_1|} < 1$ and $\lambda_1 |c_1| \neq 0$, tensor power method converges to $\boldsymbol{v}_1$.
  Note $\boldsymbol{v}_1$ is NOT necessarily the largest eigenvector.
- Converges quadraticly to $\boldsymbol{v}_1$ assuming gap $\frac{\lambda_2 |c_2|}{\lambda_1 |c_1|} < 1$.
  - Bi-linear transform permits $\boldsymbol{\mathcal{T}}(\boldsymbol{I}, \boldsymbol{u}_0, \boldsymbol{u}_0) = \sum_h \lambda_h \left(\boldsymbol{v}_h^\top \boldsymbol{u}_0\right)^2 \boldsymbol{v}_h = \sum_h \lambda_h c_h^2 \ \boldsymbol{v}_h$
    i.e., projection in $\boldsymbol{v}_h$ direction is squared then scaled by λ_h.
  - In t iterations, $\frac{\left(v_1^\top v\right)^2}{\sum_i \left(v_i^\top v\right)^2} \geq 1 - k \left(\frac{\lambda_1}{\max_{i \neq 1} \lambda_i}\right)^2 \left|\frac{v_2 c_2}{v_1 c_1}\right|^{2^{t+1}}$.

Matrix vs. tensor power iteration

Matrix power iteration:

Tensor power iteration:

Matrix vs. tensor power iteration

Matrix power iteration:

1. Requires gap between largest and second-largest eigenvalue.
   Property of the matrix only.

Tensor power iteration:

1. Requires gap between largest and second-largest $\lambda_h |c_h|$.
   Property of the tensor and initialization $\boldsymbol{u}_0$.

Matrix vs. tensor power iteration

Matrix power iteration:

1. Requires gap between largest and second-largest eigenvalue.
   Property of the matrix only.

2. Converges to top eigenvector.

Tensor power iteration:

1. Requires gap between largest and second-largest $\lambda_h |c_h|$.
   Property of the tensor and initialization $\boldsymbol{u}_0$.

2. Converges to v_i which is the largest $v_h |c_h|$. Not necessarily the largest eigenvector.

Matrix vs. tensor power iteration

Matrix power iteration:

1. Requires gap between largest and second-largest eigenvalue.
   Property of the matrix only.

2. Converges to top eigenvector.

3. Linear convergence. Need $O(\log(1/\epsilon))$ iterations.

Tensor power iteration:

1. Requires gap between largest and second-largest $\lambda_h |c_h|$.
   Property of the tensor and initialization $\boldsymbol{u}_0$.

2. Converges to v_i which is the largest $v_h |c_h|$. Not necessarily the largest eigenvector.

3. Quadratic convergence. Need $O(\log\log(1/\epsilon))$ iterations.

Spurious Eigenvectors for Tensor Eigen Decomposition

$$\mathcal{T} = \sum_{h \in [K]} \lambda_h \boldsymbol{v}_h \otimes^3 .$$

Characterization of eigenvectors: $\mathcal{T}(\boldsymbol{I}, \boldsymbol{v}, \boldsymbol{v}) = \lambda \boldsymbol{v}$?

- $\{\boldsymbol{v}_h\}_{h=1}^K$ are eigenvectors as $\mathcal{T}(\boldsymbol{I}, \boldsymbol{v}_h, \boldsymbol{v}_h) = \lambda_h \boldsymbol{v}_h$.

Spurious Eigenvectors for Tensor Eigen Decomposition

$$\mathcal{T} = \sum_{h \in [K]} \lambda_h \boldsymbol{v}_h \otimes^3 .$$

Characterization of eigenvectors: $\mathcal{T}(\boldsymbol{I}, \boldsymbol{v}, \boldsymbol{v}) = \lambda \boldsymbol{v}$?

- $\{\boldsymbol{v}_h\}_{h=1}^K$ are eigenvectors as $\mathcal{T}(\boldsymbol{I}, \boldsymbol{v}_h, \boldsymbol{v}_h) = \lambda_h \boldsymbol{v}_h$.
- Bad news: There can be other eigenvectors (unlike matrix case).

  E.g., when $\{\lambda_h\}_{h=1}^K \equiv 1$

$$\boldsymbol{v} = \frac{\boldsymbol{v}_1 + \boldsymbol{v}_2}{\sqrt{2}} \text{ satisfies } \mathcal{T}(\boldsymbol{I}, \boldsymbol{v}, \boldsymbol{v}) = \frac{1}{\sqrt{2}} \boldsymbol{v}.$$

Spurious Eigenvectors for Tensor Eigen Decomposition

$$\mathcal{T} = \sum_{h \in [K]} \lambda_h \boldsymbol{v}_h \otimes^3 .$$

Characterization of eigenvectors: $\mathcal{T}(\boldsymbol{I}, \boldsymbol{v}, \boldsymbol{v}) = \lambda \boldsymbol{v}$?

- $\{\boldsymbol{v}_h\}_{h=1}^K$ are eigenvectors as $\mathcal{T}(\boldsymbol{I}, \boldsymbol{v}_h, \boldsymbol{v}_h) = \lambda_h \boldsymbol{v}_h$.
- Bad news: There can be other eigenvectors (unlike matrix case).

  E.g., when $\{\lambda_h\}_{h=1}^K \equiv 1$

$$\boldsymbol{v} = \frac{\boldsymbol{v}_1 + \boldsymbol{v}_2}{\sqrt{2}} \text{ satisfies } \mathcal{T}(\boldsymbol{I}, \boldsymbol{v}, \boldsymbol{v}) = \frac{1}{\sqrt{2}} \boldsymbol{v}.$$

How do we avoid spurious solutions (not components $\{\boldsymbol{v}_h\}_{h=1}^K$)?

Spurious Eigenvectors for Tensor Eigen Decomposition

$$\mathcal{T} = \sum_{h \in [K]} \lambda_h \boldsymbol{v}_h \otimes^3 .$$

Characterization of eigenvectors: $\mathcal{T}(\boldsymbol{I}, \boldsymbol{v}, \boldsymbol{v}) = \lambda \boldsymbol{v}$?

- $\{\boldsymbol{v}_h\}_{h=1}^K$ are eigenvectors as $\mathcal{T}(\boldsymbol{I}, \boldsymbol{v}_h, \boldsymbol{v}_h) = \lambda_h \boldsymbol{v}_h$.
- Bad news: There can be other eigenvectors (unlike matrix case).

  E.g., when $\{\lambda_h\}_{h=1}^K \equiv 1$

  $$\boldsymbol{v} = \frac{\boldsymbol{v}_1 + \boldsymbol{v}_2}{\sqrt{2}} \text{ satisfies } \mathcal{T}(\boldsymbol{I}, \boldsymbol{v}, \boldsymbol{v}) = \frac{1}{\sqrt{2}} \boldsymbol{v}.$$

How do we avoid spurious solutions (not components $\{\boldsymbol{v}_h\}_{h=1}^K$)?

Optimization viewpoint of tensor Eigen decomposition will help.

Spurious Eigenvectors for Tensor Eigen Decomposition

$$\mathcal{T} = \sum_{h \in [K]} \lambda_h \boldsymbol{v}_h \otimes^3 .$$

Characterization of eigenvectors: $\mathcal{T}(\boldsymbol{I}, \boldsymbol{v}, \boldsymbol{v}) = \lambda \boldsymbol{v}$?

- $\{\boldsymbol{v}_h\}_{h=1}^K$ are eigenvectors as $\mathcal{T}(\boldsymbol{I}, \boldsymbol{v}_h, \boldsymbol{v}_h) = \lambda_h \boldsymbol{v}_h$.
- Bad news: There can be other eigenvectors (unlike matrix case).

  E.g., when $\{\lambda_h\}_{h=1}^K \equiv 1$

$$\boldsymbol{v} = \frac{\boldsymbol{v}_1 + \boldsymbol{v}_2}{\sqrt{2}} \text{ satisfies } \mathcal{T}(\boldsymbol{I}, \boldsymbol{v}, \boldsymbol{v}) = \frac{1}{\sqrt{2}} \boldsymbol{v}.$$

How do we avoid spurious solutions (not components $\{\boldsymbol{v}_h\}_{h=1}^K$)?

Optimization viewpoint of tensor Eigen decomposition will help.

All spurious eigenvectors are saddle points.

Optimization Viewpoint of Matrix/Tensor Eigen Decomposition

Optimization Viewpoint of Matrix/Tensor Eigen Decomposition

Optimization Problem

Matrix: $\max_v M(v,v)$ s.t. $\|v\| = 1$.

- Lagrangian:
  $L(v,\lambda) := M(v,v) - \lambda(v^\top v - 1)$.

Tensor: $\max_v T(v,v,v)$ s.t. $\|v\| = 1$.

- Lagrangian:
  $L(v,\lambda) := T(v,v,v) - 1.5\lambda(v^\top v - 1)$.

Optimization Viewpoint of Matrix/Tensor Eigen Decomposition

Optimization Problem

Matrix: $\max_v M(v, v)$ s.t. $\|v\| = 1$.

- Lagrangian:
  $L(v, \lambda) := M(v, v) - \lambda(v^\top v - 1)$.

Tensor: $\max_v T(v, v, v)$ s.t. $\|v\| = 1$.

- Lagrangian:
  $L(v, \lambda) := T(v, v, v) - 1.5\lambda(v^\top v - 1)$.

Non-convex: stationary points = {global optima, local optima, saddle point}

Optimization Viewpoint of Matrix/Tensor Eigen Decomposition

Optimization Problem

Matrix: $\max_v M(v,v)$ s.t. $\|v\| = 1$.

- Lagrangian: $L(v,\lambda) := M(v,v) - \lambda(v^\top v - 1)$.

Tensor: $\max_v T(v,v,v)$ s.t. $\|v\| = 1$.

- Lagrangian: $L(v,\lambda) := T(v,v,v) - 1.5\lambda(v^\top v - 1)$.

Non-convex: stationary points $=$ {global optima, local optima, saddle point}

Stationary Points: first derivative $\boxed{\nabla L(v,\lambda) = 0}$

$\nabla L(v,\lambda) = 2(M(I,v) - \lambda v) = 0$

- Eigenvectors are stationary points.
- Power method $v \leftarrow \frac{M(I,v)}{\|M(I,v)\|}$ is a version of gradient ascent.

$\nabla L(v,\lambda) = 3(T(I,v,v) - \lambda v) = 0$

- Eigenvectors are stationary points.
- Power method $v \leftarrow \frac{T(I,v,v)}{\|T(I,v,v)\|}$ is a version of gradient ascent.

Optimization Viewpoint of Matrix/Tensor Eigen Decomposition

Optimization Problem

Matrix: $\max_v M(v,v)$ s.t. $\|v\| = 1$.

- Lagrangian: $L(v,\lambda) := M(v,v) - \lambda(v^\top v - 1)$.

Tensor: $\max_v T(v,v,v)$ s.t. $\|v\| = 1$.

- Lagrangian: $L(v,\lambda) := T(v,v,v) - 1.5\lambda(v^\top v - 1)$.

Non-convex: stationary points $=$ {global optima, local optima, saddle point}

Stationary Points: first derivative $\boxed{\nabla L(v,\lambda) = 0}$

$\nabla L(v,\lambda) = 2(M(I,v) - \lambda v) = 0$

- Eigenvectors are stationary points.
- Power method $v \leftarrow \frac{M(I,v)}{\|M(I,v)\|}$ is a version of gradient ascent.

$\nabla L(v,\lambda) = 3(T(I,v,v) - \lambda v) = 0$

- Eigenvectors are stationary points.
- Power method $v \leftarrow \frac{T(I,v,v)}{\|T(I,v,v)\|}$ is a version of gradient ascent.

Local Optima: $\boxed{w^\top \nabla^2 L(v,\lambda) w < 0}$ for all $w \perp v$, at a stationary point v

- $\boldsymbol{v}_1$ is the only local optimum.
- All other eigenvectors are saddle points.

- $\{\boldsymbol{v}_h\}_{h=1}^K$ are the only local optima.
- All spurious eigenvectors are saddle points.

Question: What about performance under noise?

Tensor Perturbation Analysis

$$\hat{\mathcal{T}} = \mathcal{T} + \mathcal{E}, \quad \mathcal{T} = \sum_h \lambda_h \boldsymbol{v}_h \otimes^3, \quad \|\mathcal{E}\| := \max_{\boldsymbol{x}:\|\boldsymbol{x}\|=1} |\mathcal{E}(\boldsymbol{x}, \boldsymbol{x}, \boldsymbol{x})| \leq \epsilon.$$

Tensor Perturbation Analysis

$$\hat{\mathcal{T}} = \mathcal{T} + \mathcal{E}, \quad \mathcal{T} = \sum_h \lambda_h v_h \otimes^3, \quad \|\mathcal{E}\| := \max_{x:\|x\|=1} |\mathcal{E}(x,x,x)| \leq \epsilon.$$

Theorem: Let T be number of iterations. If

$$\boxed{T \geq \log K + \log\log \frac{\lambda_{\max}}{\epsilon}, \quad \epsilon < \frac{\lambda_{\min}}{K},}$$

then output (v, λ) (after polynomial restarts) satisfies

$$\|v - v_1\| \leq O\left(\frac{\epsilon}{\lambda_1}\right), \quad \|\lambda - \lambda_1\| \leq O(\epsilon),$$

where v_1 is s.t. $\lambda_1|c_1| > \lambda_2|c_2| \ldots, \quad c_i := \langle v_i, u_0 \rangle$, and u_0 is the (successful) initializer.

Tensor Perturbation Analysis

$$\hat{\mathcal{T}} = \mathcal{T} + \mathcal{E}, \quad \mathcal{T} = \sum_h \lambda_h \boldsymbol{v}_h {\otimes}^3, \quad \|\mathcal{E}\| := \max_{\boldsymbol{x}:\|\boldsymbol{x}\|=1} |\mathcal{E}(\boldsymbol{x},\boldsymbol{x},\boldsymbol{x})| \le \epsilon.$$

Theorem: Let T be number of iterations. If

$$\boxed{T \ge \log K + \log\log \frac{\lambda_{\max}}{\epsilon}, \quad \epsilon < \frac{\lambda_{\min}}{K},}$$

then output $(\boldsymbol{v}, \lambda)$ (after polynomial restarts) satisfies

$$\|\boldsymbol{v} - \boldsymbol{v}_1\| \le O\left(\frac{\epsilon}{\lambda_1}\right), \quad \|\lambda - \lambda_1\| \le O(\epsilon),$$

where $\boldsymbol{v}_1$ is s.t. $\lambda_1|c_1| > \lambda_2|c_2| \ldots, \quad c_i := \langle \boldsymbol{v}_i, \boldsymbol{u}_0 \rangle$, and $\boldsymbol{u}_0$ is the (successful) initializer.

- Careful analysis of deflation: avoid buildup of errors.
- Implies polynomial sample complexity for learning.

Other tensor decomposition techniques

Orthogonal Tensor Decomposition

Simultaneous Power Method

- (Wang & Lu, 2017)

  Simultaneous recovery of eigenvectors
  Initialization is not optimal

Orthogonalized Simultaneous Alternating Least Square

- (Sharan & Valiant, 2017)

  Random initialization
  Proved convergence for symmetric tensor

Initialization

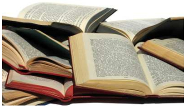
- SVD based initialization (Anandkumar & Janzamin, 2014).
- State-of-the-art (trace based) initialization (Li & Huang, 2018).

Outline

Neural Network - Nonlinear Function Approximation

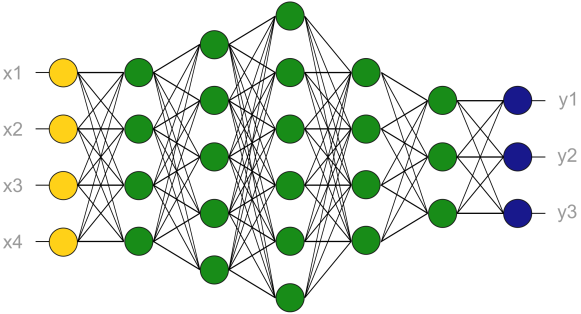
Image classification

Speech recognition



Text processing

Success of Deep Neural Networks



- computation power growth
- enormous labeled data

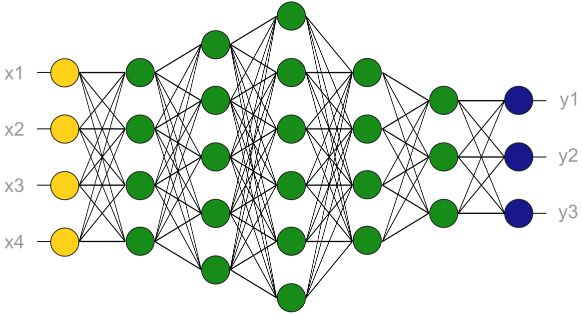
Neural Network - Nonlinear Function Approximation

Image classification

Speech recognition

Text processing

Success of Deep Neural Networks

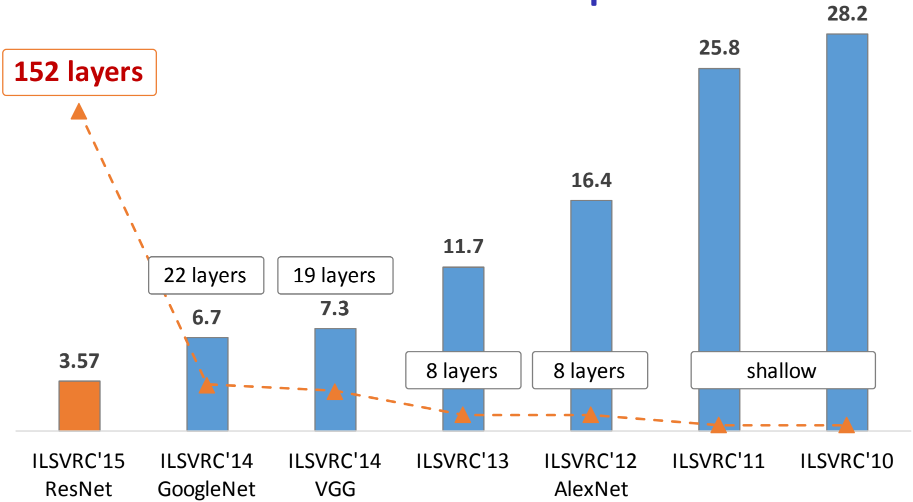


- computation power growth
- enormous labeled data

Express Power

- linear composition vs nonlinear composition
- shallow network vs deep structure

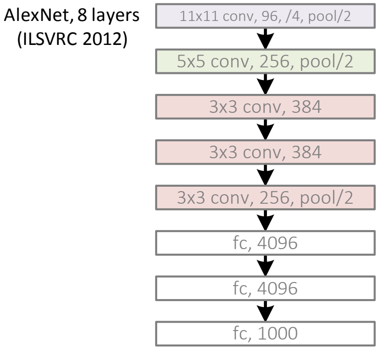
Revolution of Depth



ImageNet Classification top-5 error (%)

Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". CVPR 2016.

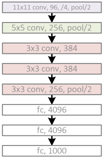
Revolution of Depth



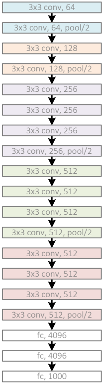
Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. “Deep Residual Learning for Image Recognition”. CVPR 2016.

Revolution of Depth

AlexNet, 8 layers
(ILSVRC 2012)



VGG, 19 layers
(ILSVRC 2014)



GoogleNet, 22 layers
(ILSVRC 2014)

Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. “Deep Residual Learning for Image Recognition”. CVPR 2016.

Revolution of Depth

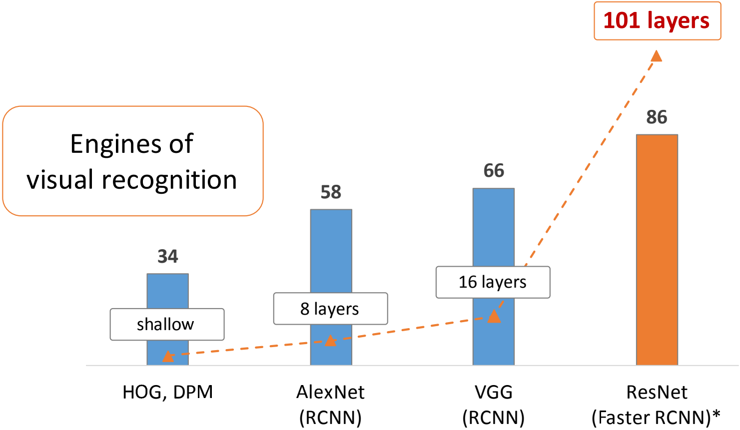
AlexNet, 8 layers
(ILSVRC 2012)

VGG, 19 layers
(ILSVRC 2014)

ResNet, 152 layers
(ILSVRC 2015)

Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". CVPR 2016.

Revolution of Depth



PASCAL VOC 2007 **Object Detection** mAP (%)

*w/ other improvements & more data

Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. “Deep Residual Learning for Image Recognition”. CVPR 2016.

Challenges For Large Deep Neural Network

Learning

- Learning takes longer, might not converge, susceptible to vanishing/exploding gradients, etc
- One-time cost.

Challenges For Large Deep Neural Network

Learning

- Learning takes longer, might not converge, susceptible to vanishing/exploding gradients, etc
- One-time cost.

Test

- Requires large amount of computation and memory storage.
  Ill-suited for smart phones or IoT device.
- Repeated cost.

Challenges For Large Deep Neural Network

Learning

- Learning takes longer, might not converge, susceptible to vanishing/exploding gradients, etc
- One-time cost.

Test

- Requires large amount of computation and memory storage.

  Ill-suited for smart phones or IoT device.
- Repeated cost.

How to compress the neural network without much performance loss?

Common Types of Tensor Decompositions

m-order tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{I_0 \times I_1 \times \cdots \times I_{m-1}}$

Common Types of Tensor Decompositions

m-order tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{I_0 \times I_1 \times \cdots \times I_{m-1}}$

CANDECOMP/PARAFAC (CP) Decomposition

- Factorize a tensor into sum of rank-1 tensors
- Rank-1 tensor is defined as outer product of multiple vectors
- $\boldsymbol{\mathcal{T}}_{i_0, \cdots, i_{m-1}} = \sum_{r=0}^{R-1} \boldsymbol{M}_{r,i_0}^{(0)} \cdots \boldsymbol{M}_{r,i_{m-1}}^{(m-1)}$

Common Types of Tensor Decompositions

m-order tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{I_0 \times I_1 \times \cdots \times I_{m-1}}$

CANDECOMP/PARAFAC (CP) Decomposition

- Factorize a tensor into sum of rank-1 tensors
- Rank-1 tensor is defined as outer product of multiple vectors
- $\boldsymbol{\mathcal{T}}_{i_0,\cdots,i_{m-1}} = \sum_{r=0}^{R-1} \boldsymbol{M}_{r,i_0}^{(0)} \cdots \boldsymbol{M}_{r,i_{m-1}}^{(m-1)}$

Tucker (TK) Decomposition

- More general than CP decomposition
- Multilinear operation on a core tensor $\boldsymbol{\mathcal{C}}$: $\boldsymbol{\mathcal{C}}(\boldsymbol{M}^{(0)}, \ldots, \boldsymbol{M}^{(m-1)})$
- $\boldsymbol{\mathcal{T}}_{i_0,\cdots,i_{m-1}} = \sum_{r_0=0}^{R_0-1} \cdots \sum_{r_{m-1}=0}^{R_{m-1}-1} \boldsymbol{\mathcal{C}}_{r_0,\ldots,r_{m-1}} \boldsymbol{M}_{r_0,i_0}^{(0)} \cdots \boldsymbol{M}_{r_{m-1},i_{m-1}}^{(m-1)}$

Common Types of Tensor Decompositions

m-order tensor $\boldsymbol{\mathcal{T}} \in \mathbb{R}^{I_0 \times I_1 \times \cdots \times I_{m-1}}$

CANDECOMP/PARAFAC (CP) Decomposition

- Factorize a tensor into sum of rank-1 tensors
- Rank-1 tensor is defined as outer product of multiple vectors
- $\boldsymbol{\mathcal{T}}_{i_0,\cdots,i_{m-1}} = \sum_{r=0}^{R-1} \boldsymbol{M}^{(0)}_{r,i_0} \cdots \boldsymbol{M}^{(m-1)}_{r,i_{m-1}}$

Tucker (TK) Decomposition

- More general than CP decomposition
- Multilinear operation on a core tensor $\boldsymbol{\mathcal{C}}$: $\boldsymbol{\mathcal{C}}(\boldsymbol{M}^{(0)}, \ldots, \boldsymbol{M}^{(m-1)})$
- $\boldsymbol{\mathcal{T}}_{i_0,\cdots,i_{m-1}} = \sum_{r_0=0}^{R_0-1} \cdots \sum_{r_{m-1}=0}^{R_{m-1}-1} \boldsymbol{\mathcal{C}}_{r_0,\ldots,r_{m-1}} \boldsymbol{M}^{(0)}_{r_0,i_0} \cdots \boldsymbol{M}^{(m-1)}_{r_{m-1},i_{m-1}}$

Tensor-Train (TT) Decomposition

- Factorize a tensor into a number of interconnected lower-order tensors
- $\boldsymbol{\mathcal{T}}_{i_0,\ldots,i_{m-1}} = \sum_{r_0=1}^{R_0-1} \cdots \sum_{r_{m-2}=1}^{R_{m-2}-1} \boldsymbol{\mathcal{T}}^{(0)}_{i_0,r_0} \boldsymbol{\mathcal{T}}^{(1)}_{r_0,i_1,r_1} \cdots \boldsymbol{\mathcal{T}}^{(m-1)}_{r_{m-2},i_{m-1}}$

Compression of Convolutional Layer w/ Tensor Decompositions

Convolutional Kernel: tensor $\mathcal{K} \in \mathbb{R}^{H \times W \times S \times T}$

Compression of Convolutional Layer w/ Tensor Decompositions

Convolutional Kernel: tensor $\mathcal{K} \in \mathbb{R}^{H \times W \times S \times T}$

- Filter height/width H/W, No. of input/output channels S/T.

Compression of Convolutional Layer w/ Tensor Decompositions

Convolutional Kernel: tensor $\mathcal{K} \in \mathbb{R}^{H \times W \times S \times T}$

- Filter height/width H/W, No. of input/output channels S/T.
- Map an input tensor $\mathcal{U} \in \mathbb{R}^{X \times Y \times S}$ to an output tensor $\mathcal{V} \in \mathbb{R}^{X' \times Y' \times T}$.

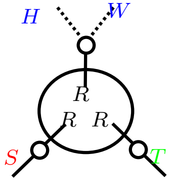
Compression of Convolutional Layer w/ Tensor Decompositions

Convolutional Kernel: tensor $\mathcal{K} \in \mathbb{R}^{H \times W \times S \times T}$

- Filter height/width H/W, No. of input/output channels S/T.
- Map an input tensor $\mathcal{U} \in \mathbb{R}^{X \times Y \times S}$ to an output tensor $\mathcal{V} \in \mathbb{R}^{X' \times Y' \times T}$.

Kernel **CP** Decomposition

- **CP**: Decompose kernel $\mathcal{K}$ into 3 factor tensors
- $\mathcal{K}_{i,j,s,t} = \sum_{r=0}^{R-1} \mathcal{K}^{(0)}_{s,r} \, \mathcal{K}^{(1)}_{i,j,r} \, \mathcal{K}^{(2)}_{r,t}$
- No. of param.: $HWST \to (HW + S + T)R$



CP decomposition

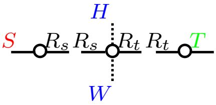
Compression of Convolutional Layer w/ Tensor Decompositions

Convolutional Kernel: tensor $\boldsymbol{\mathcal{K}} \in \mathbb{R}^{H \times W \times S \times T}$

- Filter height/width H/W, No. of input/output channels S/T.
- Map an input tensor $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{X \times Y \times S}$ to an output tensor $\boldsymbol{\mathcal{V}} \in \mathbb{R}^{X' \times Y' \times T}$.

Kernel **TK** Decomposition

- **TK**: Decompose $\boldsymbol{\mathcal{K}}$ into 1 core tensor, 2 factor tensors
- $\boldsymbol{\mathcal{K}}_{i,j,s,t} = \sum_{r_s=0}^{R_s-1} \sum_{r_t=0}^{R_t-1} \boldsymbol{\mathcal{K}}^{(0)}_{s,r_s} \, \boldsymbol{\mathcal{K}}^{(1)}_{i,j,r_s,r_t} \, \boldsymbol{\mathcal{K}}^{(2)}_{r_t,t}$
- No. of param.: $HWST \to SR_s + HWR_sR_t + R_tT$



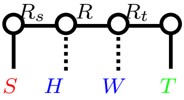
TK decomposition

Compression of Convolutional Layer w/ Tensor Decompositions

Convolutional Kernel: tensor $\mathcal{K} \in \mathbb{R}^{H \times W \times S \times T}$

- Filter height/width H/W, No. of input/output channels S/T.
- Map an input tensor $\mathcal{U} \in \mathbb{R}^{X \times Y \times S}$ to an output tensor $\mathcal{V} \in \mathbb{R}^{X' \times Y' \times T}$.

Kernel **TT** Decomposition



TT decomposition

- **TT**: Decompose $\mathcal{K}$ into 4 factor tensors
- $\mathcal{K}_{i,j,s,t} = \sum_{r_s=0}^{R_s-1} \sum_{r=0}^{R-1} \sum_{r_t=0}^{R_t-1} \mathcal{K}^{(0)}_{s,r_s} \mathcal{K}^{(1)}_{r_s,i,r} \mathcal{K}^{(2)}_{r,j,r_t} \mathcal{K}^{(3)}_{r_t,t}$
- No. of param.: $HWST \rightarrow SR_s + HR_sR + WR_tR + R_tT$

Convolutional Kernel: $\mathcal{K} \in \mathbb{R}^{H \times W \times S \times T}$ **tensorized** to $\mathcal{K}' \in \mathbb{R}^{H \times W \times S_0 \times \cdots \times S_{m-1} \times T_0 \times \cdots \times T_{m-1}}$

Convolutional Kernel: $\mathcal{K} \in \mathbb{R}^{H \times W \times S \times T}$ **tensorized** to $\mathcal{K}' \in \mathbb{R}^{H \times W \times S_0 \times \cdots \times S_{m-1} \times T_0 \times \cdots \times T_{m-1}}$

- Tensorization: kernel reshaped to higher order tensor.

Convolutional Kernel: $\mathcal{K} \in \mathbb{R}^{H \times W \times S \times T}$ **tensorized** to $\mathcal{K}' \in \mathbb{R}^{H \times W \times S_0 \times \cdots \times S_{m-1} \times T_0 \times \cdots \times T_{m-1}}$

- Tensorization: kernel reshaped to higher order tensor.
- $S = \prod_{i=0}^{m-1} S_i$ and $T = \prod_{i=0}^{m-1} T_i$.

Convolutional Kernel: $\mathcal{K} \in \mathbb{R}^{H \times W \times S \times T}$ **tensorized** to $\mathcal{K}' \in \mathbb{R}^{H \times W \times S_0 \times \cdots \times S_{m-1} \times T_0 \times \cdots \times T_{m-1}}$

- Tensorization: kernel reshaped to higher order tensor.
- $S = \prod_{i=0}^{m-1} S_i$ and $T = \prod_{i=0}^{m-1} T_i$.
- Input tensor $\mathcal{U} \in \mathbb{R}^{X \times Y \times S}$ tensorized to $\mathcal{U}' \in \mathbb{R}^{X \times Y \times S_0 \times \cdots \times S_{m-1}}$.

Convolutional Kernel: $\mathcal{K} \in \mathbb{R}^{H \times W \times S \times T}$ **tensorized** to $\mathcal{K}' \in \mathbb{R}^{H \times W \times S_0 \times \cdots \times S_{m-1} \times T_0 \times \cdots \times T_{m-1}}$

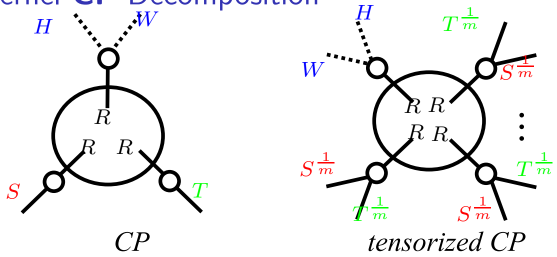
- Tensorization: kernel reshaped to higher order tensor.
- $S = \prod_{i=0}^{m-1} S_i$ and $T = \prod_{i=0}^{m-1} T_i$.
- Input tensor $\mathcal{U} \in \mathbb{R}^{X \times Y \times S}$ tensorized to $\mathcal{U}' \in \mathbb{R}^{X \times Y \times S_0 \times \cdots \times S_{m-1}}$.
- Output reshaped $\mathcal{V} \in \mathbb{R}^{X \times Y \times T}$ to $\mathcal{V}' \in \mathbb{R}^{X' \times Y' \times T_0 \times \cdots \times T_{m-1}}$.

Tensorized Spectrum Preserving Compression of Neural Networks

Convolutional Kernel: $\mathcal{K} \in \mathbb{R}^{H\times W\times S\times T}$ **tensorized** to $\mathcal{K}' \in \mathbb{R}^{H\times W\times S_0\times\cdots\times S_{m-1}\times T_0\times\cdots\times T_{m-1}}$

- Tensorization: kernel reshaped to higher order tensor.
- $S = \prod_{i=0}^{m-1} S_i$ and $T = \prod_{i=0}^{m-1} T_i$.
- Input tensor $\mathcal{U} \in \mathbb{R}^{X\times Y\times S}$ tensorized to $\mathcal{U}' \in \mathbb{R}^{X\times Y\times S_0\times\cdots\times S_{m-1}}$.
- Output reshaped $\mathcal{V} \in \mathbb{R}^{X\times Y\times T}$ to $\mathcal{V}' \in \mathbb{R}^{X'\times Y'\times T_0\times\cdots\times T_{m-1}}$.

Tensorized Kernel **CP** Decomposition



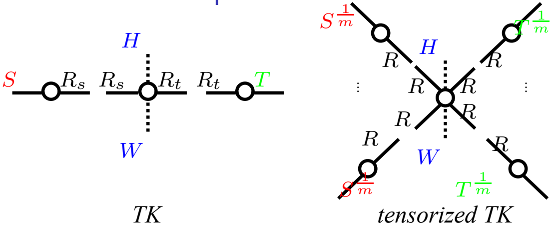
- Param. No.: $HWST \rightarrow (HW + S + T)R \rightarrow (m(ST)^{\frac{1}{m}} + HW)R$

Tensorized Spectrum Preserving Compression of Neural Networks

Convolutional Kernel: $\mathcal{K} \in \mathbb{R}^{H \times W \times S \times T}$ **tensorized** to $\mathcal{K}' \in \mathbb{R}^{H \times W \times S_0 \times \cdots \times S_{m-1} \times T_0 \times \cdots \times T_{m-1}}$

- Tensorization: kernel reshaped to higher order tensor.
- $S = \prod_{i=0}^{m-1} S_i$ and $T = \prod_{i=0}^{m-1} T_i$.
- Input tensor $\mathcal{U} \in \mathbb{R}^{X \times Y \times S}$ tensorized to $\mathcal{U}' \in \mathbb{R}^{X \times Y \times S_0 \times \cdots \times S_{m-1}}$.
- Output reshaped $\mathcal{V} \in \mathbb{R}^{X \times Y \times T}$ to $\mathcal{V}' \in \mathbb{R}^{X' \times Y' \times T_0 \times \cdots \times T_{m-1}}$.

Tensorized Kernel **TK** Decomposition

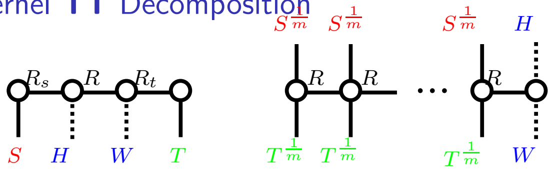


- Param. No.: $HWST \to SR_s + HWR_sR_t + R_tT \to m(S^{\frac{1}{m}} + T^{\frac{1}{m}})R + HWR^{2m}$

Convolutional Kernel: $\mathcal{K} \in \mathbb{R}^{H \times W \times S \times T}$ **tensorized** to $\mathcal{K}' \in \mathbb{R}^{H \times W \times S_0 \times \cdots \times S_{m-1} \times T_0 \times \cdots \times T_{m-1}}$

- Tensorization: kernel reshaped to higher order tensor.
- $S = \prod_{i=0}^{m-1} S_i$ and $T = \prod_{i=0}^{m-1} T_i$.
- Input tensor $\mathcal{U} \in \mathbb{R}^{X \times Y \times S}$ tensorized to $\mathcal{U}' \in \mathbb{R}^{X \times Y \times S_0 \times \cdots \times S_{m-1}}$.
- Output reshaped $\mathcal{V} \in \mathbb{R}^{X \times Y \times T}$ to $\mathcal{V}' \in \mathbb{R}^{X' \times Y' \times T_0 \times \cdots \times T_{m-1}}$.

Tensorized Kernel **TT** Decomposition



TT *tensorized TT*

- Param. No.: $HWST \to SR_s + HR_sR + WR_tR + R_tT \to (m(ST)^{\frac{1}{m}}R + HW)R$

Experiments - Compress CIFAR10 Resnet-34

Successful Compression of CIFAR10 Resnet-34 Network (Su, Li, Bhattacharjee & Huang, 2018)

	Compression rate: SPC, E2E				Compression rate: **t-SPC**, Seq.			
Method	5%	10%	20%	40%	2%	5%	10%	20%
CP	84.02	86.93	88.75	88.75	85.7	89.86	**91.28**	-
TK	83.57	86.00	88.03	89.35	61.06	71.34	81.59	87.11
TT	77.44	82.92	84.13	86.64	78.95	84.26	87.89	-

- Testing accuracies of tensor methods under compression rates.
- The uncompressed network achieves 93.2% accuracy.
- CIFAR10 Resnet-34 has 4×10^5 parameters that have to be trained and retained during testing.

Experiments - Compress ImageNet Resnet-50

Successful Compression of ImageNet Resnet-50 Network (Su, Li, Bhattacharjee & Huang, 2018)

# Epochs	Uncompressed	SPC-TT (E2E)	**t-SPC**-TT (Seq.)
0.2	4.22	0.66x	10.51x
0.3	6.23	0.64x	7.54x
0.5	9.01	0.83x	5.54x
1.0	17.3	0.74x	3.04x
2.0	30.8	0.59x	1.75x

- Testing accuracy of tensor methods compared to the uncompressed ImageNet Resnet-50.
- The accuracy of the tensor method results (both non-tensorized and tensorized) are shown normalized to the uncompressed network's accuracy.

Outline

Conclusion

- Method-of-moments can efficiently estimate parameters for many latent variable models.
    - Exploit distributional properties, multi-view structure, and other structure to determine usable moments tensors.
    - Some efficient algorithms for carrying out the tensor decomposition to obtain parameter estimates.
- Tensor decomposition of neural network kernels/weights effectively compresses the network.
- Many issues to resolve
    - Handle model misspecification, increase robustness.
    - Learning deep neural network parameters using tensor decomposition?

A Short List of Related Papers to Today's Talk

- "A Method of Moments for Mixture Models and Hidden Markov Models", by Anima Anandkumar, Daniel Hsu and Sham Kakade. In Conference on Learning Theory, 2012.
- "Tensor Decompositions for Learning Latent Variable Models", by Anima Anandkumar, Rong Ge, Daniel Hsu, Sham Kakade and Matus Telgarsky. In Journal of Machine Learning Research, 2014.
- "Escaping from saddle pointsonline stochastic gradient for tensor decomposition", Rong Ge, Furong Huang, Chi Jin and Yang Yuan. In Conference on Learning Theory, 2015.
- "Online tensor methods for learning latent variable models", Furong Huang, Niranjan U. N., Mohammad Umar Hakeem and Anima Anandkumar. The Journal of Machine Learning Research, 2016.
- "Guaranteed Simultaneous Asymmetric Tensor Decomposition via Orthogonalized Alternating Least Squares", by Jialin Li and Furong Huang, 2018.
- "Tensorized Spectrum Preserving Compression for Neural Networks", by Jiahao Su, Jingling Li, Bobby Bhattacharjee and Furong Huang, 2018.

Tensor Softwares

- Spark implementation of method of moments to learn Latent Dirichlet Allocation available at https://github.com/FurongHuang/spectrallda-tensorspark.
- Tensorly: Simple and Fast Tensor Learning in Python available at http://tensorly.org/stable/home.html.
- A general library with higher order tensor operations is coming soon.