Guaranteed Learning of Latent Variable Models through Tensor Methods

Furong Huang

University of Maryland

furongh@cs.umd.edu

ACM SIGMETRICS Tutorial 2018
Learning algorithms for latent variable models based on decompositions of moment tensors.

“Method-of-moments” (Pearson, 1894)
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

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.
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

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**

- **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, 11k diseases.

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

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

Unlabeled data → Latent variable model → Tensor Decomposition = $f_1 + f_2 + f_3$ → 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?

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
Gaussian Mixture Model

Generative Model

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

\[
H \sim \text{Cat}(\pi_1, \pi_2, \ldots, \pi_K)
\]
\[
X_{|H=h} \sim \mathcal{N}(\mu_h, \Sigma_h), \quad \forall h \in [K]
\]
Gaussian Mixture Model

Generative Model
- Samples are comprised of $K$ different Gaussians according to $\text{Cat}(\pi_1, \pi_2, \ldots, \pi_K)$
- Each sample is from one of the $K$ Gaussians, $\mathcal{N}(\mu_h, \Sigma_h)$, $\forall h \in [K]$

$$H \sim \text{Cat}(\pi_1, \pi_2, \ldots, \pi_K)$$
$$X_{|H=h} \sim \mathcal{N}(\mu_h, \Sigma_h), \quad \forall h \in [K]$$

Learning Problem
Estimate mean vector $\mu_h$, covariance matrix $\Sigma_h$, and mixing weight $\text{Cat}(\pi_1, \pi_2, \ldots, \pi_K)$ of each subpopulation from unlabeled data.
Maximum Likelihood Estimator (MLE)

- Data \( \{x_i\}_{i=1}^{n} \)

- Likelihood \( \Pr_{\theta}(\text{data}) \equiv \prod_{i=1}^{n} \Pr_{\theta}(x_i) \)

- Model parameter estimation \( \hat{\theta}_{\text{mle}} := \arg\max_{\theta \in \Theta} \log \Pr_{\theta}(\text{data}) \)

- 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 \( \{x_i\}_{i=1}^n \) and the number of Gaussian components \( K \), the model parameters to be estimated are \( \theta = \{(\mu_h, \Sigma_h, \pi_h)\}_{h=1}^K \).

\[ \hat{\theta}_{mle} \text{ for Gaussian Mixture Models} \]

\[
\hat{\theta}_{mle} := \arg\max_{\theta} \sum_{i=1}^{n} \log \left( \sum_{h=1}^{K} \frac{\pi_h}{\det(\Sigma_h)^{1/2}} \exp \left( -\frac{1}{2} (x_i - \mu_h)^\top \Sigma_h^{-1} (x_i - \mu_h) \right) \right)
\]

- Solving MLE estimator is \textbf{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 \( \{x_i\}_{i=1}^n \) are generated by distribution \( \Pr_{\theta}(x_i) \) where the model parameters \( \theta \in \Theta \) are unknown. An estimator \( \hat{\theta} \) is consistent if

\[
\mathbb{E}\|\hat{\theta} - \theta\| \to 0 \quad \text{as} \quad n \to \infty
\]

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

Information-theoretic hardness

E.g., Mossel & Roch, 2006

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 \( \min_{i \neq j} \frac{\|\mu_i - \mu_j\|^2}{\sigma_i^2 + \sigma_j^2} \) 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 \( \mu_1, \ldots, \mu_K \) span a \( K \)-dimensional space.

This tutorial: statistically and computationally efficient learning algorithms for non-degenerate instances via method-of-moments.
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
5. Algorithms for Tensor Decompositions
6. Tensor Decomposition for Neural Network Compression
7. Conclusion
Method-of-Moments At A Glance

1. Determine function of model parameters $\theta$ estimatable from observable data:
   - Moments
     \[ \mathbb{E}_\theta[f(X)] \]

2. Form estimates of moments using data (iid samples $\{x_i\}_{i=1}^n$):
   - Empirical Moments
     \[ \hat{\mathbb{E}}[f(X)] \]

3. Solve the approximate equations for parameters $\theta$:
   - Moment matching
     \[ \mathbb{E}_\theta[f(X)] \xrightarrow{n \to \infty} \hat{\mathbb{E}}[f(X)] \]

---

**Toy Example**

How to estimate Gaussian variable, i.e., $(\mu, \Sigma)$, given iid samples $\{x_i\}_{i=1}^n \sim \mathcal{N}(\mu, \Sigma^2)$?
What is a tensor?

Multi-dimensional Array

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

\[ [a \otimes b]_{i_1,i_2} = a_{i_1} b_{i_2} \]

- Rank-1 matrix

\[ [a \otimes b \otimes c]_{i_1,i_2,i_3} = a_{i_1} b_{i_2} c_{i_3} \]

- 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**

\[ X \approx \sum_{h=1}^{R} a_h \otimes b_h \otimes c_h \]

- **Rank**: Minimum number of rank-1 tensors whose sum generates the tensor.
Multi-linear Transform

Multi-linear Operation

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

$$\mathcal{T}(X, Y, Z) := \sum_{h=1}^{K} (X^\top a_h) \otimes (Y^\top b_h) \otimes (Z^\top c_h).$$

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

$$\mathcal{T}(x, y, z) := \sum_{h=1}^{K} (x^\top a_h) \otimes (y^\top b_h) \otimes (z^\top c_h).$$
Tensors in Method of Moments

Matrix: Pair-wise relationship

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

$$M_2 = \mathbb{E}[x \otimes x]$$

Tensor: Triple-wise relationship or higher

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

$$M_3 = \mathbb{E}[x \otimes x \otimes x] = \mathbb{E}[x \otimes^3]$$
Why are tensors powerful?

Matrix Orthogonal Decomposition

- **Not unique** without eigenvalue gap

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

\[
e_1 = \left[ \frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2} \right], \quad e_2 = \left[ \frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2} \right]
\]

\[
u_1 = \left[ \frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2} \right], \quad v_2 = \left[ \frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2} \right]
\]
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
  \[
  e_1 = \left[ \frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2} \right] \\
  u_1 = \left[ \frac{\sqrt{2}}{2}, -\frac{\sqrt{2}}{2} \right]
  \]
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

$\text{Tensor} = u_1 \otimes u_1 \otimes u_1 + u_2 \otimes u_2 \otimes u_2$
Why are tensors powerful?

Matrix Orthogonal Decomposition

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

Tensor Orthogonal Decomposition (Harshman, 1970)

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

\[
\text{Slice}_i = u_1(i) u_1 \otimes u_1 + u_2(i) u_2 \otimes u_2
\]
Why are tensors powerful?

Matrix Orthogonal Decomposition

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

Tensor Orthogonal Decomposition (Harshman, 1970)

- **Unique**: eigenvalue gap not needed
- Slice of tensor has eigenvalue gap
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
5. Algorithms for Tensor Decompositions
6. Tensor Decomposition for Neural Network Compression
7. Conclusion
**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 $w$
  - per document $i$, $w^{(i)} \in \Delta^{K-1}$
- Document $\sim$ mixture of topics

---

**E.g.,**

- **Word Count per Document**
  - aardvark: 0
  - athlete: 3
  - zygote: 1

- **Topic Word Matrix**

- $\Pr_\theta["play" \mid \text{sports}] = 0.0002$
- $\Pr_\theta["game" \mid \text{sports}] = 0.0003$
- $\Pr_\theta["season" \mid \text{sports}] = 0.0001$

---

**Game**

- Season
- Play

**Science**

- Game
- Season

**Politics**

- Play
- Game

**Business**

- Sports
- Politics
- Business
Topic Modeling

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 iid $\sim a_h$

**E.g.,**

**Word Count per Document**

<table>
<thead>
<tr>
<th>Word</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>aardvark</td>
<td></td>
<td></td>
</tr>
<tr>
<td>athlete</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>zygote</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

**Topic Word Matrix**

- $\Pr_\theta[\text{“play”} \mid \text{sports}] = 0.0002$
- $\Pr_\theta[\text{“game”} \mid \text{sports}] = 0.0003$
- $\Pr_\theta[\text{“season”} \mid \text{sports}] = 0.0001$
Model Parameters of Topic Model for Single-topic Documents

Estimate Topic Proportion

- Topic proportion \( w = [w_1, \ldots, w_K] \)
  \( w_h = \mathbb{P}[\text{topic of word } = h] \)

Estimate Topic Word Matrix

- Topic-word matrix \( A = [a_1, \ldots, a_K] \)
  \( A_{jh} = \mathbb{P}[\text{word } = e_j | \text{topic } = h] \)

**Goal**: to estimate model parameters \( \{(a_h, w_h)\}_{h=1}^{K} \), given iid samples of \( n \) documents (word count \( \{c_{(i)}\}_{i=1}^{n} \))

- Frequency vector \( x^{(i)} = \frac{c_{(i)}}{L} \), the length of document is \( L = \sum_{j} c_{j}^{(i)} \)
Moment Matching

Nondegenerate model (linearly independent topic-word matrix)

- Generative process:
  - Choose $h \sim \text{Cat}(w_1, \ldots, w_K)$
  - Generate $L$ words $\sim a_h$

- $\mathbb{E}[x] = \sum_{h=1}^{K} \mathbb{P}[	ext{topic} = h] \mathbb{E}[x | \text{topic} = h]$ 

- $\mathbb{E}[x | \text{topic} = h] = \sum_j \mathbb{P}[\text{word} = e_j | \text{topic} = h] e_j = a_h$
Moment Matching

Nondegenerate model (linearly independent topic-word matrix)

- Generative process:
  - Choose $h \sim \text{Cat}(w_1, \ldots, w_K)$
  - Generate $L$ words $\sim a_h$

- $\mathbb{E}[x] = \sum_{h=1}^{K} P[\text{topic} = h] \mathbb{E}[x|\text{topic} = h] = \sum_{h=1}^{K} w_h a_h$

- $\mathbb{E}[x|\text{topic} = h] = \sum_{j} P[\text{word} = e_j|\text{topic} = h] e_j = a_h$
Identifiability: how long must the documents be?

Nondegenerate model (linearly independent topic-word matrix)

- **Generative process:**
  Choose \( h \sim \text{Cat}(w_1, \ldots, w_K) \)
  Generate \( L \) words \( \sim a_h \)

\[
\mathbb{E}[x] = \sum_{h=1}^{K} P[\text{topic} = h] \mathbb{E}[x|\text{topic} = h] = \sum_{h=1}^{K} w_h a_h
\]

\[
\mathbb{E}[x|\text{topic} = h] = \sum_j P[\text{word} = e_j|\text{topic} = h] e_j = a_h
\]

\( M_1 \): Distribution of words (\( \hat{M}_1 \): Occurrence frequency of words)

\[
M_1 = \mathbb{E}[x] = \sum_h w_h a_h; \quad \hat{M}_1 = \frac{1}{n} \sum_{i=1}^{n} x^{(i)}
\]
Identifiability: how long must the documents be?

Nondegenerate model (linearly independent topic-word matrix)

- Generative process:
  - Choose \( h \sim \text{Cat}(w_1, \ldots, w_K) \)
  - Generate \( L \) words \( \sim \alpha_h \)

\[
\begin{align*}
\mathbb{E}[x] &= \sum_{h=1}^{K} \mathbb{P}[	ext{topic} = h] \mathbb{E}[x | \text{topic} = h] = \sum_{h=1}^{K} w_h \alpha_h \\
\mathbb{E}[x | \text{topic} = h] &= \sum_{j} \mathbb{P}[	ext{word} = e_j | \text{topic} = h] e_j = \alpha_h
\end{align*}
\]

\( M_1 \): Distribution of words (\( \hat{M}_1 \): Occurrence frequency of words)

\[
M_1 = \mathbb{E}[x] = \sum_h w_h \alpha_h; \quad \hat{M}_1 = \frac{1}{n} \sum_{i=1}^{n} x^{(i)}
\]

No unique decomposition of vectors
Identifiability: how long must the documents be?

Nondegenerate model (linearly independent topic-word matrix)

- Generative process:
  - Choose $h \sim \text{Cat}(w_1, \ldots, w_K)$
  - Generate $L$ words $\sim a_h$

  $E[x] = \sum_{h=1}^{K} P[\text{topic} = h]E[x|\text{topic} = h] = \sum_{h=1}^{K} w_h a_h$

  $E[x|\text{topic} = h] = \sum_{j} P[\text{word} = e_j|\text{topic} = h] e_j = a_h$

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

$M_2 = E[x \otimes x] = \sum_{h} w_h a_h \otimes a_h$; \quad $\hat{M}_2 = \frac{1}{n} \sum_{i=1}^{n} x^{(i)} \otimes x^{(i)}$
Identifiability: how long must the documents be?

Nondegenerate model (linearly independent topic-word matrix)

- Generative process:
  - Choose \( h \sim \text{Cat}(w_1, \ldots, w_K) \)
  - Generate \( L \) words \( \sim a_h \)

  \[
  \mathbb{E}[x] = \sum_{h=1}^{K} \mathbb{P}[\text{topic} = h] \mathbb{E}[x | \text{topic} = h] = \sum_{h=1}^{K} w_h a_h
  \]

  \[
  \mathbb{E}[x | \text{topic} = h] = \sum_{j} \mathbb{P}[\text{word} = e_j | \text{topic} = h] e_j = a_h
  \]

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

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

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 ($\hat{M}_2$: Co-occurrence of word pairs)

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

Many such $W$'s, find one such that $v_h = W^\top \alpha_h$ orthogonal
Identifiability: how long must the documents be?

Nondegenerate model (linearly independent topic-word matrix)

Know a $W$ such that

\[
\begin{align*}
\mathcal{M}_3 &:= \mathbb{E}[x \otimes^3] = \sum_h w_h \alpha_h \otimes^3; \\
\hat{\mathcal{M}}_3 &:= \frac{1}{n} \sum_{i=1}^n x^{(i)} \otimes^3
\end{align*}
\]

Orthogonalize the tensor, project data with $W$: $\mathcal{M}_3(W, W, W)$
Identifiability: how long must the documents be?

Nondegenerate model (linearly independent topic-word matrix)

Know a $W$ such that

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

$\mathbb{M}_3(W, W, W) = \mathbb{E}[(W^T x) \otimes^3] = \sum_h w_h (W^T a_h) \otimes^3$; $\hat{\mathbb{M}}_3(W, W, W) = \frac{1}{n} \sum_{i=1}^n (W^T x^{(i)}) \otimes^3$

Unique orthogonal tensor decomposition $\{\hat{v}_h\}_{h=1}^K$
Identifiability: how long must the documents be?

Nondegenerate model (linearly independent topic-word matrix)

Know a $W$ such that

\[ M_3 : \text{Distribution of word triples} \]
\[ \hat{M}_3 : \text{Co-occurrence of word triples} \]

\[
M_3(W, W, W) = \mathbb{E}[(W^T x) \otimes^3] = \sum_h w_h (W^T a_h) \otimes^3; \quad \hat{M}_3(W, W, W) = \frac{1}{n} \sum_{i=1}^n (W^T x^{(i)}) \otimes^3
\]

Model parameter estimation:

\[ \hat{a}_h = (W^T)^\dagger \hat{v}_h \]
Identifiability: how long must the documents be?

Nondegenerate model (linearly independent topic-word matrix)

Know a \( W \) such that

\[
\mathcal{M}_3(W, W, W) = \mathbb{E}[(W^\top x) \otimes^3] = \sum_h w_h (W^\top a_h) \otimes^3; \quad \tilde{\mathcal{M}}_3(W, W, W) = \frac{1}{n} \sum_{i=1}^n (W^\top x^{(i)}) \otimes^3
\]

\( L \geq 3: \) Learning Topic Models through Matrix/Tensor Decomposition
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.

\[ M_3 = \mathbb{E}[x \otimes x \otimes x] = \sum_h \omega_h a_h \otimes a_h \otimes a_h \]

\( \widehat{M}_3 \): 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

![Perplexity Comparison](image1)

![Running Time Comparison](image2)
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

![Perplexity Comparison](chart1.png)

![Running Time Comparison](chart2.png)

Learning Communities from Graph Connectivity

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

![Error / Group Comparison](chart3.png)

![Running Times Comparison](chart4.png)

Orders of Magnitude Faster & More Accurate

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
5. Algorithms for Tensor Decompositions
6. Tensor Decomposition for Neural Network Compression
7. Conclusion
Jennrich’s Algorithm (Simplified)

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

$$\text{Tensor} = u_1 \otimes u_1 \otimes u_1 + u_2 \otimes u_2 \otimes u_2$$
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).

Properties of Tensor Slices

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

$$\sum_i c_i \text{Slice}_i = \text{SmileyFace} + \text{SmileyFace} \quad \text{w.h.p.} \neq$$
Jennrich’s Algorithm (Simplified)

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

Properties of Tensor Slices

- Linear combination of slices $\mathbf{T}(I, I, c) = \sum_h <\mu_h, c> \mu_h \otimes \mu_h$

Intuitions for Jennrich’s Algorithm
**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).

**Properties of Tensor Slices**
- Linear combination of slices \( \mathcal{T}(I, I, c) = \sum_{h} < \mu_h, c > \mu_h \otimes \mu_h \)

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

**Properties of Tensor Slices**

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

**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 \( \mathcal{T} = \sum_{h=1}^{K} \mu_h \otimes^3 \) 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 \( \{ \hat{\mu}_h \}_{h=1}^K \) a.s. \( \approx \{ \mu_h \}_{h=1}^K \)

1. Sample \( c \) and \( c' \) independently & uniformly at random from \( S^{d-1} \)
2. Return \( \{ \hat{\mu}_h \}_{h=1}^K \leftarrow \) eigenvectors of \( (\mathcal{T}(I, I, c) \mathcal{T}(I, I, c')^\dagger) \)
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 $\{\hat{\mu}_h\}_{h=1}^{K} \stackrel{\text{a.s.}}{=} \{\mu_h\}_{h=1}^{K}$

1. Sample $c$ and $c'$ independently & uniformly at random from $S^{d-1}$
2. Return $\{\hat{\mu}_h\}_{h=1}^{K} \leftarrow$ eigenvectors of $(\mathcal{T}(I, I, c)\mathcal{T}(I, I, c')^\dagger)$

**Consistency of Jennrich’s Algorithm?**

Estimators $\{\hat{\mu}_h\}_{h=1}^{K} \equiv$ unknown components $\{\mu_h\}_{h=1}^{K}$ (up to scaling)?
Analysis of Consistency of Jennrich’s algorithm

Recall: Linear comb. of slices share eigenvectors \( \{\mu_h\}_{h=1}^K \), i.e.,

\[
\mathcal{T}(I, I, c)\mathcal{T}(I, I, c')^\dagger \overset{a.s.}{=} UD_c U^\top (U^\top)^\dagger D_c^{-1} U^\dagger \overset{a.s.}{=} U(D_c D_c^{-1})^\dagger ,
\]

where \( U = [\mu_1|\ldots|\mu_K] \) are the linearly independent tensor components and \( D_c = \text{Diag}\left( <\mu_1, c>, \ldots, <\mu_K, c> \right) \) is diagonal.
Analysis of Consistency of Jennrich’s algorithm

Recall: Linear comb. of slices share eigenvectors \( \{\mu_h\}_{h=1}^K \), i.e.,

\[
\mathcal{T}(I, I, c)\mathcal{T}(I, I, c') \overset{\text{a.s.}}{=} UD_c U^\top (U^\top)^\dagger D_c'^{-1} U^\dagger \overset{\text{a.s.}}{=} U(D_c D_c'^{-1}) U^\dagger,
\]

where \( U = [\mu_1|\ldots|\mu_K] \) are the linearly independent tensor components and \( D_c = \text{Diag}(<\mu_1, c>, \ldots, <\mu_K, c>) \) is diagonal.

By linear independence of \( \{\mu_i\}_{i=1}^K \) and random choice of \( c \) and \( c' \):

1. \( U \) has rank \( K \);
Analysis of Consistency of Jennrich’s algorithm

Recall: Linear comb. of slices share eigenvectors \( \{ \mu_h \}_{h=1}^K \), i.e.,

\[
\mathcal{T}(I, I, c)\mathcal{T}(I, I, c')^\dagger \overset{\text{a.s.}}{=} UD_c U^\top (U^\top)^\dagger D_c^{-1} U^\dagger \overset{\text{a.s.}}{=} U(D_c D_c^{-1}) U^\dagger,
\]

where \( U = [\mu_1|\ldots|\mu_K] \) are the linearly independent tensor components and \( D_c = \text{Diag}(\langle \mu_1, c \rangle, \ldots, \langle \mu_K, c \rangle) \) is diagonal.

By linear independence of \( \{ \mu_i \}_{i=1}^K \) and random choice of \( c \) and \( c' \):

1. \( U \) has rank \( K \);
2. \( D_c \) and \( D_c' \) are invertible (a.s.).
Analysis of Consistency of Jennrich’s algorithm

Recall: Linear comb. of slices share eigenvectors \( \{ \mu_h \}_{h=1}^K \), i.e.,

\[
T(I, I, c)T(I, I, c') \dagger \text{a.s.} = UD_c U^\top (U^\top)^\dagger D_c^{-1} U \dagger \text{a.s.} = U (D_c D_c^{-1}) U \dagger,
\]

where \( U = [\mu_1 | \ldots | \mu_K] \) are the linearly independent tensor components and \( D_c = \text{Diag}(<\mu_1, c>, \ldots, <\mu_K, c>) \) is diagonal.

By linear independence of \( \{ \mu_i \}_{i=1}^K \) and random choice of \( c \) and \( c' \):

1. \( U \) has rank \( K \);
2. \( D_c \) and \( D_c' \) are invertible \( \text{(a.s.)} \); 
3. Diagonal entries of \( D_c D_c^{-1} \) are distinct \( \text{(a.s.)} \);
Analysis of Consistency of Jennrich’s algorithm

Recall: Linear comb. of slices share eigenvectors \( \{ \mu_h \}_{h=1}^K \), i.e.,

\[
\mathcal{T}(I, I, c) \mathcal{T}(I, I, c') \dagger \overset{\text{a.s.}}{=} UD_c U^\top (U^\top)^\dagger D_c'^{-1} U^\dagger \overset{\text{a.s.}}{=} U(D_c D_c'^{-1}) U^\dagger,
\]

where \( U = [\mu_1|\ldots|\mu_K] \) are the linearly independent tensor components and \( D_c = \text{Diag}(<\mu_1, c>, \ldots, <\mu_K, c>) \) is diagonal.

By linear independence of \( \{ \mu_i \}_{i=1}^K \) and random choice of \( c \) and \( c' \):

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

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

Jennrich’s algorithm is consistent
Error-tolerant algorithms for tensor decompositions
Moment Estimator: Empirical Moments
**Moment Estimator: Empirical Moments**

- Moments $\mathbb{E}_{\theta}[f(X)]$ are functions of model parameters $\theta$
- Empirical Moments $\hat{\mathbb{E}}[f(X)]$ are computed using iid samples $\{x_i\}_{i=1}^n$ only
Moment Estimator: Empirical Moments

- Moments $E_\theta[f(X)]$ are functions of model parameters $\theta$
- Empirical Moments $\hat{E}[f(X)]$ are computed using iid samples $\{x_i\}_{i=1}^n$ only

Example

- Third Order Moment: distribution of word triples
  $E[x \otimes x \otimes x] = \sum_h w_h a_h \otimes a_h \otimes a_h$
- Empirical Third Order Moment: co-occurrence frequency of word triples
  $\hat{E}[x \otimes x \otimes x] = \frac{1}{n} \sum_{i=1}^n x_i \otimes x_i \otimes x_i$
Moment Estimator: Empirical Moments

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

Example

- Third Order Moment: distribution of word triples
  $$\mathbb{E}[x \otimes x \otimes x] = \sum_h w_h \mathbf{a}_h \otimes \mathbf{a}_h \otimes \mathbf{a}_h$$
- Empirical Third Order Moment: co-occurrence frequency of word triples
  $$\hat{\mathbb{E}}[x \otimes x \otimes x] = \frac{1}{n} \sum_{i=1}^{n} x_i \otimes x_i \otimes x_i$$

- Inevitably expect error of order $n^{-1/2}$ in some norm, e.g.,
  - Operator norm: $||\mathbb{E}[x \otimes x \otimes x] - \hat{\mathbb{E}}[x \otimes x \otimes x]|| \lesssim n^{-1/2}$
    where $||\mathcal{T}|| := \sup_{x,y,z \in S^{d-1}} \mathcal{T}(x,y,z)$
  - Frobenius norm: $||\mathbb{E}[x \otimes x \otimes x] - \hat{\mathbb{E}}[x \otimes x \otimes x]||_F \lesssim n^{-1/2}$
    where $||\mathcal{T}||_F := \sqrt{\sum_{i,j,k} T_{i,j,k}^2}$
Stability of Jennrich’s Algorithm

Recall Jennrich’s algorithm

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

---

**Algorithm**  Jennrich’s Algorithm

**Require:** Tensor \( \mathbf{T} \in \mathbb{R}^{d \times d \times d} \)

**Ensure:** Components \( \{\hat{\mu}_h\}_{h=1}^{K} \) a.s. = \( \{\mu_h\}_{h=1}^{K} \)

1. Sample \( c \) and \( c' \) independently & uniformly at random from \( S^{d-1} \)
2. Return \( \{\hat{\mu}_h\}_{h=1}^{K} \leftarrow \) eigenvectors of \( (\mathbf{T}(I, I, c)\mathbf{T}(I, I, c')^\dagger) \)
Stability of Jennrich’s Algorithm

Recall Jennrich’s algorithm

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

**Algorithm** Jennrich’s Algorithm

<table>
<thead>
<tr>
<th>Require:</th>
<th>Tensor ( \mathbf{T} \in \mathbb{R}^{d \times d \times d} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ensure:</td>
<td>Components ( { \hat{\mu}<em>h }</em>{h=1}^{K} ) a.s. ( { \mu_h }_{h=1}^{K} )</td>
</tr>
</tbody>
</table>

1. Sample \( c \) and \( c' \) independently & uniformly at random from \( S^{d-1} \)
2. Return \( \{ \hat{\mu}_h \}_{h=1}^{K} \leftarrow \) eigenvectors of \( (\mathbf{T}(I, I, c) \mathbf{T}(I, I, c')^{\dagger}) \)

**Challenge:** Only have access to \( \hat{\mathbf{T}} \) such that \( \| \hat{\mathbf{T}} - \mathbf{T} \| \lesssim n^{-\frac{1}{2}} \)
Stability of Jennrich’s Algorithm

Recall Jennrich’s algorithm

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

**Algorithm Jennrich’s Algorithm**

<table>
<thead>
<tr>
<th>Require:</th>
<th>Tensor $\hat{\mathbf{T}} \in \mathbb{R}^{d \times d \times d}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ensure:</td>
<td>Components ${\hat{\mu}<em>h}</em>{h=1}^{K}$ a.s. ${\mu_h}_{h=1}^{K}$</td>
</tr>
<tr>
<td>1:</td>
<td>Sample $c$ and $c'$ independently &amp; uniformly at random from $S^{d-1}$</td>
</tr>
<tr>
<td>2:</td>
<td>Return ${\hat{\mu}<em>h}</em>{h=1}^{K} \leftarrow$ eigenvectors of $\left(\hat{\mathbf{T}}(I, I, c)\hat{\mathbf{T}}(I, I, c')^\dagger\right)$</td>
</tr>
</tbody>
</table>

**Challenge:** Only have access to $\hat{\mathbf{T}}$ such that $\|\hat{\mathbf{T}} - \mathbf{T}\| \lesssim n^{-\frac{1}{2}}$
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 $\hat{\mathcal{T}} \in \mathbb{R}^{d \times d \times d}$

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

1: Sample $c$ and $c'$ independently & uniformly at random from $S^{d-1}$
2: Return $\{\hat{\mu}_h\}_{h=1}^{K} \leftarrow$ eigenvectors of $\left(\hat{\mathcal{T}}(I, I, c)\hat{\mathcal{T}}(I, I, c')^{\dagger}\right)$

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 $\hat{\mathcal{T}} \in \mathbb{R}^{d \times d \times d}$
Ensure: Components $\{\hat{\mu}_h\}_{h=1}^{K}$ a.s. $\{\mu_h\}_{h=1}^{K}$

1: Sample $c$ and $c'$ independently & uniformly at random from $S^{d-1}$
2: Return $\{\hat{\mu}_h\}_{h=1}^{K} \leftarrow$ eigenvectors of $\left(\hat{\mathcal{T}}(I, I, c)\hat{\mathcal{T}}(I, I, c')^\dagger\right)$

Stability of eigenvectors requires eigenvalue gaps

- To ensure eigenvalue gaps for $\hat{\mathcal{T}}(\cdot, \cdot, c)\hat{\mathcal{T}}(\cdot, \cdot, c)^\dagger$, $\|\hat{\mathcal{T}}(\cdot, \cdot, c)\hat{\mathcal{T}}(\cdot, \cdot, c)^\dagger - \mathcal{T}(\cdot, \cdot, c)\mathcal{T}(\cdot, \cdot, c)^\dagger\| \ll \Delta$ is needed.
Stability of Jennrich’s Algorithm

Recall Jennrich’s algorithm

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

**Algorithm** Jennrich’s Algorithm

**Require:** Tensor \( \hat{\mathcal{T}} \in \mathbb{R}^{d \times d \times d} \)

**Ensure:** Components \( \{ \hat{\mu}_h \}_{h=1}^{K} \) a.s. \( \{ \mu_h \}_{h=1}^{K} \)?

1. Sample \( c \) and \( c' \) independently & uniformly at random from \( S^{d-1} \)
2. Return \( \{ \hat{\mu}_h \}_{h=1}^{K} \leftarrow \) eigenvectors of \( \left( \hat{\mathcal{T}}(I, I, c) \hat{\mathcal{T}}(I, I, c')^\dagger \right) \)

Stability of eigenvectors requires eigenvalue gaps

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

<table>
<thead>
<tr>
<th>Require:</th>
<th>Tensor ( \hat{T} \in \mathbb{R}^{d \times d \times d} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ensure:</td>
<td>Components ( {\hat{\mu}<em>h}</em>{h=1}^{K} ) a.s. = ( {\mu_h}_{h=1}^{K} )?</td>
</tr>
</tbody>
</table>

1. Sample \( c \) and \( c' \) independently & uniformly at random from \( S^{d-1} \)
2. Return \( \{\hat{\mu}_h\}_{h=1}^{K} \leftarrow \text{eigenvectors of} \left( \hat{T}(I, I, c) \hat{T}(I, I, c')^\dagger \right) \)

#### Stability of eigenvectors requires eigenvalue gaps

- To ensure eigenvalue gaps for \( \hat{T}(\cdot, \cdot, c) \hat{T}(\cdot, \cdot, c)^\dagger \), \( \| \hat{T}(\cdot, \cdot, c) \hat{T}(\cdot, \cdot, c)^\dagger - \mathcal{T}(\cdot, \cdot, c) \mathcal{T}(\cdot, \cdot, c)^\dagger \| \ll \Delta \) is needed.
- Ultimately, \( \| \hat{T} - \mathcal{T} \|_F \ll \frac{1}{\text{poly} \, d} \) is required. **A different approach?**
Initial Ideas

In many applications, we estimate moments of the form

\[ \mathcal{M}_3 = \sum_{h=1}^{K} w_h a_h \otimes^3, \]

where \( \{a_h\}_{h=1}^{K} \) are assumed to be linearly independent.

What if \( \{a_h\}_{h=1}^{K} \) has orthonormal columns?
Initial Ideas

In many applications, we estimate moments of the form

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

where \( \{a_h\}_{h=1}^{K} \) are assumed to be linearly independent.

What if \( \{a_h\}_{h=1}^{K} \) has orthonormal columns?

$$\mathcal{M}_3(I, a_i, a_i) = \sum_h w_h \langle a_h, a_i \rangle^2 a_h = w_i a_i, \forall i.$$
Initial Ideas

In many applications, we estimate moments of the form

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

where \( \{a_h\}_{h=1}^{K} \) are assumed to be linearly independent.

What if \( \{a_h\}_{h=1}^{K} \) has orthonormal columns?

$$\mathcal{M}_3(I, a_i, a_i) = \sum_h w_h \langle a_h, a_i \rangle^2 a_h = w_i a_i, \ \forall i.$$

- Analogous to matrix eigenvectors: \( Mv = M(I, v) = \lambda v. \)
Initial Ideas

In many applications, we estimate moments of the form

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

where \( \{a_h\}_{h=1}^{K} \) are assumed to be linearly independent.

What if \( \{a_h\}_{h=1}^{K} \) has orthonormal columns?

$$\mathcal{M}_3(I, a_i, a_i) = \sum_h w_h \langle a_h, a_i \rangle^2 a_h = w_i a_i, \ \forall i.$$  

- Analogous to matrix eigenvectors: \( Mv = M(I, v) = \lambda v \).
- **Define** orthonormal \( \{a_h\}_{h=1}^{K} \) as eigenvectors of tensor \( \mathcal{M}_3 \).
Initial Ideas

In many applications, we estimate moments of the form

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

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

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

$$\mathcal{M}_3(I, a_i, a_i) = \sum_h w_h \langle a_h, a_i \rangle^2 a_h = w_i a_i, \forall i.$$  

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

Two Problems

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

In many applications, we estimate moments of the form

\[ \mathcal{M}_3 = \sum_{h=1}^{K} w_h a_h \otimes^3, \]

where \( \{a_h\}_{h=1}^{K} \) are assumed to be linearly independent.

What if \( \{a_h\}_{h=1}^{K} \) has orthonormal columns?

\[ \mathcal{M}_3(I, a_i, a_i) = \sum_h w_h \langle a_h, a_i \rangle^2 a_h = w_i a_i, \ \forall i. \]

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

Two Problems

- \( \{a_h\}_{h=1}^{K} \) is not orthogonal in general.
- How to find eigenvectors of a tensor?
**Whitening** is the process of finding a whitening matrix $W$ such that multi-linear operation (using $W$) on $M_3$ orthogonalize its components:

$$M_3(W, W, W) = \sum_h w_h (W^\top a_h) \otimes^3$$

$$= \sum_h w_h v_h \otimes^3, \quad v_h \perp v_{h'}, \quad \forall h \neq h'$$
Whitening

Given

\[ M_3 = \sum_h w_h a_h \otimes^3, \quad M_2 = \sum_h w_h a_h \otimes a_h, \]
Whitening

Given

\[ \mathcal{M}_3 = \sum_h w_h a_h \otimes^3, \quad \mathcal{M}_2 = \sum_h w_h a_h \otimes a_h, \]

Find whitening matrix \( W \) s.t. \( W^\top a_h = v_h \) are orthogonal.
Whitening

Given

\[ M_3 = \sum_h w_h a_h \otimes a_h \otimes a_h, \quad M_2 = \sum_h w_h a_h \otimes a_h, \]

Find whitening matrix \( W \) s.t. \( W^\top a_h = v_h \) are orthogonal.

When \( \{a_h\}_{h=1}^K \in \mathbb{R}^{d \times K} \) has full column rank, it is an invertible transformation.
Using Whitening to Obtain Orthogonal Tensor

Tensor $M$ $ightarrow$ Tensor $T$
Using Whitening to Obtain Orthogonal Tensor

\[ T = M_3(W, W, W) = \sum_h w_h (W^T a_h)^\otimes 3. \]
Using Whitening to Obtain Orthogonal Tensor

Multi-linear transform

- $\mathcal{T} = \mathcal{M}_3(W, W, W) = \sum_h w_h (W^\top a_h) \otimes^3$.
- $\mathcal{T} = \sum_{h \in [K]} w_h \cdot v_h \otimes^3$ has orthogonal components.
Using Whitening to Obtain Orthogonal Tensor

Multi-linear transform

- $\mathcal{T} = \mathcal{M}_3(W, W, W) = \sum_h w_h (W^\top a_h)^\otimes 3$.  
- $\mathcal{T} = \sum_{h \in [K]} w_h \cdot v_h \otimes 3$ 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

\[ M_3 = \sum_h w_h a_h \otimes^3, \quad M_2 = \sum_h w_h a_h \otimes a_h, \]

Goal: \( W \) such that

\[ a_1 \quad a_2 \quad a_3 \quad W \]

\[ \rightarrow v_1 \quad v_2 \quad v_3 \]
How to Find Whitening Matrix?

Given

\[ \mathcal{M}_3 = \sum_h w_h a_h \otimes 3, \quad M_2 = \sum_h w_h a_h \otimes a_h, \]

Goal: \( W \) such that

- Use pairwise moments \( M_2 \) to find \( W \) s.t. \( W^\top M_2 W = I \).
How to Find Whitening Matrix?

Given

\[ M_3 = \sum_h w_h a_h \otimes 3, \quad M_2 = \sum_h w_h a_h \otimes a_h, \]

Goal: \( W \) such that

- Use pairwise moments \( M_2 \) to find \( W \) s.t. \( W^\top M_2 W = I \).
- \( W = U \text{Diag}(\tilde{\lambda}^{-1/2}) \), where Eigen-decomposition \( M_2 = U \text{Diag}(\tilde{\lambda}) U^\top \).
How to Find Whitening Matrix?

Given

\[ M_3 = \sum_h w_h a_h \otimes^3, \quad M_2 = \sum_h w_h a_h \otimes a_h, \]

**Goal:** \( W \) such that

- Use pairwise moments \( M_2 \) to find \( W \) s.t. \( W^\top M_2 W = I \).
- \( W = U \text{Diag}(\tilde{\lambda}^{-1/2}) \), where Eigen-decomposition
  \( M_2 = U \text{Diag}(\tilde{\lambda}) U^\top \).
- \( V := W^\top A \text{Diag}(w)^{1/2} \) is an orthogonal matrix.

\[ T = M_3(W, W, W) = \sum_h w_h^{-1/2} (W^\top a_h \sqrt{w_h}) \otimes^3 \]
\[ = \sum_h \lambda_h v_h \otimes^3, \quad \lambda_h := w_h^{-1/2}. \]

\( T \) is an orthogonal tensor.
Initial Ideas

In many applications, we estimate moments of the form
\[
\mathcal{M}_3 = \sum_h w_h a_h \otimes^3,
\]
where \(\{a_h\}_{h=1}^K\) are assumed to be linearly independent.

What if \(\{a_h\}_{h=1}^K\) has orthonormal columns?

\[
\mathcal{M}_3(I, a_i, a_i) = \sum_h w_h \langle a_h, a_i \rangle^2 a_h = w_i a_i, \quad \forall i.
\]

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

Two Problems

- \(\{a_h\}_{h=1}^K\) is not orthogonal in general.
- **How to find eigenvectors of a tensor?**
Task: Given matrix $\mathbf{M} = \sum_{h=1}^{K} \lambda_h \mathbf{v}_h \otimes \mathbf{v}_h$ with orthonormal components $\{\mathbf{v}_h\}_{h=1}^{K}$ ($\mathbf{v}_h \perp \mathbf{v}_{h'}$, $\forall h \neq h'$), find the components/eigenvectors.
**Task:** Given matrix $M = \sum_{h=1}^{K} \lambda_h v_h \otimes v_h$ with orthonormal components $\{v_h\}_{h=1}^{K}$ ($v_h \perp v_{h'}$, $\forall h \neq h'$), find the components/eigenvectors.

**Properties of Matrix Eigenvectors**

- **Fixed point:** linear transform $M(I, v_i) = \sum_{h} \lambda_h \langle v_i, v_h \rangle v_h = \lambda_i v_i$
**Task:** Given matrix $M = \sum_{h=1}^{K} \lambda_h v_h \otimes v_h$ with orthonormal components $\{v_h\}_{h=1}^{K}$ ($v_h \perp v_{h'}$, $\forall h \neq h'$), find the components/eigenvectors.

**Properties of Matrix Eigenvectors**

- Fixed point: linear transform $M(I, v_i) = \sum_h \lambda_h \langle v_i, v_h \rangle v_h = \lambda_i v_i$

**Intuitions for Matrix Power Method**
Review: Orthogonal Matrix Eigen Decomposition

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

Properties of Matrix Eigenvectors

- Fixed point: linear transform \( M(I, v_i) = \sum_{h} \lambda_h \langle v_i, v_h \rangle v_h = \lambda_i v_i \)

Intuitions for Matrix Power Method

**Linear** transform on eigenvectors \( \{v_h\}_{h=1}^{K} \) preserve direction
Orthogonal Tensor Eigen Decomposition

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

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

**Properties of Tensor Eigenvectors**

- **Fixed point: bi-linear transform**
  
  $\mathcal{T}(I, v_i, v_i) = \sum_h \lambda_h \langle v_i, v_h \rangle^2 v_h = \lambda_i v_i$

  $\text{Tensor} = u_1 \otimes u_1 \otimes u_1 + u_2 \otimes u_2 \otimes u_2$
Orthogonal Tensor Eigen Decomposition

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

**Properties of Tensor Eigenvectors**

- **Fixed point:** bi-linear transform
  
  \[ \mathbf{T}(I, \mathbf{v}_i, \mathbf{v}_i) = \sum_h \lambda_h \langle \mathbf{v}_i, \mathbf{v}_h \rangle^2 \mathbf{v}_h = \lambda_i \mathbf{v}_i \]

**Intuitions for Tensor Power Method**
Orthogonal Tensor Eigen Decomposition

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

**Properties of Tensor Eigenvectors**

- **Fixed point:** bi-linear transform
  
  $\mathcal{T}(I, v_i, v_i) = \sum_h \lambda_h \langle v_i, v_h \rangle^2 v_h = \lambda_i v_i$

**Intuitions for Tensor Power Method**

- **Bilinear** transform on eigenvectors $\{v_h\}_{h=1}^{K}$ preserve direction
Orthogonal Matrix Eigen Decomposition

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

**Algorithm** Matrix Power Method

**Require:** Matrix \( M \in \mathbb{R}^{K \times K} \)

**Ensure:** Components \( \{ \hat{v}_h \}_{h=1}^{K} \) w.h.p. = \( \{ v_h \}_{h=1}^{K} \)

1: for \( h = 1 : K \) do
2: Sample \( u_0 \) uniformly at random from \( S^{K-1} \)
3: for \( i = 1 : T \) do
4: \( u_i \leftarrow \frac{M(I, u_{i-1})}{\|M(I, u_{i-1})\|} \)
5: end for
6: \( \hat{v}_h \leftarrow u_T, \hat{\lambda}_h \leftarrow M(\hat{v}_h, \hat{v}_h) \)
7: Deflate \( M \leftarrow M - \hat{\lambda}_h \hat{v}_h \otimes 2 \)
8: end for
Orthogonal Matrix Eigen Decomposition

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

**Algorithm** Matrix Power Method

<table>
<thead>
<tr>
<th>Require: Matrix ( M \in \mathbb{R}^{K \times K} )</th>
</tr>
</thead>
</table>
| Ensure: Components \( \{\hat{v}_h\}_{h=1}^{K} \) w.h.p. \( \equiv \{v_h\}_{h=1}^{K} \)

1: for \( h = 1 : K \) do
2: Sample \( u_0 \) uniformly at random from \( S^{K-1} \)
3: for \( i = 1 : T \) do
4: \( u_i \leftarrow \frac{M(I,u_{i-1})}{\|M(I,u_{i-1})\|} \)
5: end for
6: \( \hat{v}_h \leftarrow u_T, \hat{\lambda}_h \leftarrow M(\hat{v}_h, \hat{v}_h) \)
7: Deflate \( M \leftarrow M - \hat{\lambda}_h \hat{v}_h \otimes 2 \)
8: end for

Consistency of Matrix Power Method?

Is there convergence? \( \{\hat{v}_h\}_{h=1}^{K} \equiv \{v_h\}_{h=1}^{K} \) w.h.p.?
Orthogonal Matrix Eigen Decomposition

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

**Algorithm** Matrix Power Method

**Require:** Matrix \( M \in \mathbb{R}^{K \times K} \)

**Ensure:** Components \( \{\hat{v}_h\}_{h=1}^{K} \) \( \overset{w.h.p.}{=} \) \( \{v_h\}_{h=1}^{K} \)

1: for \( h = 1 : K \) do
2: Sample \( u_0 \) uniformly at random from \( S^{K-1} \)
3: for \( i = 1 : T \) do
4: \( u_i \leftarrow \frac{M(I,u_{i-1})}{\|M(I,u_{i-1})\|} \)
5: end for
6: \( \hat{v}_h \leftarrow u_T, \hat{\lambda}_h \leftarrow M(\hat{v}_h,\hat{v}_h) \)
7: Deflate \( M \leftarrow M - \hat{\lambda}_h \hat{v}_h \otimes 2 \)
8: end for

Consistency of Matrix Power Method?

Is there convergence? \( \{\hat{v}_h\}_{h=1}^{K} \overset{?}{=} \{v_h\}_{h=1}^{K} \) \( w.h.p. \)?
Does the convergence depend on initialization?
Orthogonal Tensor Eigen Decomposition

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

---

**Algorithm**  
**Tensor Power Method**

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

**Ensure:** Components $\{\hat{\mathbf{v}}_h\}_{h=1}^{K}$ w.h.p. $= \{\mathbf{v}_h\}_{h=1}^{K}$

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

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

Algorithm Tensor Power Method

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

Ensure: Components $\{\hat{\mathbf{v}}_h\}_{h=1}^{K}$ w.h.p. $\equiv \{\mathbf{v}_h\}_{h=1}^{K}$

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

Consistency of Tensor Power Method?

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

Does the convergence depend on initialization?
Analysis of Consistency of Matrix Power Method

- Order eigenvectors \( \{v_h\}_{h=1}^K \) such that corresponding eigenvalues satisfy \( \lambda_1 \geq \lambda_2 \ldots \geq \lambda_K \).
- Project initial point \( u_0 \) onto eigenvectors \( \{v_h\}_{h=1}^K \)

\[
c_h = \langle u_0, v_h \rangle, \forall h
\]

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 \( v_1 \).
- Converges linearly to \( v_1 \) assuming gap \( \lambda_2/\lambda_1 < 1 \).
  - Linear transform permits \( M(I, u_0) = \sum_h \lambda_h (v_h^T u_0) v_h = \sum_h \lambda_h c_h v_h \), i.e., projection in \( v_h \) direction is scaled by \( \lambda_h \).
  - In \( t \) iterations,

\[
\frac{\left( v_1^T v \right)^2}{\sum_i (v_i^T v)^2} \geq 1 - K \left( \frac{\lambda_2}{\lambda_1} \right)^{2t}
\]
Analysis of Consistency of Tensor Power Method

- Project initial point \( u_0 \) onto eigenvectors \( c_h = \langle u_0, v_h \rangle, \ \forall h \).
- Order eigenvectors \( \{v_h\}_{h=1}^K \) 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 \( v_1 \).
  Note \( v_1 \) is NOT necessarily the largest eigenvector.
- Converges quadratically to \( v_1 \) assuming gap \( \frac{\lambda_2 |c_2|}{\lambda_1 |c_1|} < 1 \).
  - Bi-linear transform permits \( T(I, u_0, u_0) = \sum_h \lambda_h (v_h^T u_0)^2 v_h = \sum_h \lambda_h c_h^2 v_h \)
    i.e., projection in \( v_h \) direction is squared then scaled by \( \lambda_h \).
  - In \( t \) iterations,
    \[
    \frac{(v_1^T v)^2}{\sum_i (v_i^T v)^2} \geq 1 - k \left( \frac{\lambda_1}{\max_{i \neq 1} \lambda_i} \right)^2 \left| \frac{v_2 c_2}{v_1 c_1} \right|^{2t+1}.
    \]
Matrix vs. tensor power iteration

Matrix power iteration:

Tensor power iteration:
Matrix vs. tensor power iteration

Matrix power iteration:
- Requires gap between largest and second-largest eigenvalue. 
  Property of the matrix only.

Tensor power iteration:
- Requires gap between largest and second-largest $\lambda_h |c_h|$. 
  Property of the tensor and initialization $u_0$. 
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 $u_0$.
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 $u_0$.
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

\[ \mathcal{T} = \sum_{h \in [K]} \lambda_h v_h \otimes^3. \]

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

- \( \{v_h\}_{h=1}^K \) are eigenvectors as \( \mathcal{T}(I, v_h, v_h) = \lambda_h v_h. \)
Spurious Eigenvectors for Tensor Eigen Decomposition

\[ \mathcal{T} = \sum_{h \in [K]} \lambda_h v_h \otimes^3. \]

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

- \( \{v_h\}_{h=1}^K \) are eigenvectors as \( \mathcal{T}(I, v_h, v_h) = \lambda_h v_h. \)
- **Bad news:** There can be other eigenvectors (unlike matrix case).
  
  E.g., when \( \{\lambda_h\}_{h=1}^K \equiv 1 \)
  
  \[ v = \frac{v_1 + v_2}{\sqrt{2}} \]
  satisfies \( \mathcal{T}(I, v, v) = \frac{1}{\sqrt{2}} v. \)
Spurious Eigenvectors for Tensor Eigen Decomposition

\[
\mathcal{T} = \sum_{h \in [K]} \lambda_h v_h \otimes^3.
\]

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

- \(\{v_h\}_{h=1}^K\) are eigenvectors as \(\mathcal{T}(I, v_h, v_h) = \lambda_h v_h\).
- **Bad news**: There can be other eigenvectors (unlike matrix case).
  
  E.g., when \(\{\lambda_h\}_{h=1}^K \equiv 1\)
  \[
  v = \frac{v_1 + v_2}{\sqrt{2}} \text{ satisfies } \mathcal{T}(I, v, v) = \frac{1}{\sqrt{2}} v.
  \]

How do we avoid spurious solutions (not components \(\{v_h\}_{h=1}^K\)?)
Spurious Eigenvectors for Tensor Eigen Decomposition

\[ T = \sum_{h \in [K]} \lambda_h v_h \otimes^3. \]

Characterization of eigenvectors: \( T(I, v, v) = \lambda v? \)

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

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

  \[ v = \frac{v_1 + v_2}{\sqrt{2}} \text{ satisfies } T(I, v, v) = \frac{1}{\sqrt{2}} v. \]

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

Optimization viewpoint of tensor Eigen decomposition will help.
Spurious Eigenvectors for Tensor Eigen Decomposition

\[ \mathcal{T} = \sum_{h \in [K]} \lambda_h v_h \otimes^3. \]

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

- \( \{v_h\}_{h=1}^K \) are eigenvectors as \( \mathcal{T}(I, v, v) = \lambda v_h \).
- **Bad news:** There can be other eigenvectors (unlike matrix case).
  
  E.g., when \( \{\lambda_h\}_{h=1}^K \equiv 1 \)

  \[ v = \frac{v_1 + v_2}{\sqrt{2}} \text{ satisfies } \mathcal{T}(I, v, v) = \frac{1}{\sqrt{2}} v. \]

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

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:**
  \[
  L(v, \lambda) := M(v, v) - \lambda(v^T v - 1).
  \]

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

- **Lagrangian:**
  \[
  L(v, \lambda) := T(v, v, v) - 1.5\lambda(v^T v - 1).
  \]
Optimization Viewpoint of Matrix/Tensor Eigen Decomposition

Optimization Problem
Matrix: $\max_v M(v, v) \text{ s.t. } \|v\| = 1.$

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

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

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

Non-convex: stationary points $= \{\text{global optima, local optima, saddle point}\}$
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 \( \nabla L(v, \lambda) = 0 \)

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

- 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

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

- 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:**

\[ w^\top \nabla^2 L(v, \lambda) w < 0 \]

- \( v_1 \) is the only local optimum.
- All other eigenvectors are saddle points.

- \( \{v_h\}_{h=1}^K \) are the only local optima.
- All spurious eigenvectors are saddle points.
Question: What about performance under noise?
Tensor Perturbation Analysis

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

\[ \hat{T} = T + \mathcal{E}, \quad T = \sum h \lambda_h v_h \otimes 3, \quad \| \mathcal{E} \| := \max_{x: \|x\| = 1} |\mathcal{E}(x, x, x)| \leq \epsilon. \]

**Theorem:** Let \( T \) be number of iterations. If

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

then output \((v, \lambda)\) (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. \( \lambda_1|c_1| > \lambda_2|c_2| \ldots \), \( c_i := \langle v_i, u_0 \rangle \), and \( u_0 \) is the (successful) initializer.
Tensor Perturbation Analysis

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

**Theorem:** Let \( T \) be number of iterations. If

\[ T \geq \log K + \log \log \frac{\lambda_{\text{max}}}{\epsilon}, \quad \epsilon < \frac{\lambda_{\text{min}}}{K}, \]

then output \((v, \lambda)\) (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.

- 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
- State-of-the-art (trace based) initialization (Li & Huang, 2018).
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
5. Algorithms for Tensor Decompositions
6. Tensor Decomposition for Neural Network Compression
7. Conclusion
Neural Network - Nonlinear Function Approximation

Success of Deep Neural Networks

- Image classification
- Speech recognition
- Text processing

- computation power growth
- enormous labeled data
Neural Network - Nonlinear Function Approximation

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

AlexNet, 8 layers (ILSVRC 2012)

11x11 conv, 96, /4, pool/2

5x5 conv, 256, pool/2

3x3 conv, 384

3x3 conv, 384

3x3 conv, 256, pool/2

fc, 4096

fc, 4096

fc, 1000

Revolution of Depth

AlexNet, 8 layers
(ILSVRC 2012)

VGG, 19 layers
(ILSVRC 2014)

GoogleNet, 22 layers
(ILSVRC 2014)

Revolution of Depth

AlexNet, 8 layers (ILSVRC 2012)  VGG, 19 layers (ILSVRC 2014)  ResNet, 152 layers (ILSVRC 2015)

Revolution of Depth

Engines of visual recognition

HOG, DPM
AlexNet (RCNN)
VGG (RCNN)
ResNet (Faster RCNN)*

34 shallow
8 layers
16 layers
101 layers

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

*with other improvements & more data

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 $\mathcal{T} \in \mathbb{R}^{I_0 \times I_1 \times \cdots \times I_{m-1}}$
Common Types of Tensor Decompositions

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

**CANDECOMP/PARAFAC (CP) Decomposition**

- Factorize a tensor into sum of rank-1 tensors
- Rank-1 tensor is defined as outer product of multiple vectors

$$\mathbf{T}_{i_0,\ldots,i_{m-1}} = \sum_{r=0}^{R-1} \mathbf{M}_{r,i_0}^{(0)} \cdots \mathbf{M}_{r,i_{m-1}}^{(m-1)}$$
Common Types of Tensor Decompositions

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

CANDECOMP/PARAFAC (CP) Decomposition

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

Tucker (TK) Decomposition

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

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

CANDDECOMP/PARAFAC (CP) Decomposition

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

Tucker (TK) Decomposition

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

Tensor-Train (TT) Decomposition

- Factorize a tensor into a number of interconnected lower-order tensors
- $\mathcal{T}_{i_0,\ldots,i_{m-1}} = \sum_{r_0=1}^{R_0-1} \cdots \sum_{r_{m-2}=1}^{R_{m-2}-1} \mathcal{T}_{i_0,r_0}^{(0)} \mathcal{T}_{r_0,i_1,r_1}^{(1)} \cdots \mathcal{T}_{r_{m-2},i_{m-1}}^{(m-1)}$
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 $\mathbf{K} \in \mathbb{R}^{H \times W \times S \times T}$

- Filter height/width $H/W$, No. of input/output channels $S/T$. 
Compression of Convolutional Layer w/ Tensor Decompositions

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

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

  \[
  \mathcal{K}_{i,j,s,t} = \sum_{r=0}^{R-1} \mathcal{K}_{s,r}^{(0)} \mathcal{K}_{i,j,r}^{(1)} \mathcal{K}_{r,t}^{(2)}
  \]

- No. of param.: \( H W S T \rightarrow (H W + S + T)R \)
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 TK Decomposition**

**TK**: Decompose $\mathcal{K}$ into 1 core tensor, 2 factor tensors

$$
\mathcal{K}_{i,j,s,t} = \sum_{r_s=0}^{R_s-1} \sum_{r_t=0}^{R_t-1} \mathcal{K}^{(0)}_{s,r_s} \mathcal{K}^{(1)}_{i,j,r_s,r_t} \mathcal{K}^{(2)}_{r_t,t}
$$

- No. of param.: $HWST \rightarrow SR_s + HW R_s R_t + R_t T$

TK decomposition
Compression of Convolutional Layer w/ Tensor Decompositions

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

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

**Kernel TT Decomposition**

- **TT:** Decompose $\mathbf{K}$ into 4 factor tensors

$$\mathbf{K}_{i,j,s,t} = \sum_{r_s=0}^{R_s-1} \sum_{r=0}^{R-1} \sum_{r_t=0}^{R_t-1} \mathbf{K}^{(0)}_{s,r_s,i,r} \mathbf{K}^{(1)}_{r,s,i,r} \mathbf{K}^{(2)}_{r,j,r_t} \mathbf{K}^{(3)}_{r_t,t}$$

- No. of param.: $HWST \rightarrow SR_s + HR_sR + WR_tR + R_tT$
Convolutional Kernel: $\mathbf{K} \in \mathbb{R}^{H \times W \times S \times T}$ tensorized to $\mathbf{K}' \in \mathbb{R}^{H \times W \times S_0 \times \ldots \times S_{m-1} \times T_0 \times \ldots \times T_{m-1}}$
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: $\mathbf{K} \in \mathbb{R}^{H \times W \times S \times T}$ tensorized to $\mathbf{K}' \in \mathbb{R}^{H \times W \times S_0 \times \cdots \times S_{m-1} \times T_0 \times \cdots \times T_{m-1}}$

- Tensorization: kernel reshaped to higher order tensor.
- $S = \prod_{i=0}^{m-1} S_i$ and $T = \prod_{i=0}^{m-1} T_i$. 
Tensorized Spectrum Preserving Compression of Neural Networks

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

- 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 $\mathbf{U} \in \mathbb{R}^{X \times Y \times S}$ tensorized to $\mathbf{U}' \in \mathbb{R}^{X \times Y \times S_0 \times \cdots \times S_{m-1}}$. 
Convolutional Kernel: \( \mathbf{K} \in \mathbb{R}^{H \times W \times S \times T} \) tensorized to \( \mathbf{K}' \in \mathbb{R}^{H \times W \times S_0 \times \cdots \times S_{m-1} \times T_0 \times \cdots \times T_{m-1}} \)

- 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 \( \mathbf{U} \in \mathbb{R}^{X \times Y \times S} \) tensorized to \( \mathbf{U}' \in \mathbb{R}^{X \times Y \times S_0 \times \cdots \times S_{m-1}} \).
- Output reshaped \( \mathbf{V} \in \mathbb{R}^{X \times Y \times T} \) to \( \mathbf{V}' \in \mathbb{R}^{X' \times Y' \times T_0 \times \cdots \times T_{m-1}} \).
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 \textbf{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 \( \mathbf{TK} \) Decomposition**

- Param. No.: \( HWST \rightarrow SR_s + HW R_s R_t + R_t T \rightarrow m(S^{\frac{1}{m}} + T^{\frac{1}{m}})R + HW R^{2m} \)
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 TT Decomposition**

- Param. No.: \( HWST \rightarrow SR_s + HR_sR + W R_tR + R_tT \rightarrow (m(ST)^{1/m} R + HW)R \)
Successful Compression of CIFAR10 Resnet-34 Network (Su, Li, Bhattacharjee & Huang, 2018)

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

- Testing accuracies of tensor methods under compression rates.
- The uncompressed network achieves 93.2% accuracy.
- CIFAR10 Resnet-34 has $4 \times 10^5$ 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)

<table>
<thead>
<tr>
<th># Epochs</th>
<th>Uncompressed</th>
<th>SPC-TT (E2E)</th>
<th>t-SPC-TT (Seq.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>4.22</td>
<td>0.66x</td>
<td>10.51x</td>
</tr>
<tr>
<td>0.3</td>
<td>6.23</td>
<td>0.64x</td>
<td>7.54x</td>
</tr>
<tr>
<td>0.5</td>
<td>9.01</td>
<td>0.83x</td>
<td>5.54x</td>
</tr>
<tr>
<td>1.0</td>
<td>17.3</td>
<td>0.74x</td>
<td>3.04x</td>
</tr>
<tr>
<td>2.0</td>
<td>30.8</td>
<td>0.59x</td>
<td>1.75x</td>
</tr>
</tbody>
</table>

- 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.
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
5. Algorithms for Tensor Decompositions
6. Tensor Decomposition for Neural Network Compression
7. Conclusion
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

- “Tensorized Spectrum Preserving Compression for Neural Networks”, by Jiahao Su, Jingling Li, Bobby Bhattacharjee and Furong Huang, 2018.


A general library with higher order tensor operations is coming soon.