SAND2019-8423C

Illustration by Chris Brigman

Low-Rank Tensor Approximation of Higher-Order Statistical Moments

Tamara G. Kolda
Sandia National Labs, USA

Joint with Samantha Sherman (University of Notre Dame, USA)

SIAM Conference on Applied Algebraic Geometry

Tuesday 9-Saturday 13 July 2019
Bern, Switzerland

Supported by the Laboratory Directed Research and Development program at Sandia National Laboratories, a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.

Thanks for the invitation to Switzerland!


Zermatt

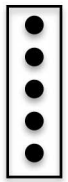
Lausanne

Sandia National Laboratories

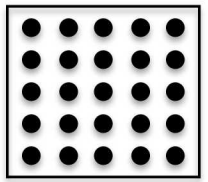
Low-Rank Tensor Decomposition

A Tensor is a Multi-Way Array

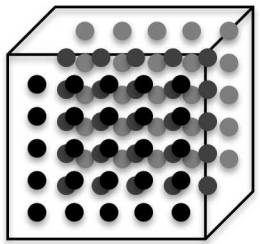
Vector
d = 1



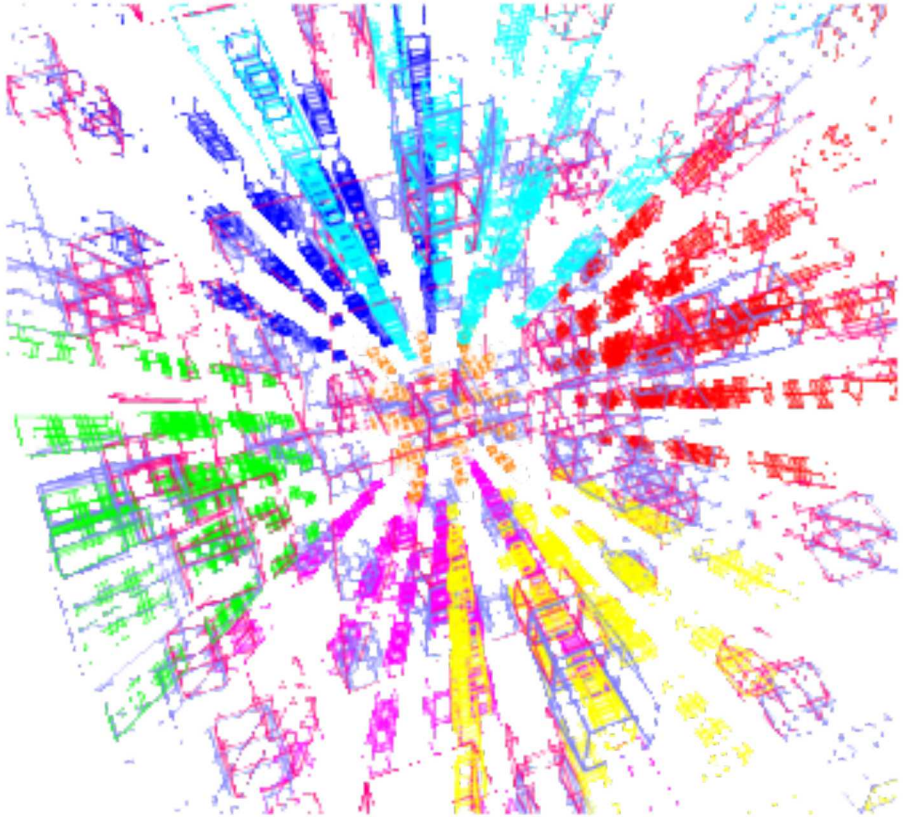
Matrix
d =2



3rd-order Tensor
d = 3

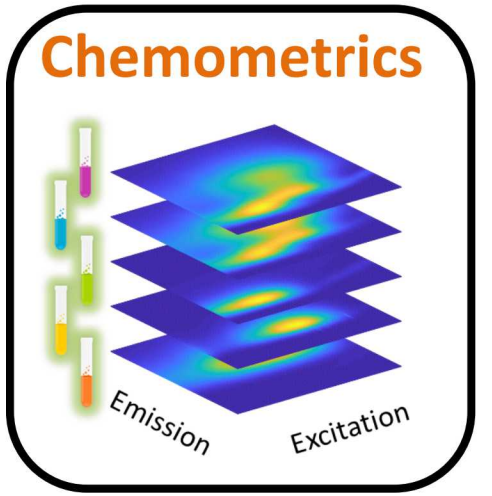


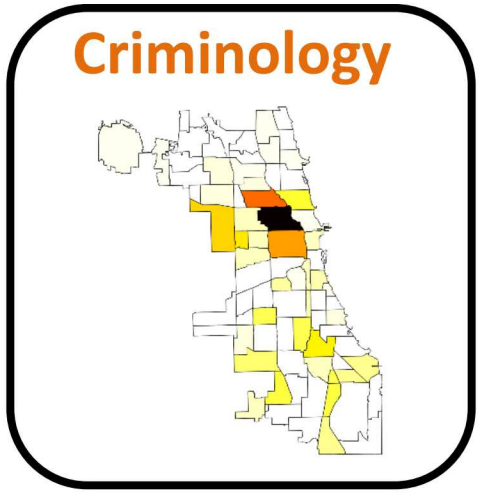
d^{th}-order Tensor
d > 3

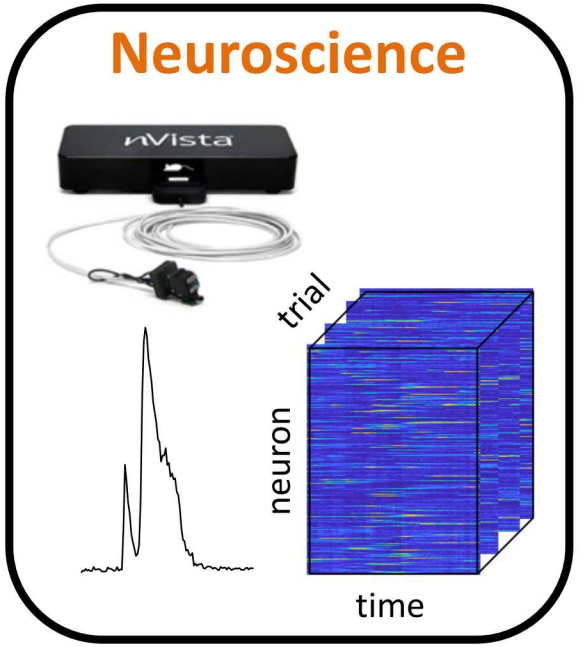


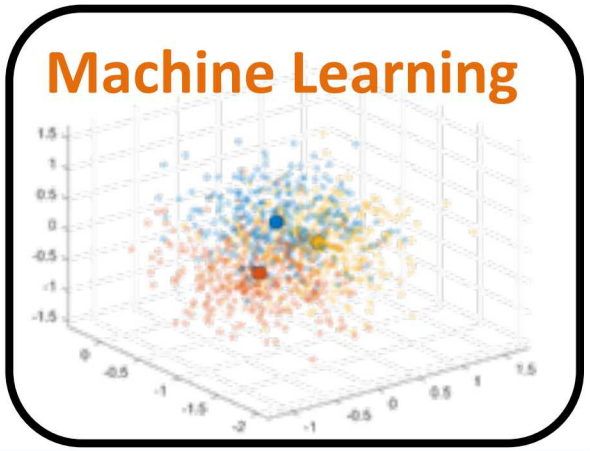
Tensors Come From Many Applications

- **Chemometrics:** Emission x Excitation x Samples (Fluorescence Spectroscopy)
- **Neuroscience:** Neuron x Time x Trial
- **Criminology:** Day x Hour x Location x Crime (Chicago Crime Reports)
- **Symmetric Higher-order Empirical Moments:** Multivariate Gaussian Distributions in Machine Learning
- **Transportation:** Pickup x Dropoff x Time (Taxis)
- **Sports:** Player x Statistic x Season (Basketball)
- **Cyber-Traffic:** IP x IP x Port x Time
- **Social Network:** Person x Person x Time x Interaction-Type
- **Symmetric Higher-order Derivatives:** From Optimization



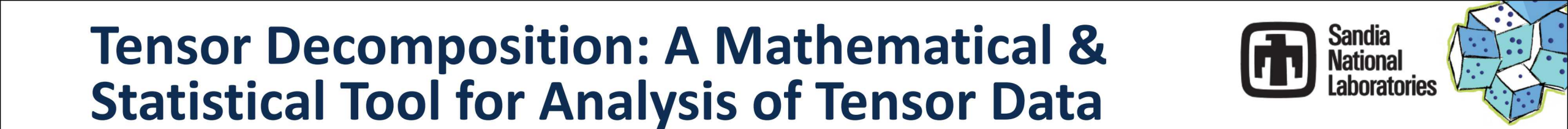




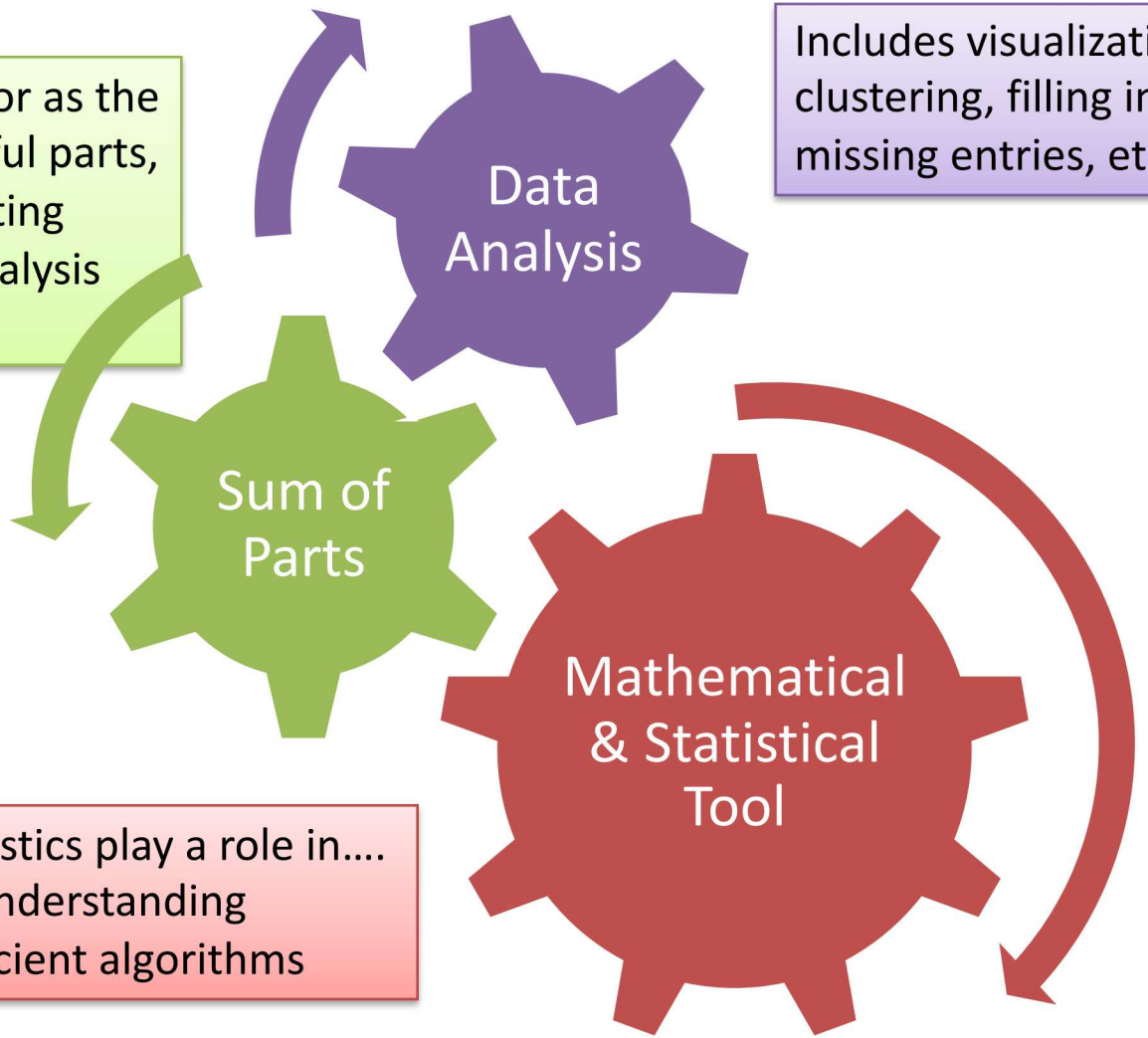


Tensor Decomposition: A Mathematical & Statistical Tool for Analysis of Tensor Data

Sandia National Laboratories



Express the tensor as the sum of meaningful parts, which is the starting point for data analysis activities



Includes visualization, clustering, filling in missing entries, etc.

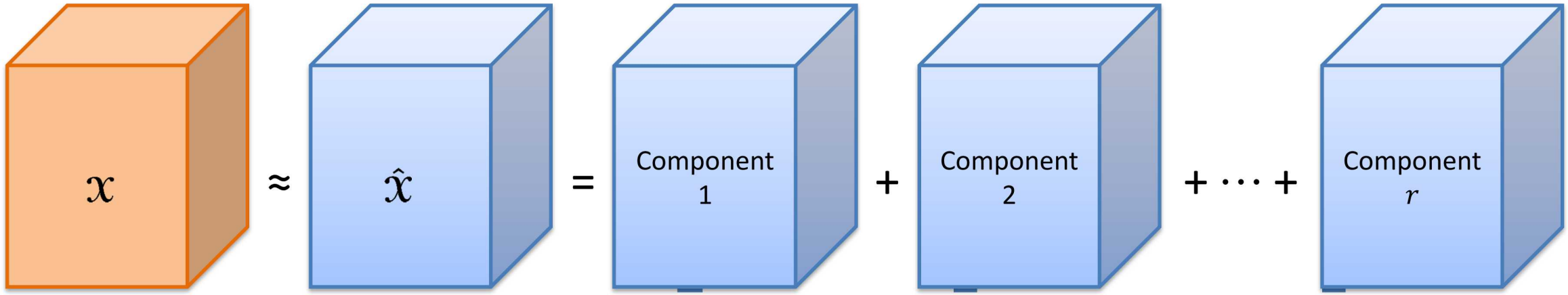
Mathematics/Statistics play a role in….

- Foundational understanding
- Developing efficient algorithms

Related Concepts for Matrices

- Singular value decomposition (SVD)
- Principal component analysis (PCA)
- Eigenvalue decomposition (EVD)
- Nonnegative matrix factorization (NMF)
- Sparse matrix factorization
- Matrix completion

Decompose into Understandable Components



Data Tensor

$n \times n \times n$

Model Tensor

$n \times n \times n$

$n \times n \times n$

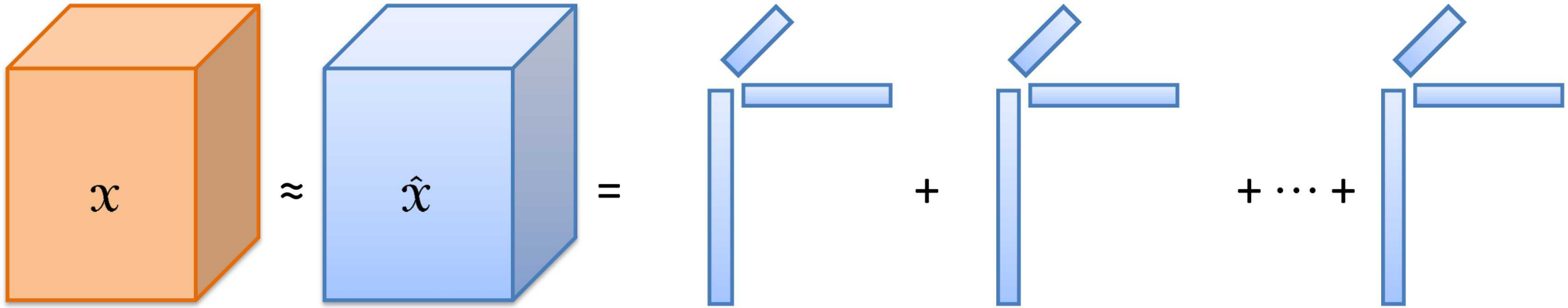
$n \times n \times n$

$n \times n \times n$

More generally, we allow each mode to have a different size:

$n_1 \times n_2 \times n_3$

Decompose into Understandable Components



Data Tensor

$n \times n \times n$

Model Tensor

$n \times n \times n$

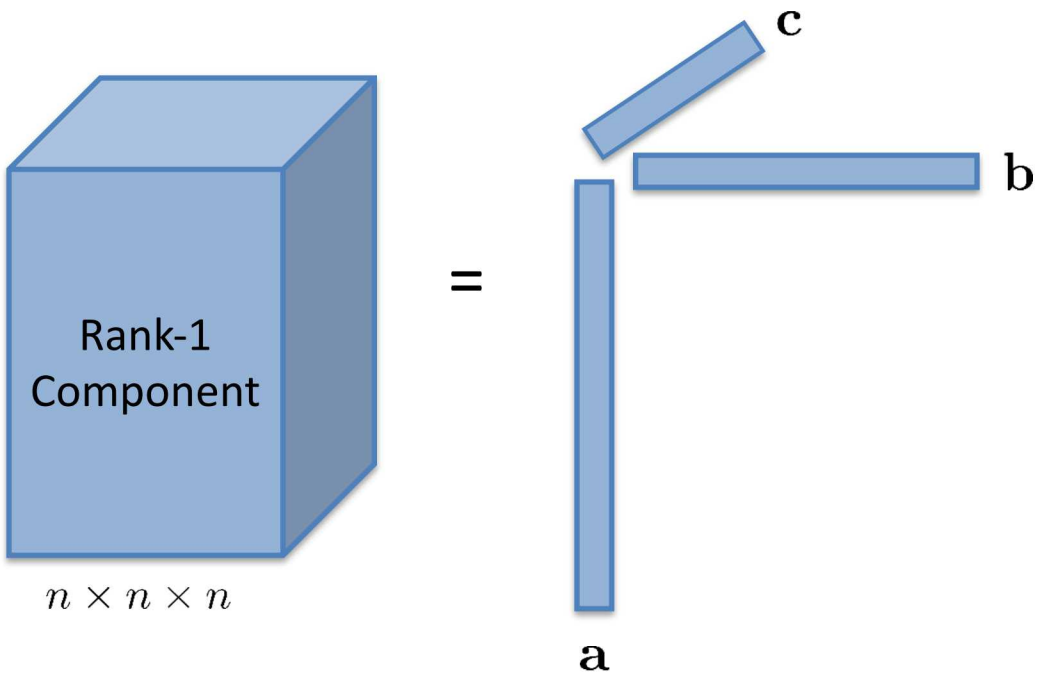
$n \times n \times n$

$n \times n \times n$

$n \times n \times n$

Key: The components have structure!

Rank-1 Tensors are the Components



$$(\mathbf{a} \otimes \mathbf{b} \otimes \mathbf{c})_{ijk} = a_i b_j c_k$$

Given ***d*** **vectors**:

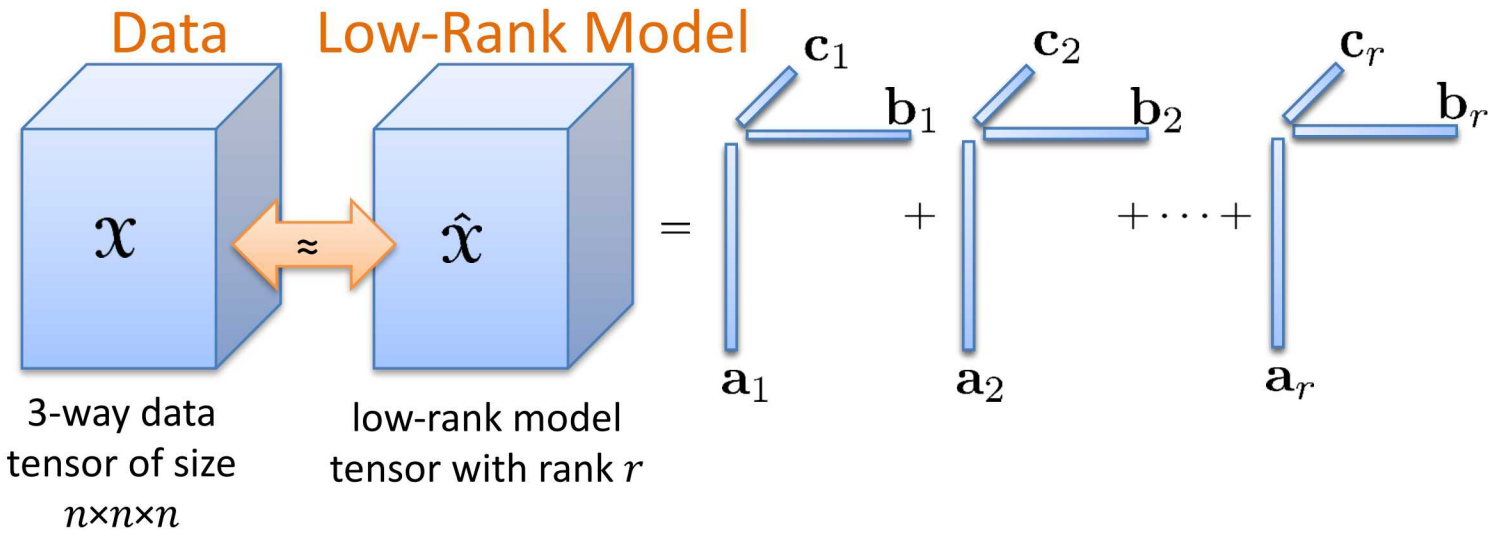
$$\mathbf{a}^{(k)} \in \mathbb{R}^{n_k} \text{ for } k = 1, \ldots, d$$

The **tensor outer product** is

$$\left(\mathbf{a}^{(1)} \otimes \mathbf{a}^{(2)} \otimes \cdots \otimes \mathbf{a}^{(d)}\right) \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$$

$$\left(\mathbf{a}^{(1)} \otimes \mathbf{a}^{(2)} \otimes \cdots \otimes \mathbf{a}^{(d)}\right)_{i_1 i_2 \ldots i_d} = \prod_{k=1}^{d} (\mathbf{a}^{(k)})_{i_k}$$

CANDECOMP/PARAFAC (CP) Tensor Factorization Uncovers the Rank-1 Parts



3-way data tensor of size $n \times n \times n$

low-rank model tensor with rank r

$$\mathcal{X} \approx \hat{\mathcal{X}} \quad \text{where} \quad \hat{\mathcal{X}} = \sum_{j=1}^{r} \mathbf{a}_j \otimes \mathbf{b}_j \otimes \mathbf{c}_j$$

Low-rank: $\operatorname{rank}(\hat{\mathcal{X}}) \leq r \ll n^3$

Hitchcock, 1927; Carroll and Chang, 1970; Harshman, 1970

CP First Invented in 1927

Frank Lauren Hitchcock
MIT Professor
(1875–1957)

THE EXPRESSION OF A TENSOR OR A POLYADIC AS A SUM OF PRODUCTS

By Frank L. Hitchcock

1. **Addition and Multiplication.**

Tensors are *added* by adding corresponding components. The *product* of a covariant tensor $A_{i_1 \cdots i_p}$ of order p *into* a covariant tensor $B_{i_{p+1} \cdots i_{p+q}}$ of order q is defined by writing

$$A_{i_1 \cdots i_p} B_{i_{p+1} \cdots i_{p+q}} = C_{i_1 \cdots i_{p+q}} \tag{1}$$

where the product $C_{i_1 \cdots i_{p+q}}$ is a covariant tensor of order $p+q$. When no confusion results indices may be omitted giving

$$\mathbf{AB} = \mathbf{C} \tag{1_a}$$

equivalent to the n^{p+q} equations (1). Boldface type is convenient for indicating that the letters do not denote merely numbers or scalars. Products of contravariant and of mixed tensors may be similarly defined.

A partial statement of the problem to be considered is as follows: to find under what conditions a given tensor can be expressed as a sum of products of assigned form. A more general statement of the problem will be given below.

2. **Polyadic form of a tensor.**

Any covariant tensor $A_{i_1 \cdots i_p}$ can be expressed as the sum of a finite number of tensors each of which is the product of p covariant vectors.

$$A_{i_1 \cdots i_p} = \sum_{j=1}^{j=h} a_{1j, i_1} a_{2j, i_2} \cdots a_{pj, i_p} \tag{2}$$

where a_{ij, i_r}, etc., are a set of hp covariant vectors. When the indices $i_1 \cdots i_p$ can be omitted this may be written

$$\mathbf{A} = \sum_{j=1}^{j=h} \mathbf{a}_{1j} \mathbf{a}_{2j} \cdots \mathbf{a}_{pj}. \tag{2_a}$$

The right member is now identical in appearance with a Gibbs

F. L. Hitchcock, ***The Expression of a Tensor or a Polyadic as a Sum of Products***, Journal of Mathematics and Physics, 1927

2. **Polyadic form of a tensor.**

Any covariant tensor $A_{i_1 \cdots i_p}$ can be expressed as the sum of a finite number of tensors each of which is the product of p covariant vectors.

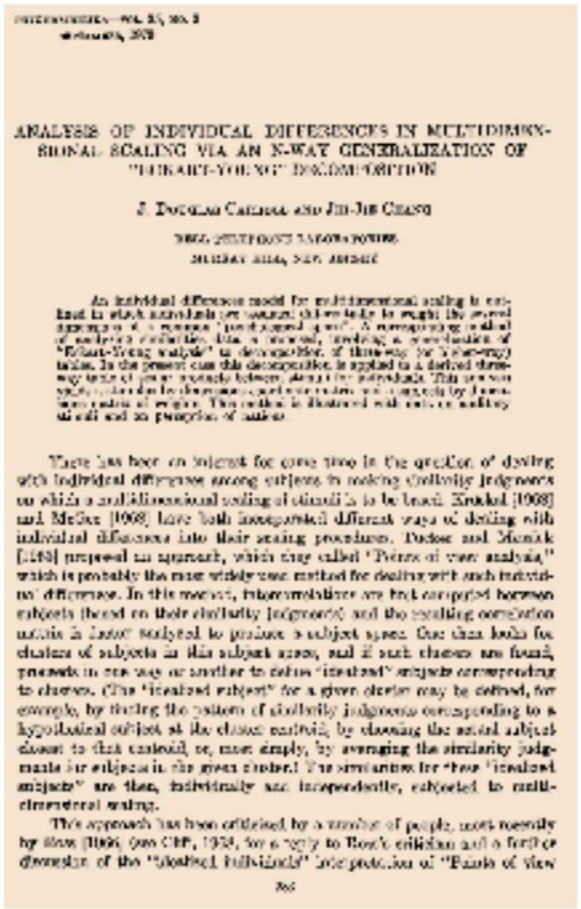
$$A_{i_1 \cdots i_p} = \sum_{j=1}^{j=h} a_{1j, i_1} a_{2j, i_2} \cdots a_{pj, i_p} \tag{2}$$

where a_{ij, i_r}, etc., are a set of hp covariant vectors. When the indices $i_1 \cdots i_p$ can be omitted this may be written

$$\mathbf{A} = \sum_{j=1}^{j=h} \mathbf{a}_{1j} \mathbf{a}_{2j} \cdots \mathbf{a}_{pj}. \tag{2_a}$$

CP Independently Reinvented (twice) in 1970

CANDECOMP: Canonical Decomposition



ANALYSIS OF INDIVIDUAL DIFFERENCES IN MULTIDIMENSIONAL SCALING VIA AN N-WAY GENERALIZATION OF "ECKART-YOUNG" DECOMPOSITION

J. DOUGLAS CARROLL AND JIH-JIE CHANG

J. Douglas Carroll
Bell Labs
(1939-2011)

Jih-Jie Chang
Bell Labs
(1927-2007)

PARAFAC: Parallel Factors

Richard A. Harshman
Univ. Ontario
(1943-2008)

FOUNDATIONS OF THE PARAFAC PROCEDURE: MODELS AND CONDITIONS FOR AN "EXPLANATORY" MULTIMODAL FACTOR ANALYSIS

by

Richard A. Harshman

UCLA

Working Papers in Phonetics

16

December 1970

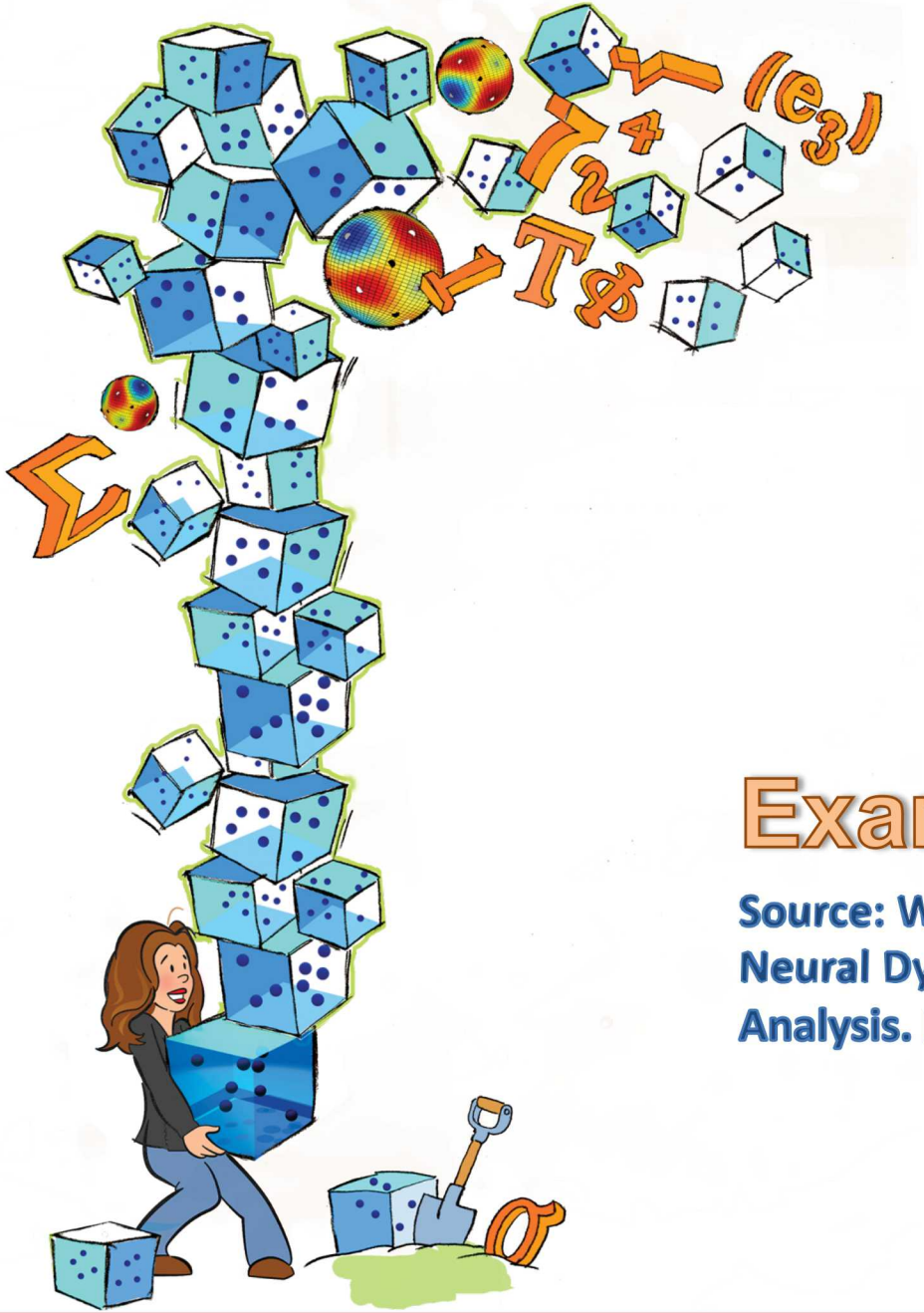
CP: CANDECOMP/PARAFAC

In 2000, Henk Kiers proposed this *compromise* name

CP: Canonical Polyadic

2010: Pierre Comon, Lieven DeLathauwer, and others reverse-engineered CP, revising some of Hitchcock's terminology

Many thanks to the following persons for helping me learn about Jih-Jie Chang: Fan Chung, Ron Graham, Shen Lin (husband), May Chang (niece), Lili Bruer (daughter).



Example Tensor from Neuroscience

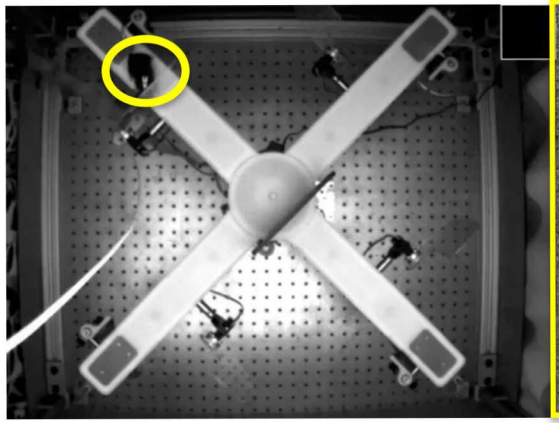
Source: Williams, et al. Unsupervised Discovery of Demixed, Low-dimensional Neural Dynamics across Multiple Timescales through Tensor Components Analysis. Neuron, 2018. https://doi.org/10.1016/j.neuron.2018.05.015

Activity of Single Neuron Measured Over Time Produces Vector Data

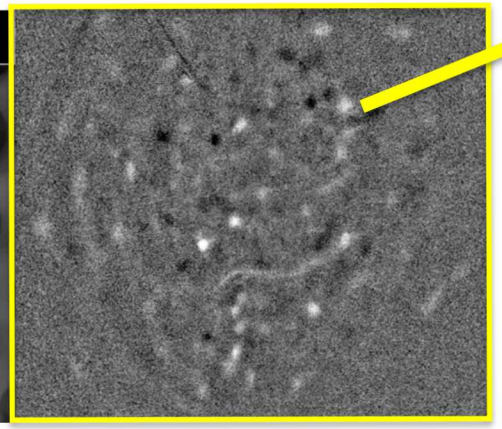
Thanks to Schnitzer Group @ Stanford
Mark Schnitzer, Fori Wang, Tony Kim

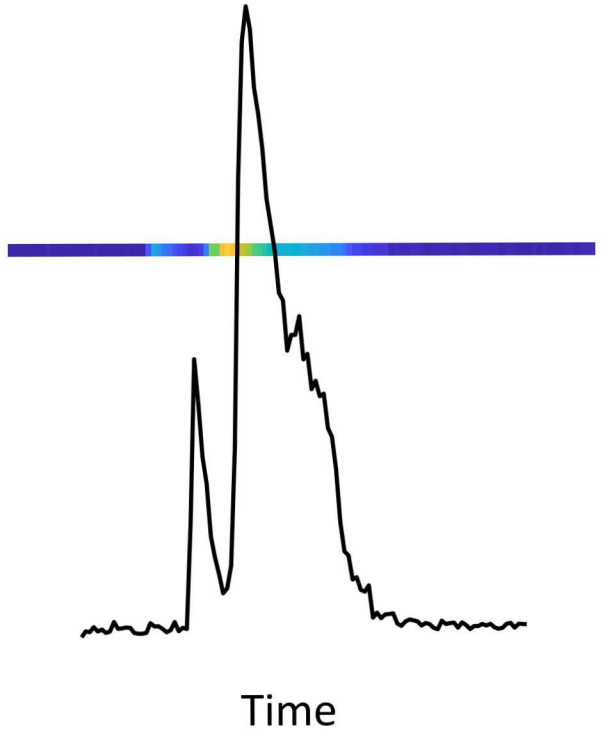
Microscope by Inscopix

mouse in maze



neural activity via calcium imaging





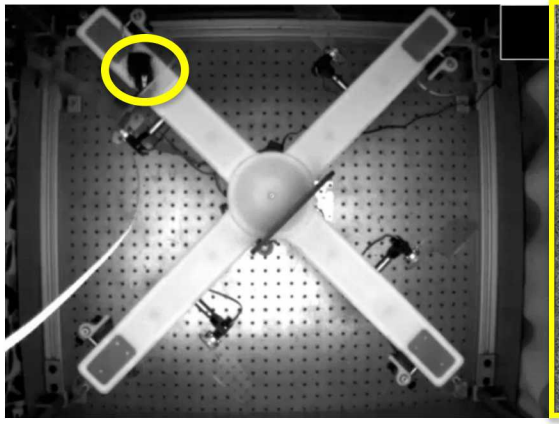
Williams et al., Neuron, 2018

Multiple Neurons Measured Over Time Produces Matrix

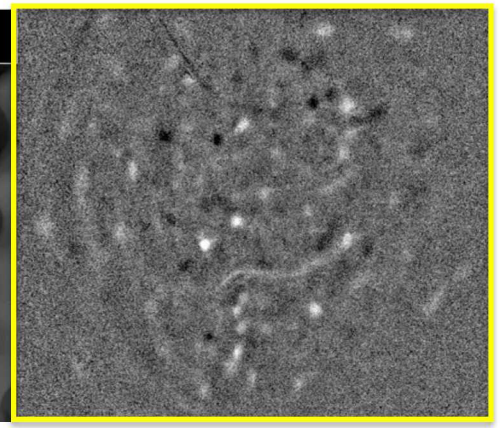
Thanks to Schnitzer Group @ Stanford
Mark Schnitzer, Fori Wang, Tony Kim

Microscope by Inscopix

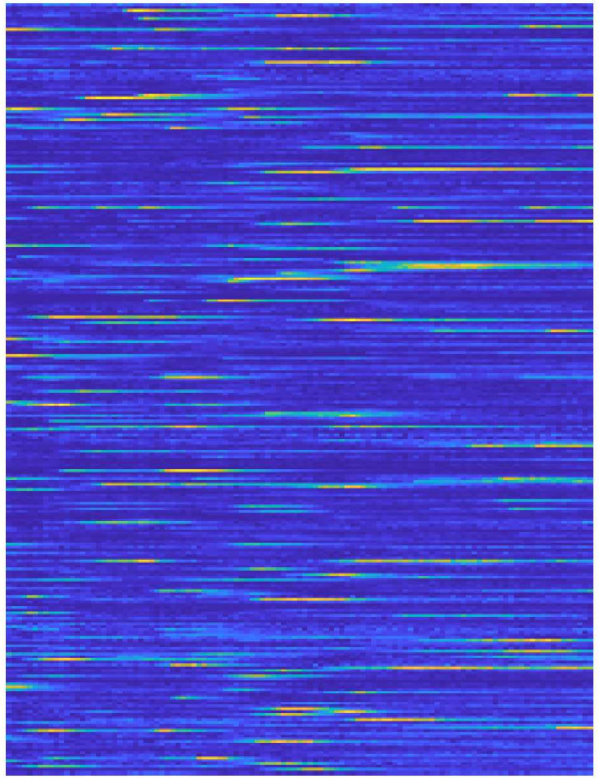
mouse in maze



neural activity via calcium imaging

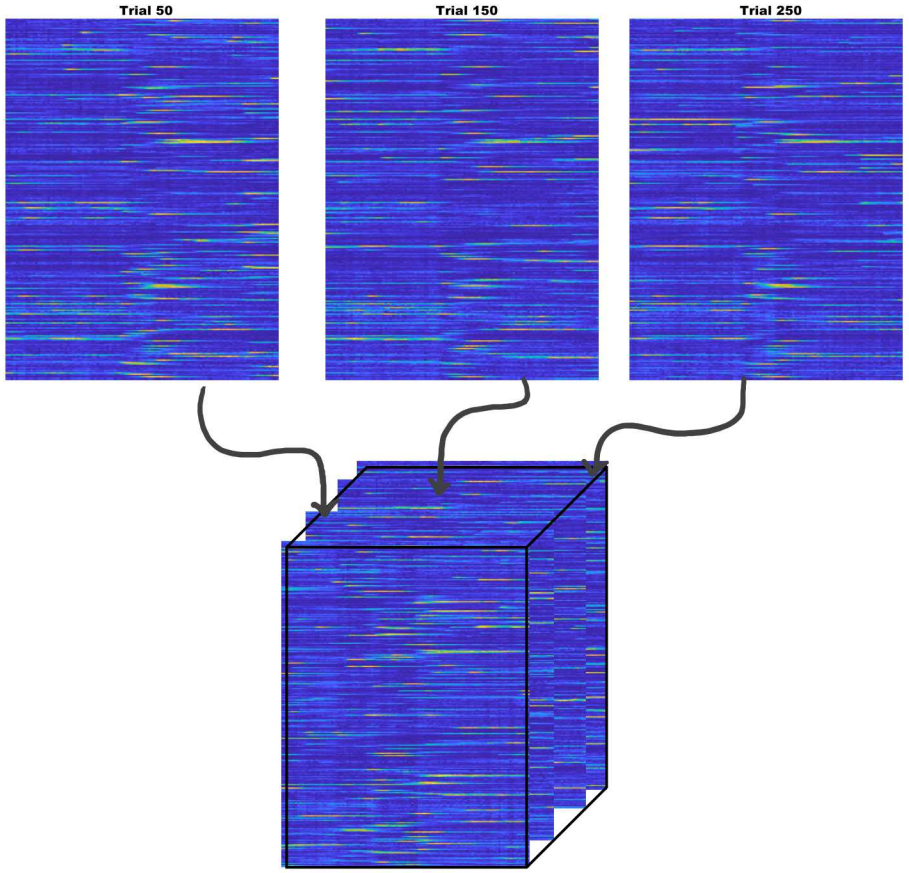


282 neurons × 111 time bins

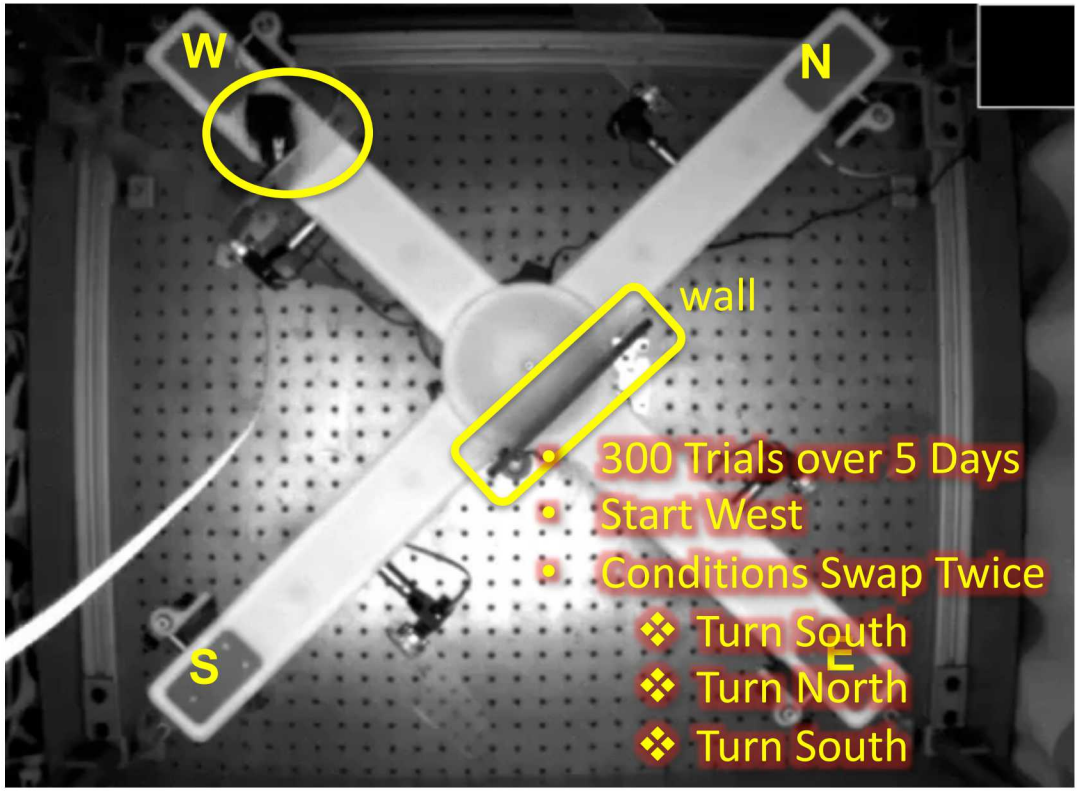


Williams et al., Neuron, 2018

Multiple Trials Produces 3-way Tensor



282 neurons × 111 time bins × 300 trials

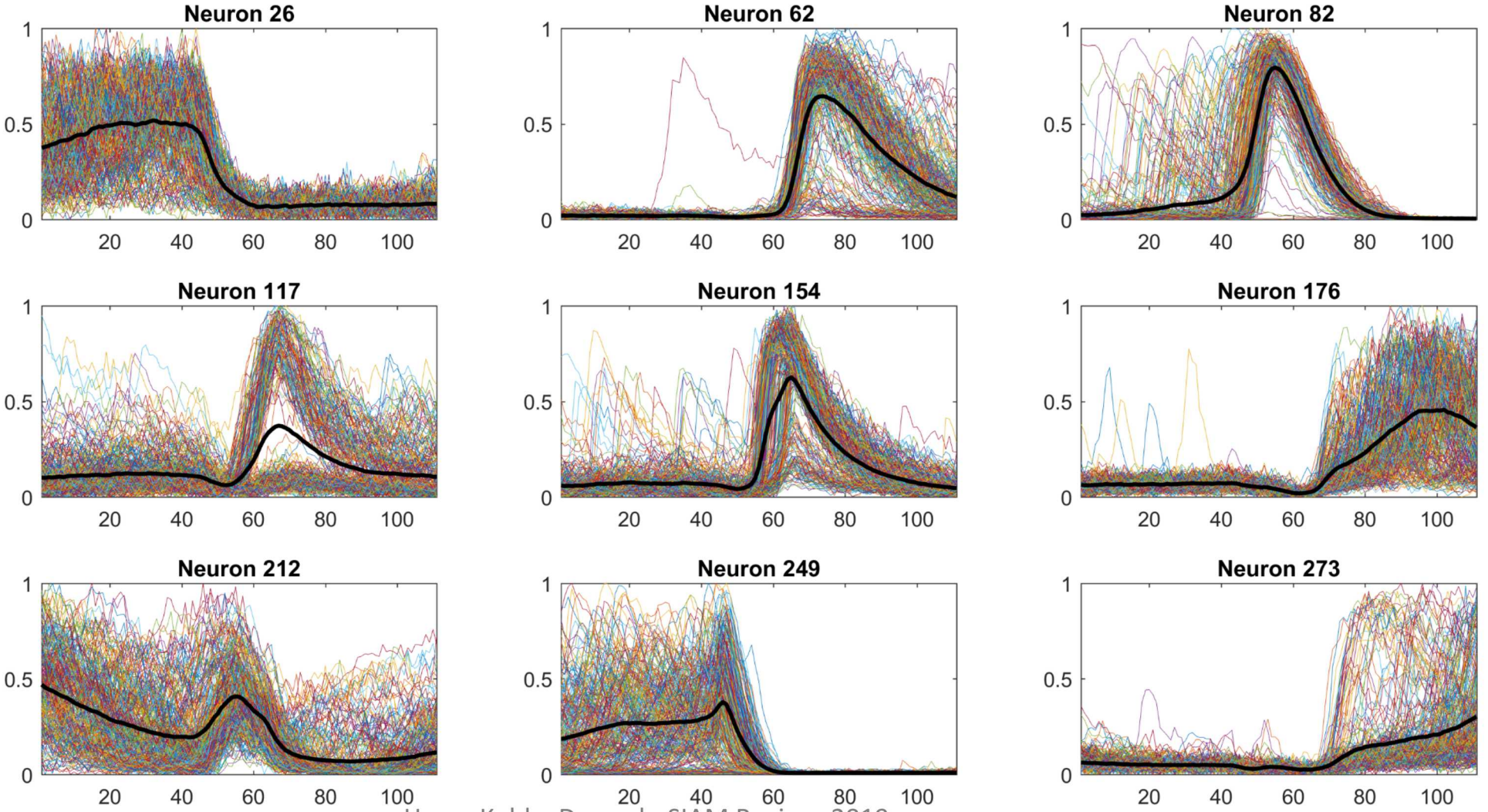


Williams et al., Neuron, 2018

Example Neuron Activity

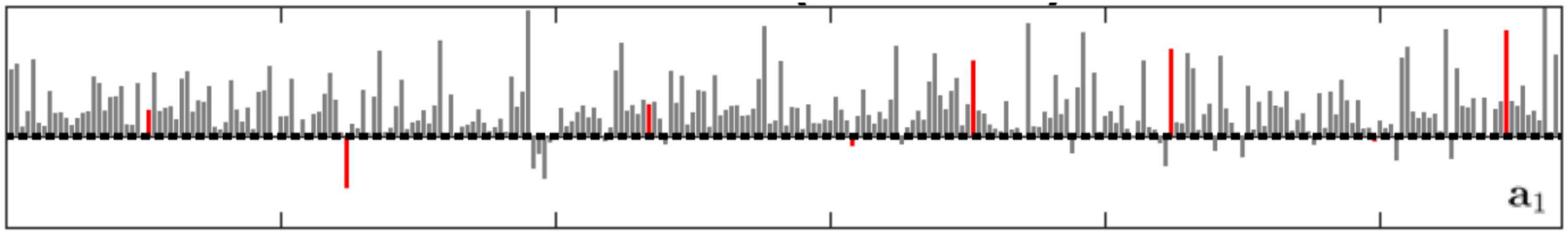
Thin lines show 300 individual trials

Thick line is average

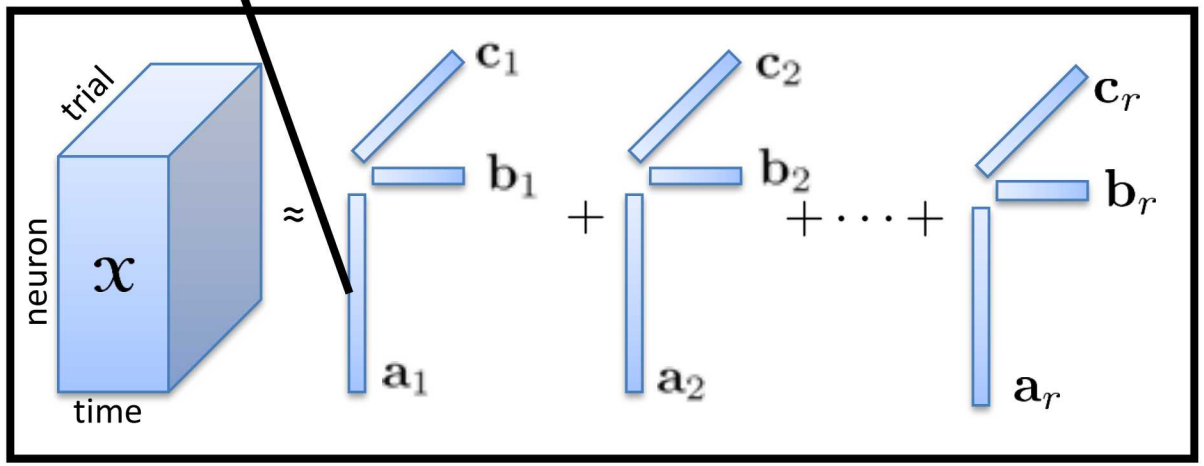


Hong, Kolda, Duersch, SIAM Review, 2019

Neuron Factor Vector Visualized as Bar Chart

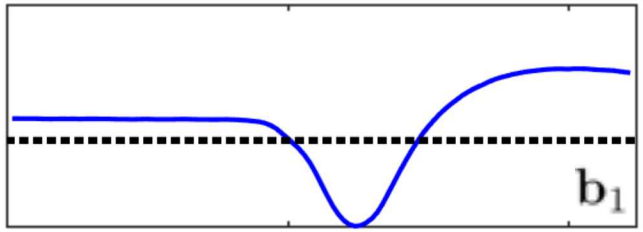


Neuron Modes Plotted as a Bar Chart
(Red Lines Correspond to Examples in Previous Slide)

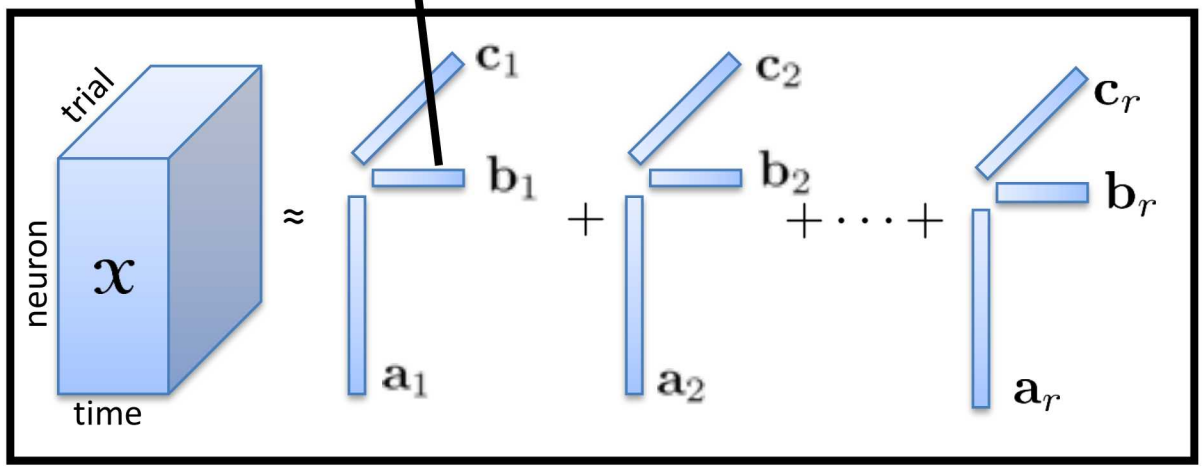


Hong, Kolda, Duersch, SIAM Review, 2019

Time Factor Vector Visualized as Line

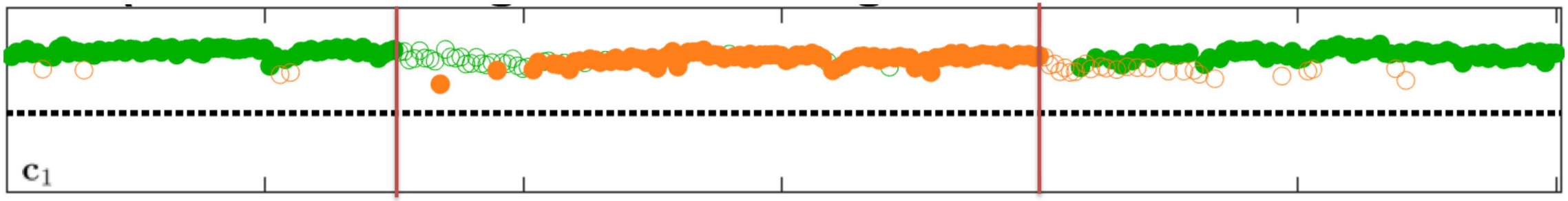


Time (within trial) Plotted as a Line
(Dashed Line is Zero)



Hong, Kolda, Duersch, SIAM Review, 2019

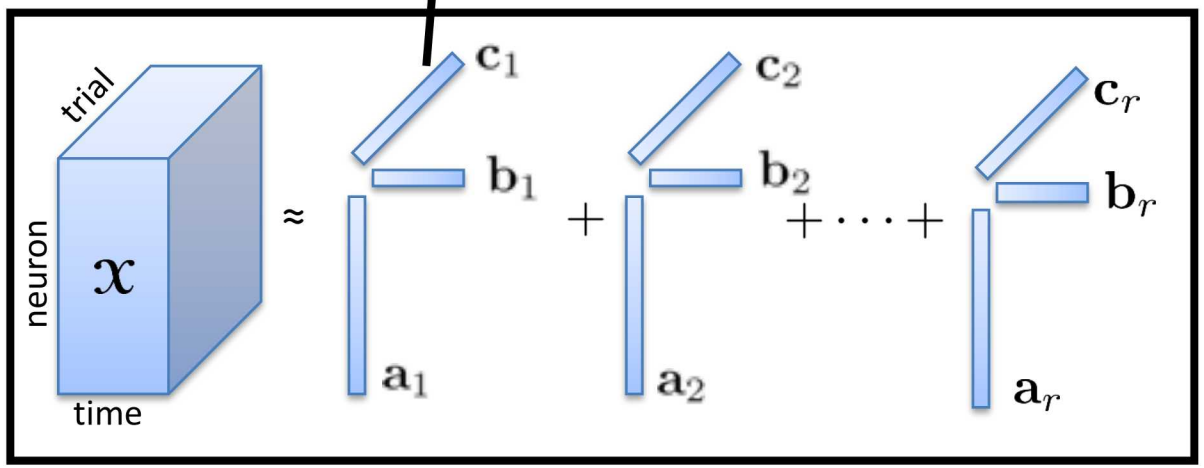
Trial Factor Vector Visualized as Color-Coded Scatter Plot



Rule Change

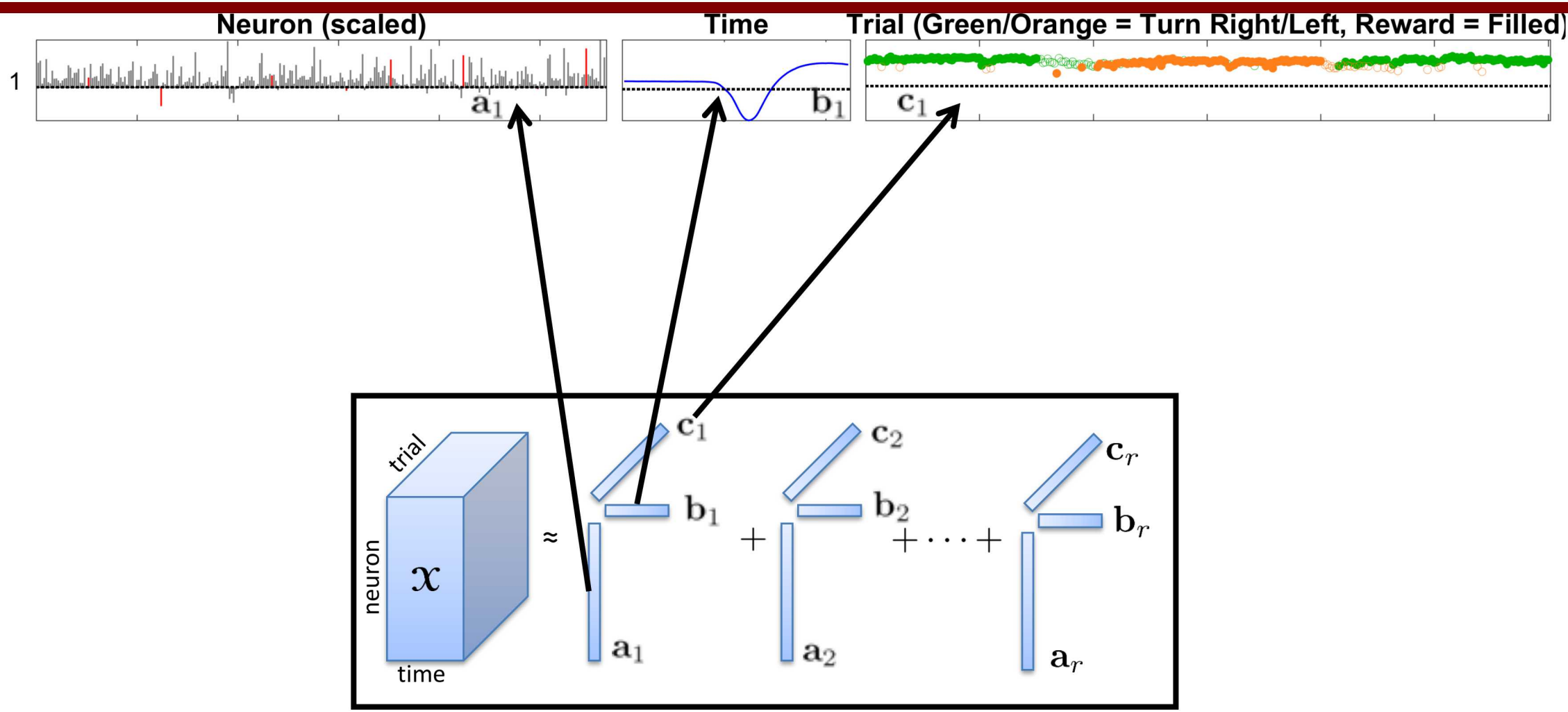
Trial Plotted as Scatter Graph
Right turn = Green
Left turn = Orange
Filled = Reward

Rule Change



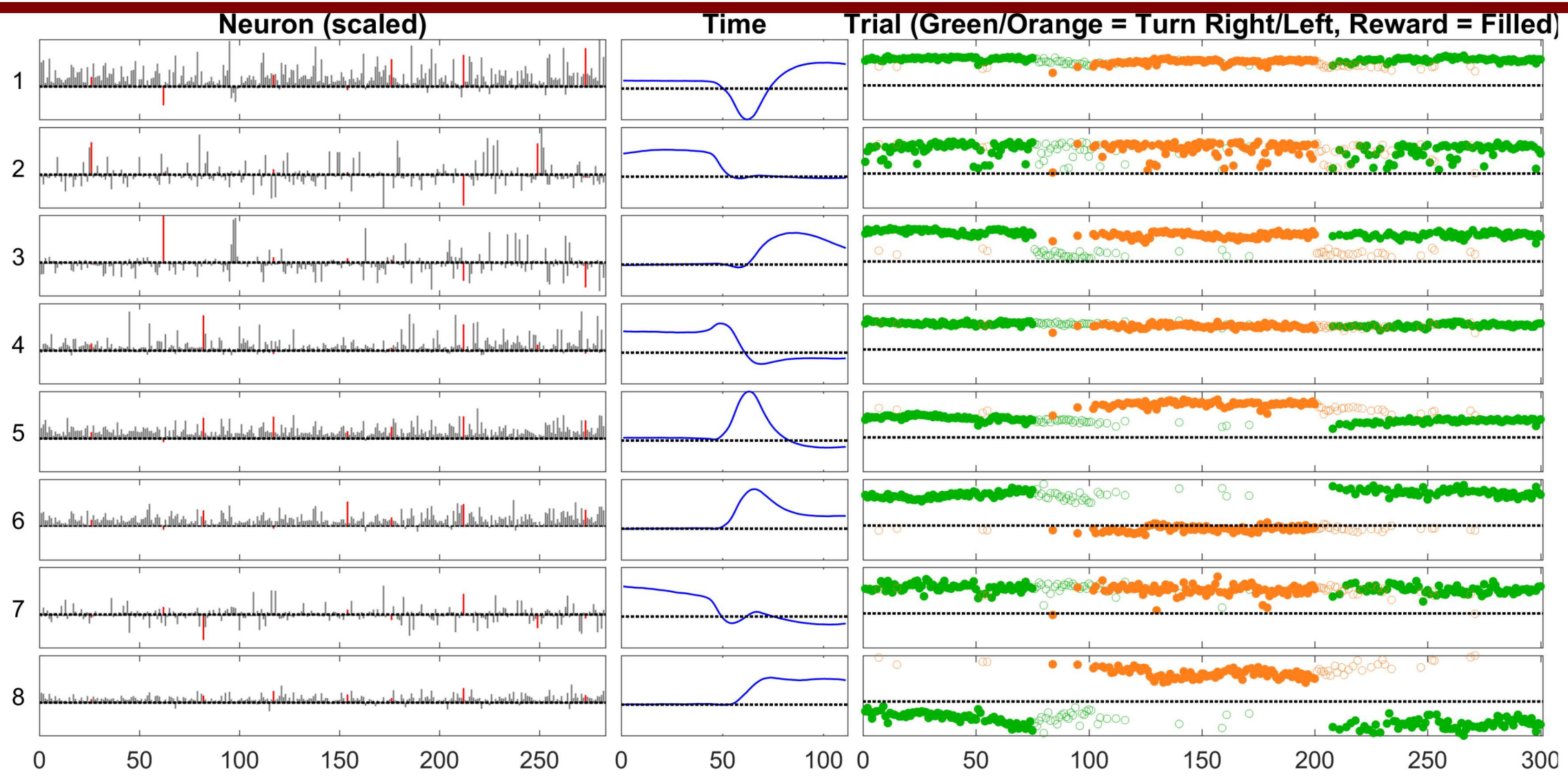
Hong, Kolda, Duersch, SIAM Review, 2019

Visualization of CP Tensor Decomposition Shows the Factors (Vectors)

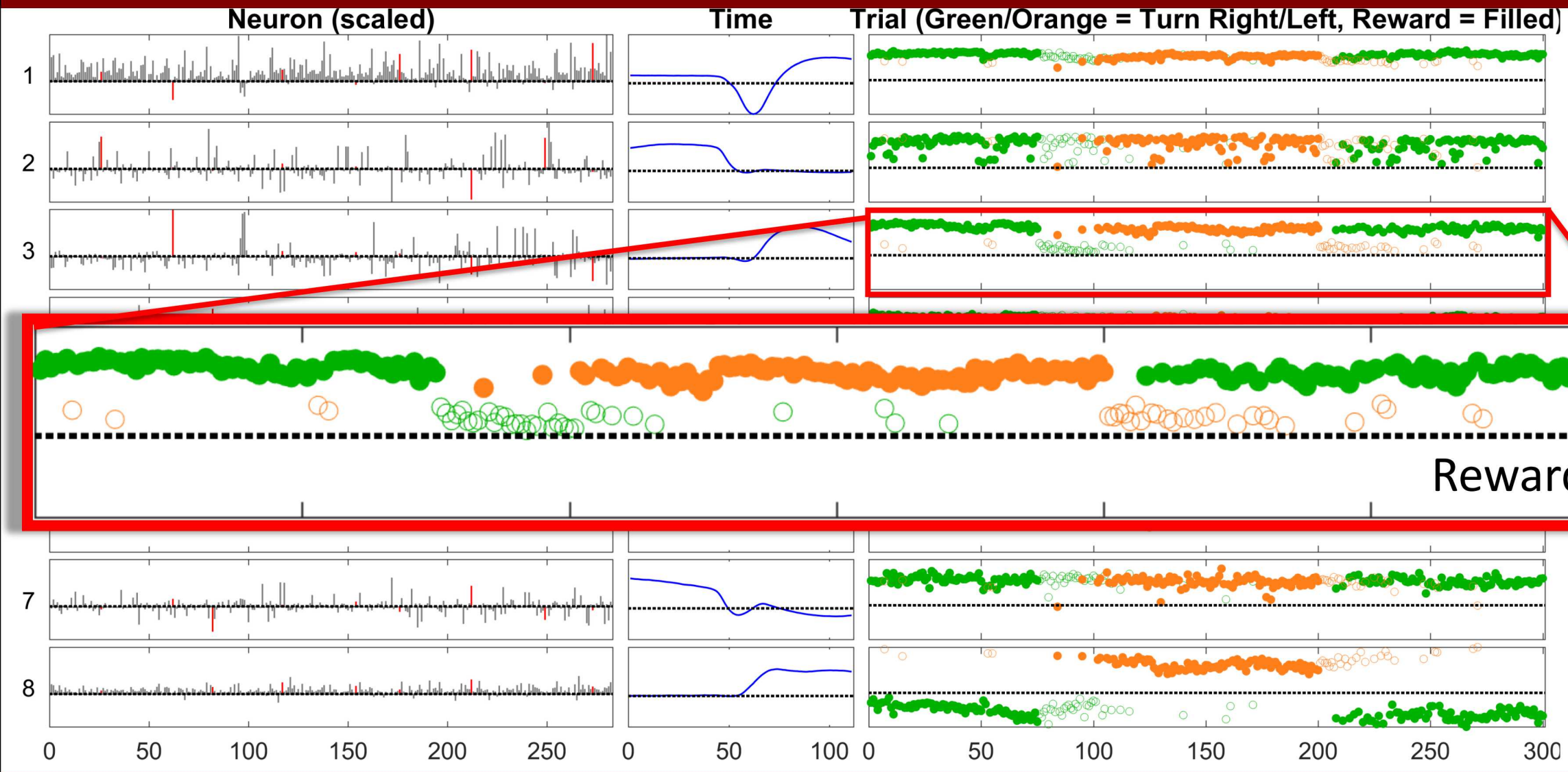


Hong, Kolda, Duersch, SIAM Review, 2019

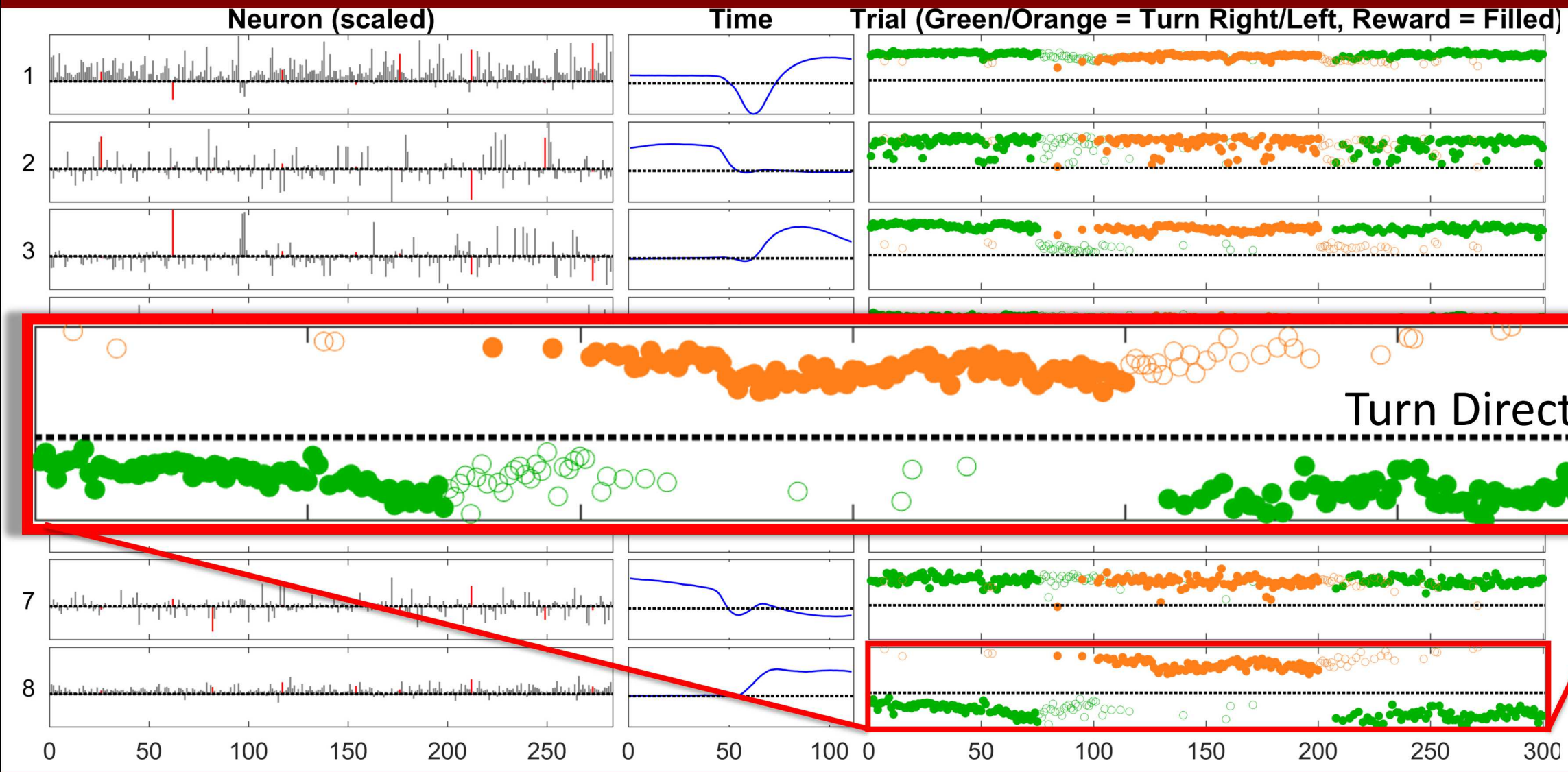
CP Decomposition of Mouse Data



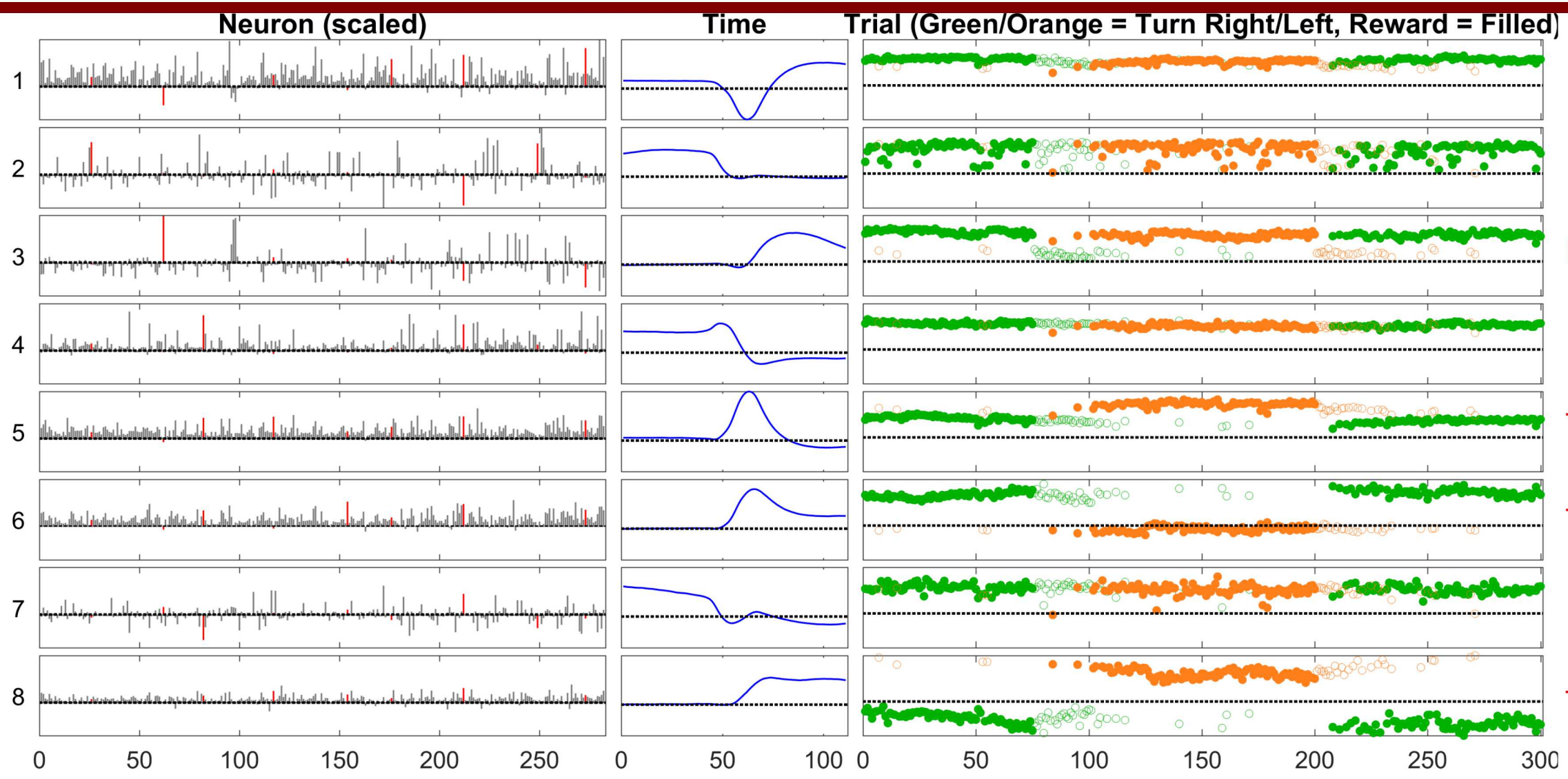
CP Tensor Decomposition "Sees" Reward



CP Tensor Decomposition "Sees" Turn Direction



CP Tensor Decomposition Can be Tough to Interpret due to Negative Entries



Quick Aside: Generalized CP Tensor Decomposition

Generalized CP

Standard CP Loss Metric

$$\|\mathcal{X} - \hat{\mathcal{X}}\|_F^2 = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}(x_{ijk} - \hat{x}_{ijk})^2$$

Generalized CP (GCP) Loss Metric

$$F(\mathcal{X}, \hat{\mathcal{X}}) = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} f(x_{ijk}, \hat{x}_{ijk})$$

Example Loss Metrics

Sum of squared error (SSE)

$$f(x, \hat{x}) = (x - \hat{x})^2$$

Poisson MLE (Kullback-Leibler divergence)

$$f(x, \hat{x}) = \hat{x} - x \log \hat{x} \quad (x \in \mathbb{N}, \hat{x} > 0)$$

Bernoulli MLE (odds link)

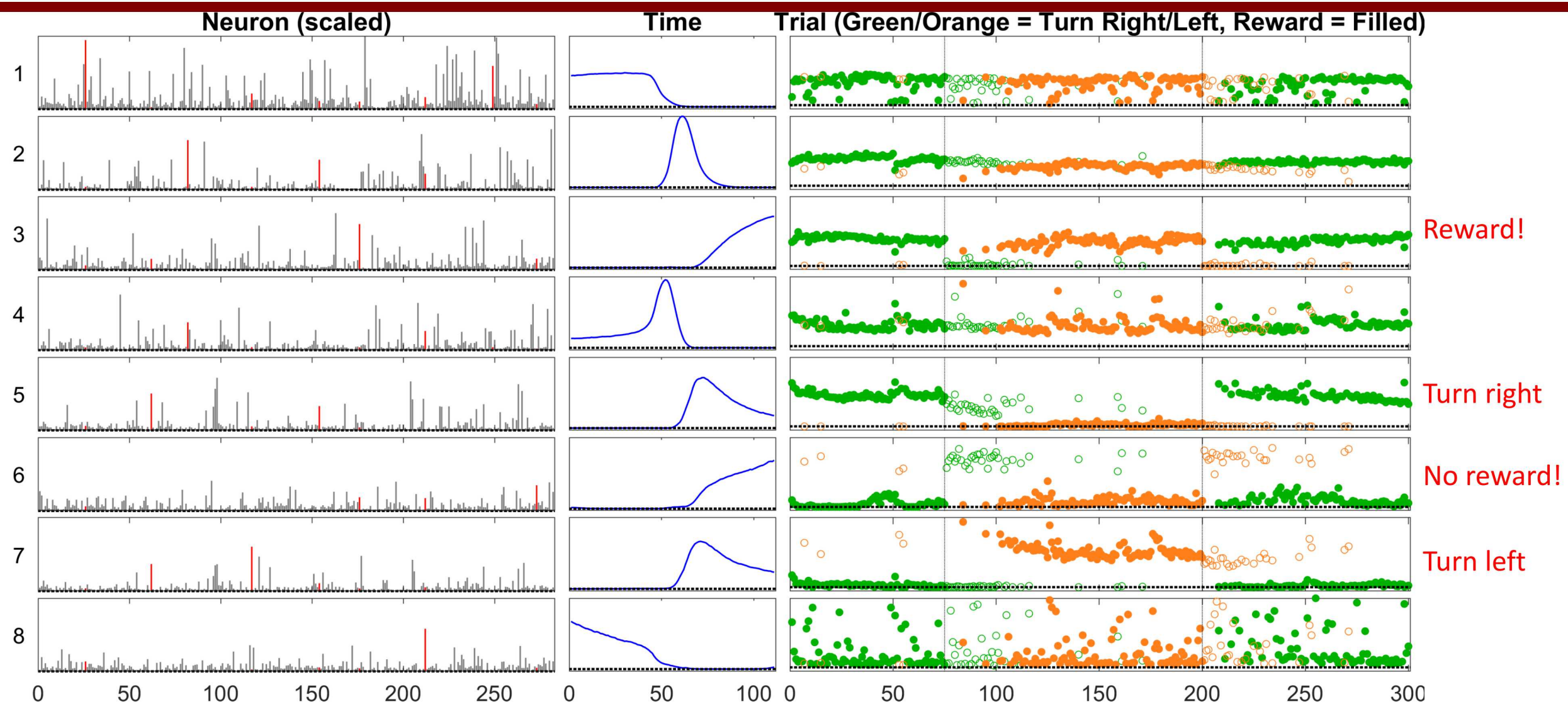
$$f(x, \hat{x}) = \log(\hat{x} + 1) - x \log \hat{x} \quad (x \in \{0, 1\}, \hat{x} > 0)$$

Beta Divergence

$$f(x, \hat{x}) = \frac{m^\beta}{\beta} - \frac{x m^{\beta-1}}{\beta - 1} \quad (x > 0, \hat{x} > 0)$$

Hong, Kolda, Duersch, SIAM Review, 2019

Generalized CP Decomposition (Beta Divergence, $\beta = 0.5$)



Symmetric CP Tensor Decomposition

Symmetric Tensors

A tensor is symmetric if its entries are invariant under permutation of the indices

$$a_{ijk} = a_{ikj} = a_{jik} = a_{kij} = a_{jki} = a_{kji}$$

For d-way tensor, $d!$ permutations of the d indices

Number of unique entries:

$$\binom{n+d-1}{d} \approx \frac{n^d}{d!}$$

Example 1.2 from Nie (2014)
3×3×3 symmetric tensor (10 distinct entries)

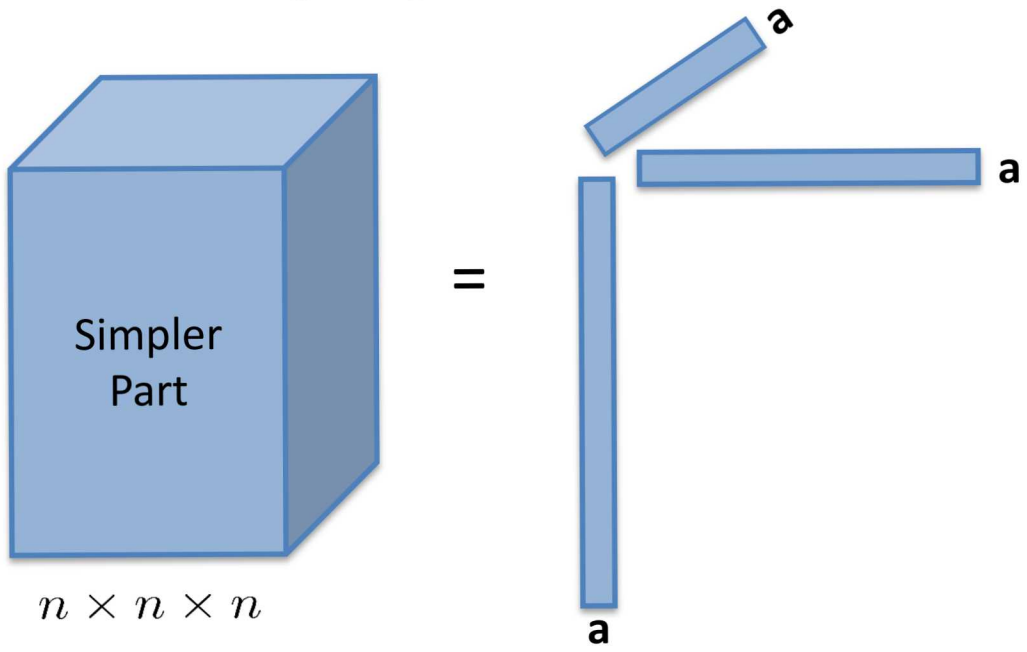
$$\mathcal{X} = \left(\begin{array}{ccc|ccc|ccc} 7 & -3 & 9 & -3 & 13 & 20 & 9 & 20 & 19 \\ -3 & 13 & 20 & 13 & -27 & 6 & 20 & 6 & 6 \\ 9 & 20 & 19 & 20 & 6 & 6 & 19 & 6 & 45 \end{array}\right)$$

```
Xsym = symtensor(X)

Xsym is a symmetric tensor with 3 modes of dimension 3
        (1,1,1)     7
        (1,1,2)    -3
        (1,1,3)     9
        (1,2,2)    13
        (1,2,3)    20
        (1,3,3)    19
        (2,2,2)   -27
        (2,2,3)     6
        (2,3,3)     6
        (3,3,3)    45
```

Symmetric Rank-1 Tensors are the "Parts"

$$\mathbf{a}^{\otimes 3} \equiv \mathbf{a} \otimes \mathbf{a} \otimes \mathbf{a}$$



$n \times n \times n$

$$(\mathbf{a}^{\otimes 3})_{ijk} = a_i a_j a_k$$

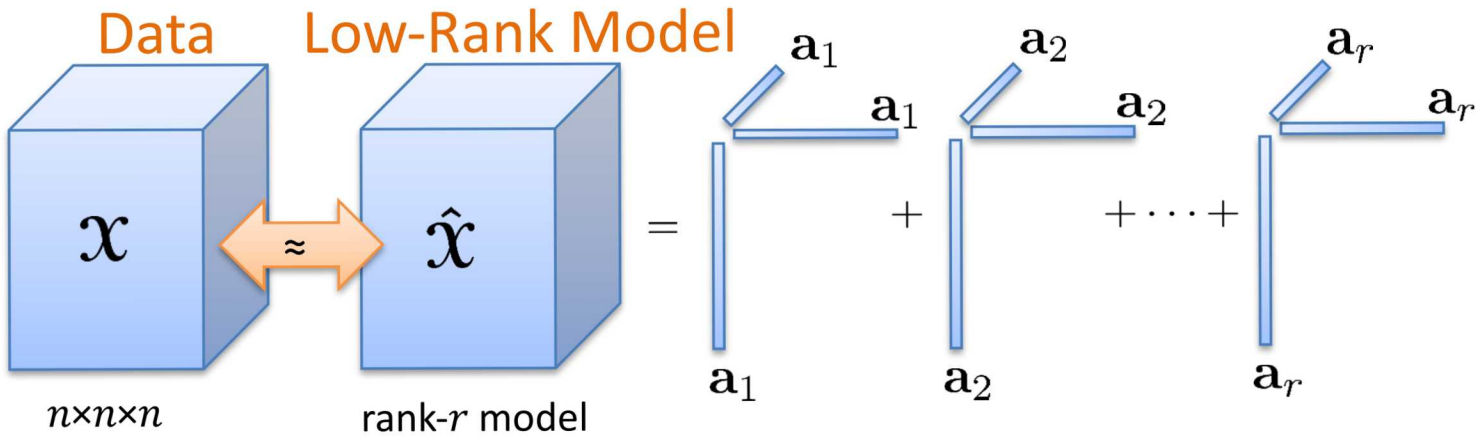
Given **a vector**:

$$\mathbf{a} \in \mathbb{R}^n$$

The **outer product** is

$$\mathcal{P} = \mathbf{a}^{\otimes d} \in \mathbb{R}^{n \times n \times \cdots \times n}$$

Symmetric CP Tensor Factorization



$$\mathcal{X} \approx \hat{\mathcal{X}} \quad \text{where} \quad \hat{\mathcal{X}} = \sum_{j=1}^{r} \lambda_j \, \mathbf{a}_j \otimes \mathbf{a}_j \otimes \mathbf{a}_j = \sum_{j=1}^{r} \lambda_j \, \mathbf{a}_j^{\otimes 3}$$

Variables that define the low-rank model:

$$\boldsymbol{\lambda} = \begin{bmatrix} \lambda_1 & \lambda_2 & \cdots \lambda_r \end{bmatrix} \in \mathbb{R}^r$$

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \cdots \mathbf{a}_r \end{bmatrix} \in \mathbb{R}^{n \times r}$$

Symmetric Tensor Rank & Decomposition

Example 1.2 from Nie (2014)
3×3×3 symmetric tensor (10 distinct entries)

$$\mathcal{X} = \left(\begin{array}{ccc|ccc|ccc} 7 & -3 & 9 & -3 & 13 & 20 & 9 & 20 & 19 \\ -3 & 13 & 20 & 13 & -27 & 6 & 20 & 6 & 6 \\ 9 & 20 & 19 & 20 & 6 & 6 & 19 & 6 & 45 \end{array}\right)$$

$$\operatorname{rank}(\mathcal{X}) = \min\{\, r \mid \mathcal{X} = \mathbf{a}_1^{\otimes d} + \cdots + \mathbf{a}_r^{\otimes d} \,\}$$

Rank decomposition

$$\mathcal{X} = 2 \cdot \begin{bmatrix} 1 \\ -2 \\ -1 \end{bmatrix}^{\otimes 3} + 5 \cdot \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix}^{\otimes 3} - \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}^{\otimes 3}$$

- Symmetric tensor rank
  - For any given tensor, NP-hard to compute its rank (Hillar & Lim, 2013)
  - *Typical rank* known over $\mathbb{C}$, per Alexander-Horshowitz (Comon, Golub, Lim, Mourraine, 2008)
  - In practice, trial and error!
- Symmetric tensor decomposition
  - Waring decomposition, e.g., Oeding & Ottaviani (2013), Landsberg (2012)
  - Algebraic methods based on Grobner bases or numerical root-finding method (Nie, 2014)
  - Direct optimization formulation (Kolda, 2015)

Direct Optimization for Symmetric CP (d=3)

Formulation:

$$\min_{\boldsymbol{\lambda},\mathbf{A}} \frac{1}{2}\|\mathcal{X} - \hat{\mathcal{X}}\|^2 \text{ where } \hat{\mathcal{X}} = \sum_{j=1}^{r} \lambda_j \, \mathbf{a}_j^{\otimes d}$$

Gradients:

$$\frac{\partial f}{\partial \mathbf{a}_j} = -d\lambda_j \underbrace{\mathcal{X}\mathbf{a}_j^{d-1}}_{\text{TTSV}} + d\lambda_j \sum_{k=1}^{r} \lambda_k \langle \mathbf{a}_j, \mathbf{a}_k \rangle^{d-1} \mathbf{a}_k$$

$$\frac{\partial f}{\partial \lambda_j} = -\underbrace{\mathcal{X}\mathbf{a}_j^{d}}_{\text{TTSV}} + \sum_{k=1}^{r} \lambda_k \langle \mathbf{a}_j, \mathbf{a}_k \rangle^{d}$$

Use gradients with standard first-order numerical optimization method such as steepest descent or limited-memory BFGS.

Norm

$$\|\mathcal{X}\|^2 = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} x_{ijk}^2$$

Vector Dot Product

$$\langle \mathbf{a}, \mathbf{b} \rangle = \sum_{i=1}^{n} a_i b_i$$

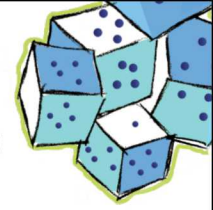
TTSV: Tensor Times Same Vector ($d-1$) times

$$(\mathcal{X}\mathbf{a}^{d-1})_i = \sum_{j=1}^{n}\sum_{k=1}^{n} x_{ijk} a_j a_k$$

TTSV: Tensor Times Same Vector d times

$$\mathcal{X}\mathbf{a}^{d} = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} x_{ijk} a_i a_j a_k$$

Solving Small Problems is Easy with Tensor Toolbox for MATLAB



```
tic
optopt.DisplayIters = 10;
M = cp_sym(X,3,'alg_options',optopt)
```

```
 Iter  FuncEvals       F(X)          ||G(X)||/N
------ --------- ---------------- ----------------
     0         1    3787.04851027      27.69717065
    10        39    1172.79483308      15.12379151
    20        74     129.01389491      31.29890861
    30       101      10.09840492      15.75940546
    40       122       0.91539600       4.77960918
    50       144       0.00464623       0.19586470
    60       164       0.00011886       0.02724764
    70       186       0.00000155       0.00441564
    80       207       0.00000015       0.00130310
    90       229       0.00000000       0.00001130
    91       232       0.00000000       0.00000669
M is a symktensor of order 3 and dimension 3
        M.lambda = [ 0.29082      1.3271      0.5169 ]
        M.U =
                     2.1769    1.5561   -1.2460
                    -4.3537    1.5561   -2.4921
                    -2.1769    3.1121    2.4921
```

```
full(M)
```

```
ans is a symmetric tensor with 3 modes of dimension 3
        (1,1,1)    7.0000
        (1,1,2)   -3.0000
        (1,1,3)    9.0000
        (1,2,2)   13.0000
        (1,2,3)   20.0000
        (1,3,3)   19.0000
        (2,2,2)  -27.0000
        (2,2,3)    6.0000
        (2,3,3)    6.0000
        (3,3,3)   45.0000
```

```
toc
```

```
Elapsed time is 0.211744 seconds.
```

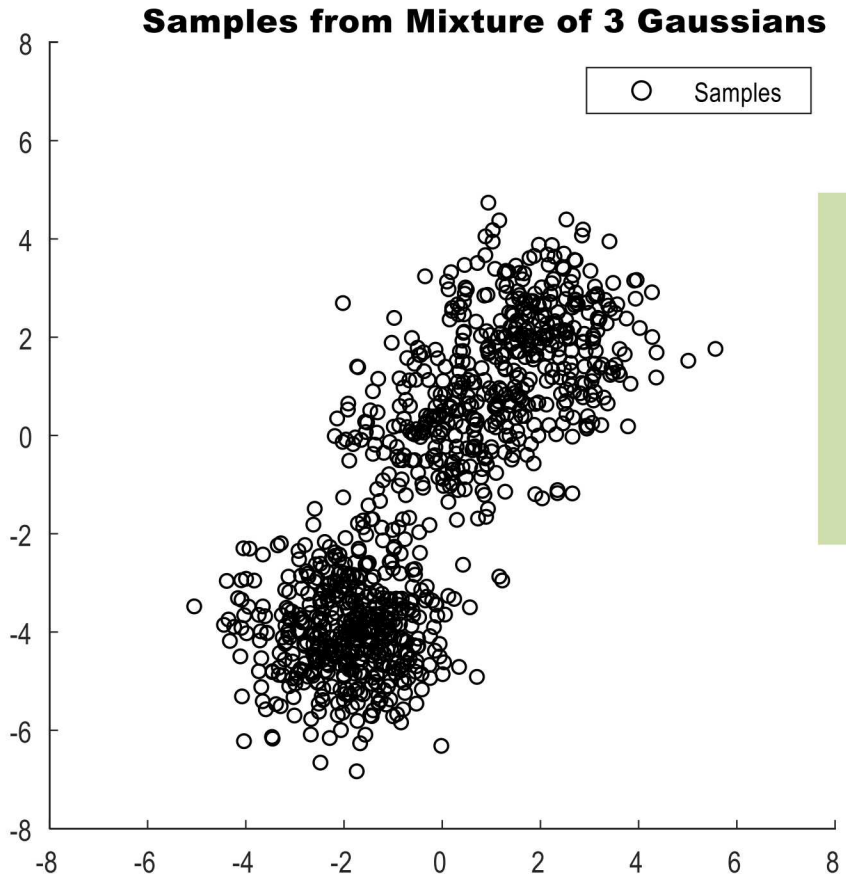
True solution from Nie 2014

```
Mtrue = symktensor([3 5 -1]', [1 1 1; -2 1 2; -1 2 -2],3)
```

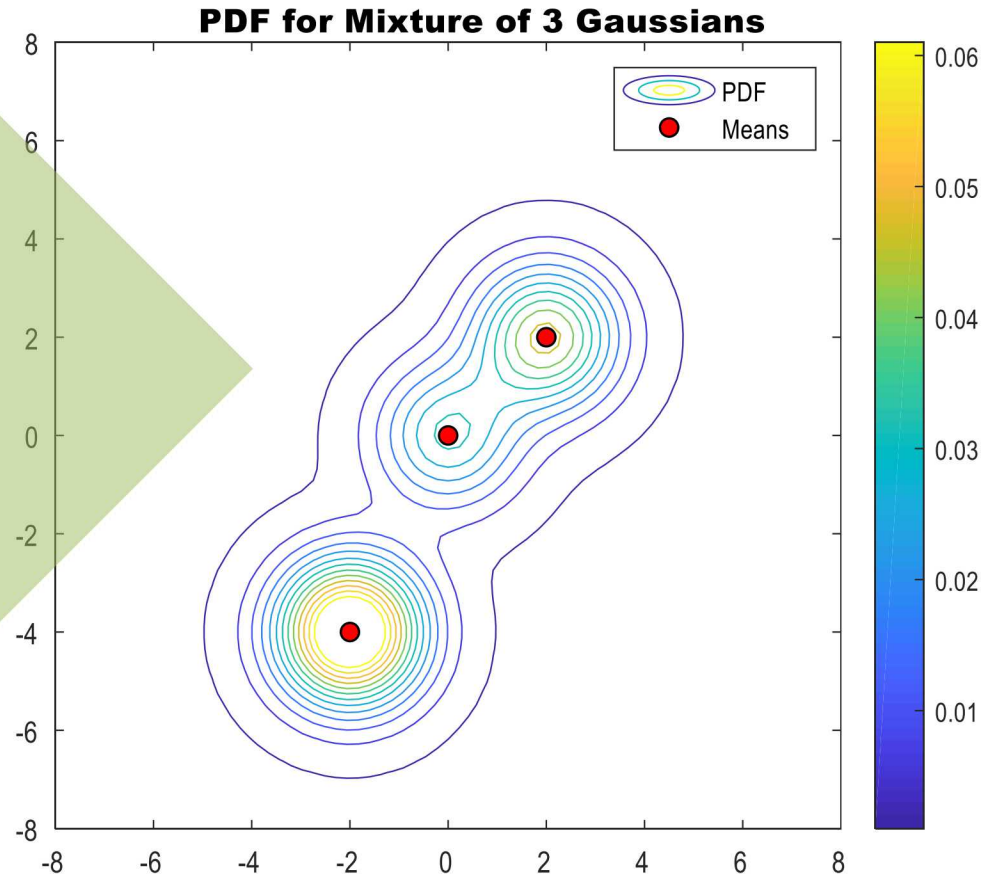
```
Mtrue is a symktensor of order 3 and dimension 3
        Mtrue.lambda = [ 3  5 -1 ]
        Mtrue.U =
                     1     1     1
                    -2     1     2
                    -1     2    -2
```

Gaussian Mixture Model – 2D

For ease of illustration, we focus on $n = 2$ dimensions.
Generally interested in much higher dimensions!

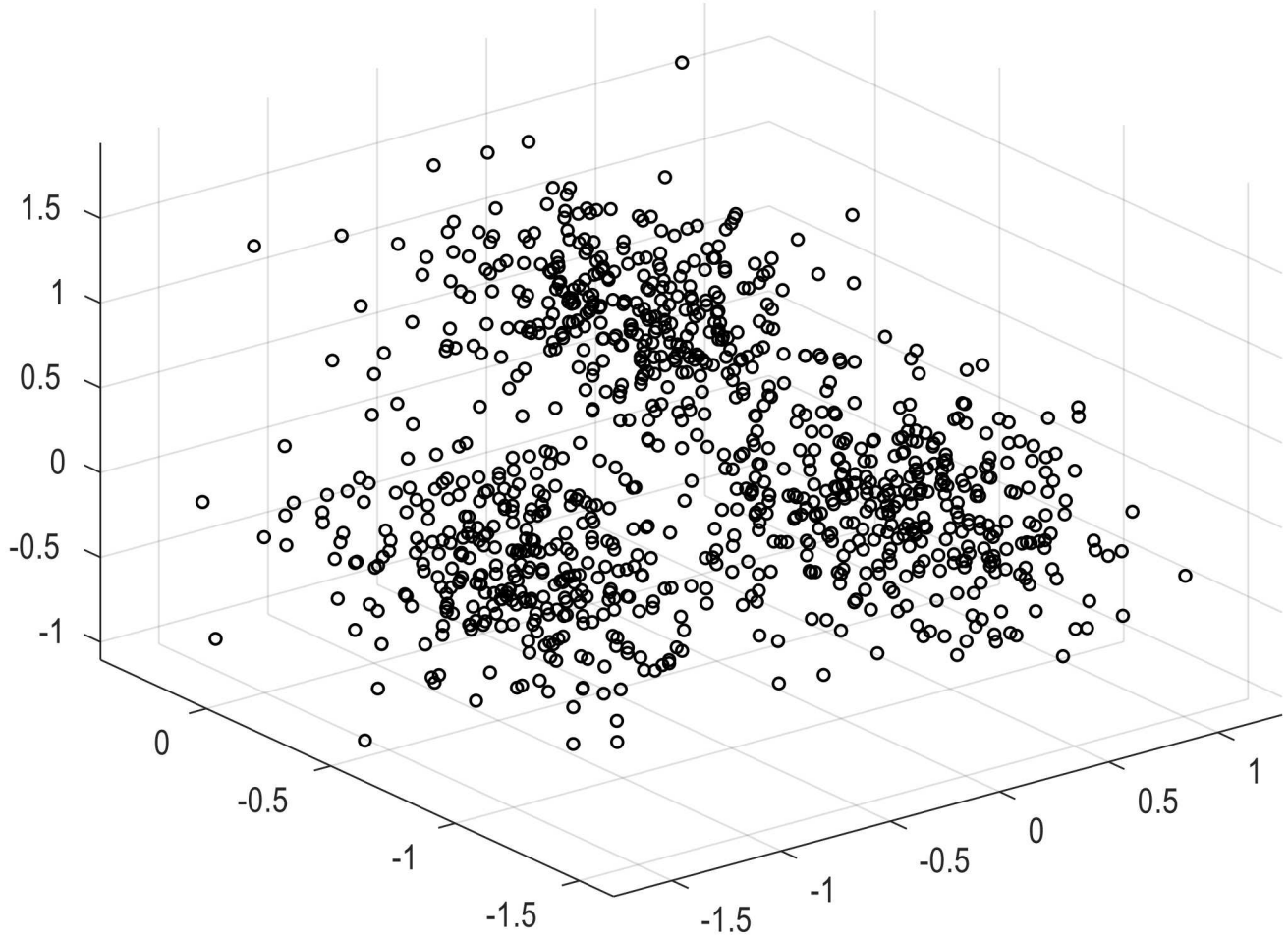


Given just the samples (point cloud), can we recover the means?



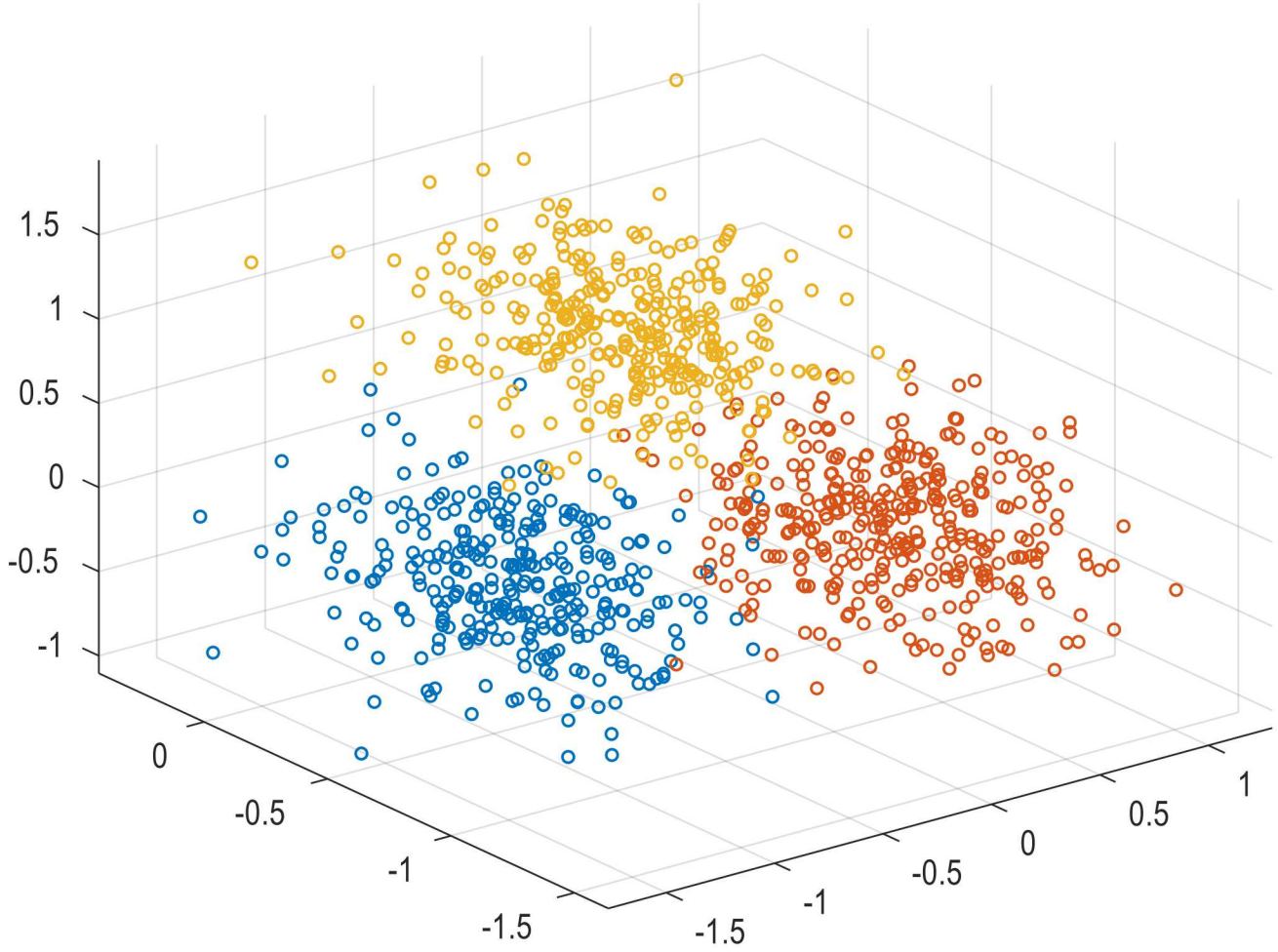
Gaussian Mixture Model – 3D

What if we observe a collection of points from an unknown mixture of Gaussians?



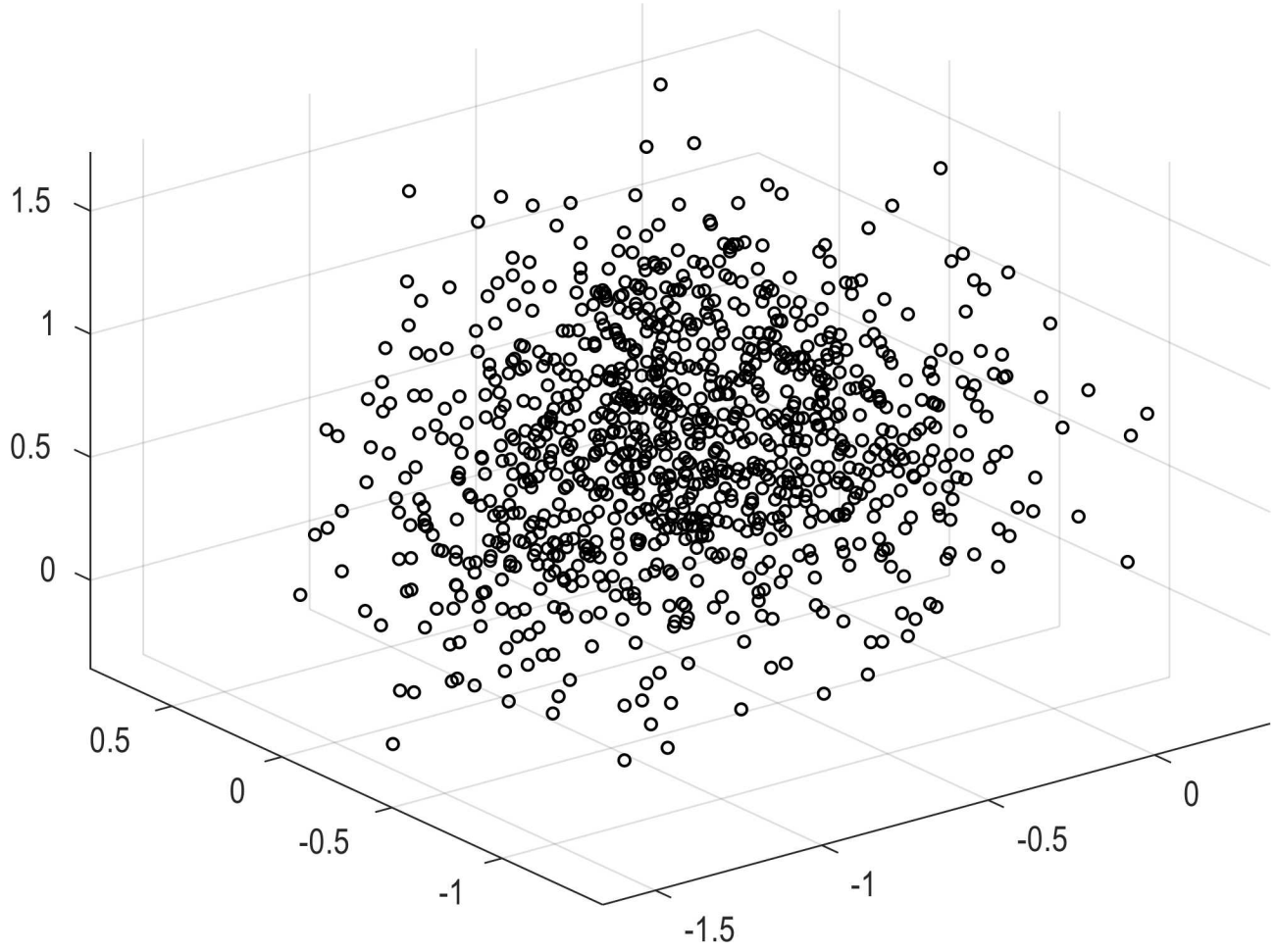
Recovery is Each if Means Separated

Points colored by which Gaussian they belong to – pretty easy to separate



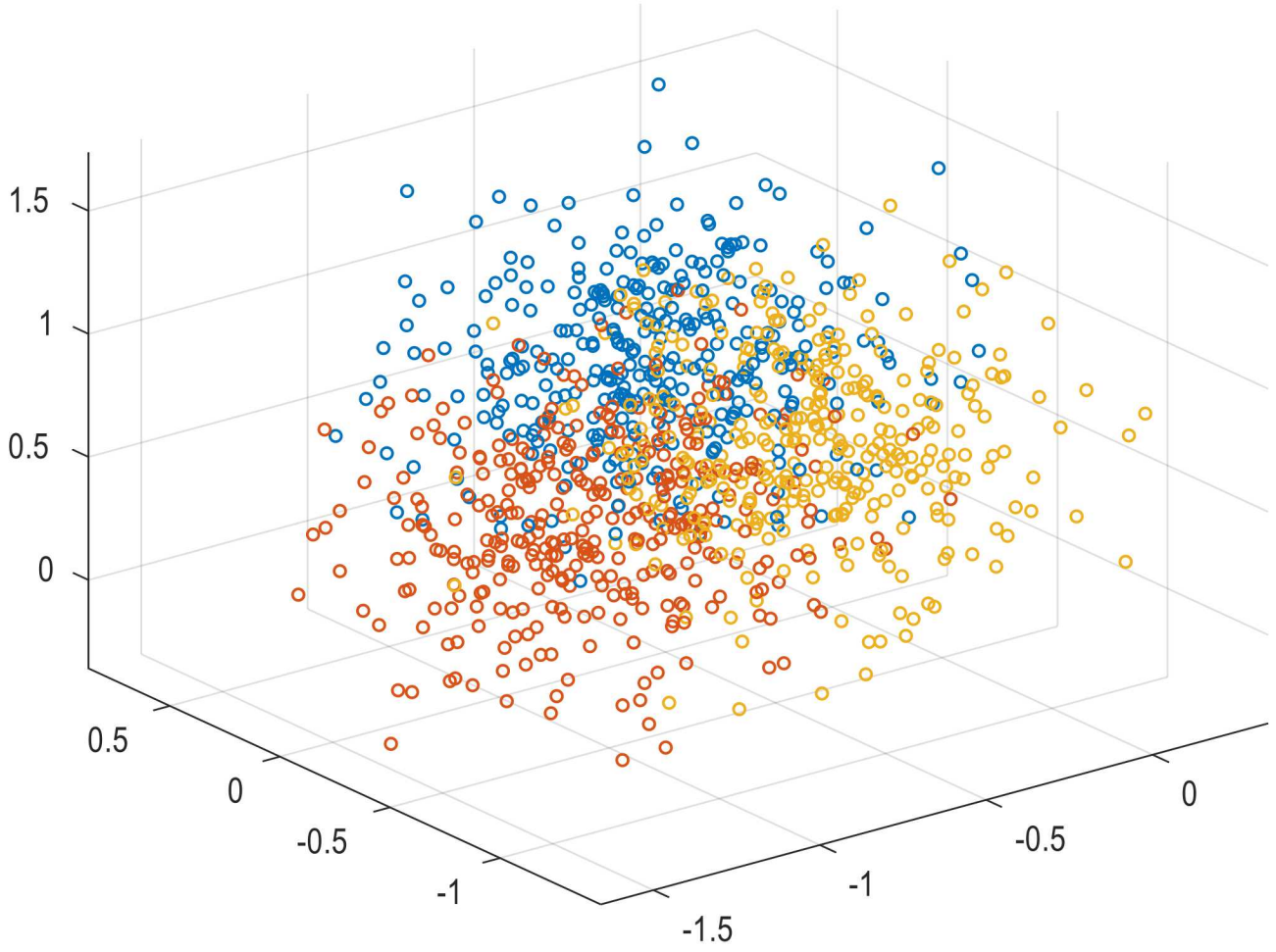
Gaussian Mixture Model – 3D

What if we observe a collection of points from an unknown mixture of Gaussians?



Recovery More Difficult if Means Nearby

Points colored by which Gaussian they belong to – more difficult to separate

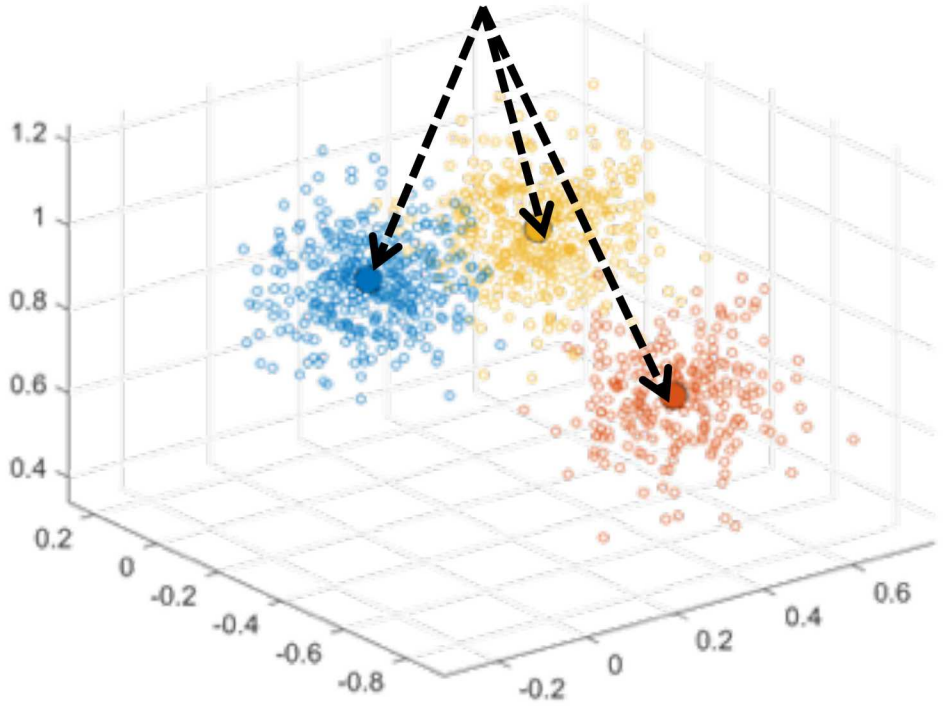


Machine Learning Motivation: Observations from Unknown Mixture of Gaussians

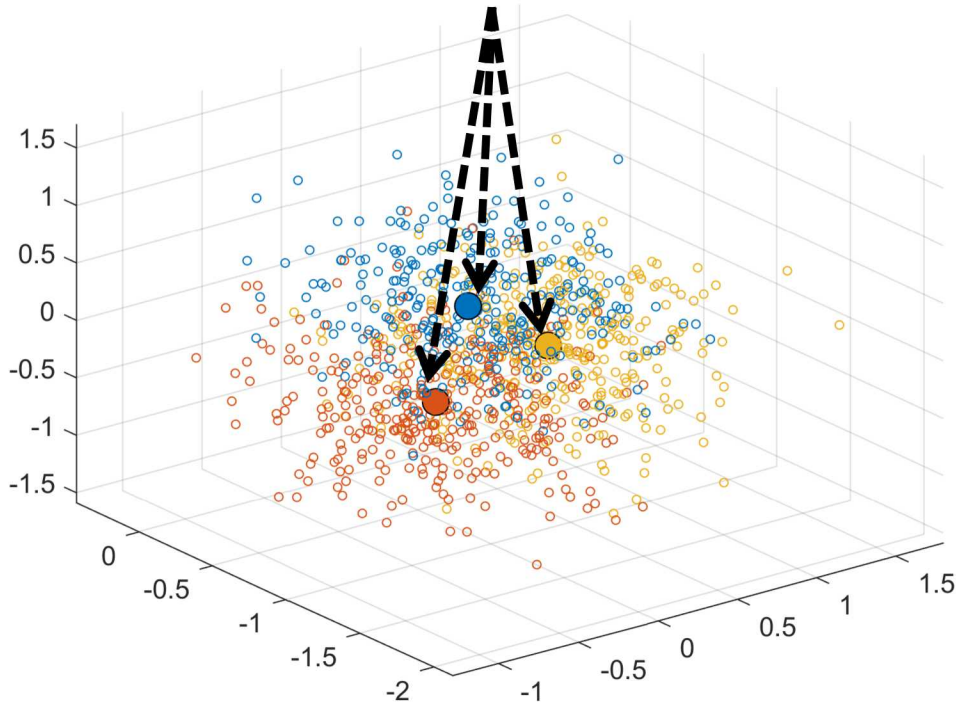
We observe p random vectors of length n coming from a mixture of r Gaussian distributions.

Can we recover the means of the Gaussians?

Easy: Means Well Separated



Hard: Means Close Together



For these pictures: p = 1000, n = 3, r = 3. Means shown as filled in larger circles. Samples as open circles.

We care about larger values of all of these parameters!

Empirical Higher-order Moments

Let $\mathbf{z} \in \mathbb{R}$ for $i = 1, \ldots, p$ be the observations of the random variable Z

$$\mathbf{Z} = \begin{bmatrix} \mathbf{z}_1 & \mathbf{z}_2 & \cdots & \mathbf{z}_p \end{bmatrix} \in \mathbb{R}^{n \times p} \quad \text{(observation matrix)}$$

Z = random variable
$\mathbb{E}(Z)$ = 1st order moment
$\mathbb{E}(Z \circ Z)$ = 2nd order moment
$\mathbb{E}(Z \circ Z \circ Z)$ = 3rd order moment

1st order empirical moment: $\frac{1}{p} \sum_{\ell=1}^{p} \mathbf{z}_\ell \in \mathbb{R}^n$ (mean)

2nd order empirical moment: $\frac{1}{p} \sum_{\ell=1}^{p} \mathbf{z}_\ell \otimes \mathbf{z}_\ell \in \mathbb{R}^{n \times n}$ (covariance)

Interesting Fact: If Z is Gaussian with mean zero, then its 3rd order moment is zero!

3rd order empirical moment: $\mathcal{X} = \frac{1}{p} \sum_{\ell=1}^{p} \mathbf{z}_\ell \otimes \mathbf{z}_\ell \otimes \mathbf{z}_\ell \in \mathbb{R}^{n \times n \times n} \quad \Leftrightarrow \quad x_{ijk} = \frac{1}{p} \sum_{\ell=1}^{p} z_{i\ell} z_{j\ell} z_{k\ell}$

A rank-r symmetric tensor decomposition of the 3rd-order moment can recover the means!

Symmetric Tensor Decomposition of 3rd-order moment recovers the Means

Each observation is of the form:

$$\text{Choose random } j \in \{1, \ldots, r\} \text{ and then } \mathbf{z}_\ell \sim \mathcal{N}(\mathbf{a}_j, \sigma \mathbf{I})$$

$$\mathbf{Z} = \begin{bmatrix} \mathbf{z}_1 & \mathbf{z}_2 & \cdots & \mathbf{z}_p \end{bmatrix} \in \mathbb{R}^{n \times p} \quad \text{(observation matrix)}$$

If we have only $\mathbf{Z}$, can we recover $\mathbf{A}$?

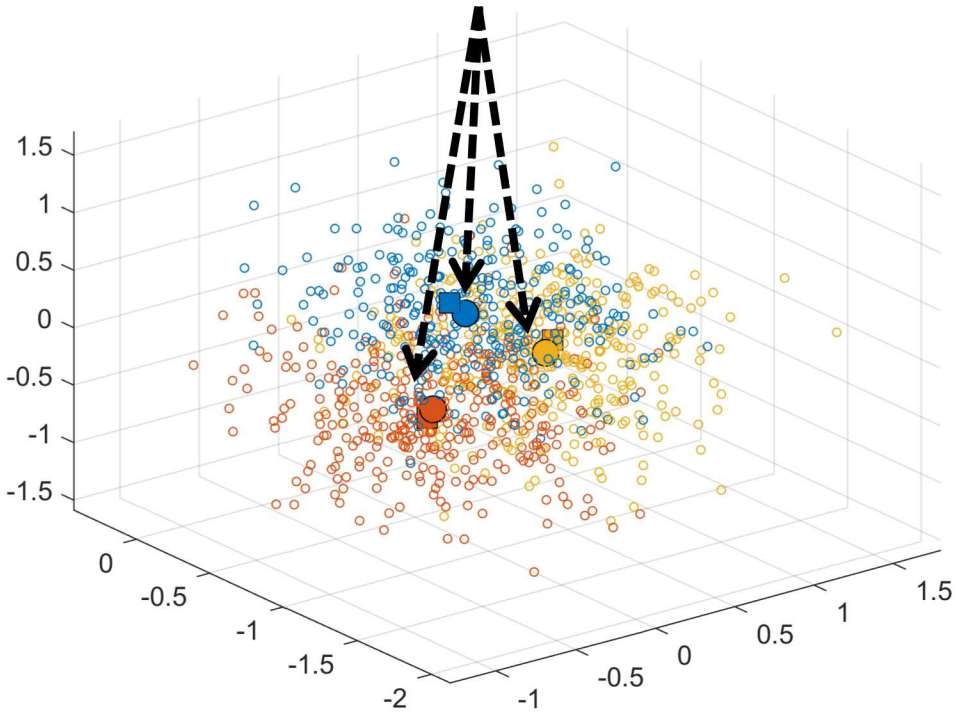
$$\mathcal{X} = \frac{1}{p} \sum_{\ell=1}^{p} \mathbf{z}_\ell^{\otimes 3} \in \mathbb{R}^{n \times n \times n}$$

Compute symmetric rank r = 3 (# of Gaussians) approximation to 3rd-order moment:

$$\hat{\mathcal{X}} = \sum_{j=1}^{r} \lambda_j \mathbf{a}_j^{\otimes 3} \in \mathbb{R}^{n \times n \times n}$$

More generally, any density can be approximated by a mixture of Gaussians

big circles show true means, squares show recoveries from noisy data



p = 1000, n = 3, r = 3
Samples as open circles

Large Dimension (n) or Number of Observations (p) Make Problem Intractable

Observations: $\mathbf{Z} = \begin{bmatrix} \mathbf{z}_1 & \mathbf{z}_2 & \cdots & \mathbf{z}_p \end{bmatrix} \in \mathbb{R}^{n \times p}$

d^{th} order empirical moment tensor: $\mathcal{X} = \frac{1}{p} \sum_{\ell=1}^{p} \mathbf{z}_\ell^{\otimes d} \in \mathbb{R}^{n \times n \times n}$

Optimization function and gradients:

$$\min_{\boldsymbol{\lambda}, \mathbf{A}} F(\mathcal{X}, \hat{\mathcal{X}}) \equiv \frac{1}{2} \|\mathcal{X} - \hat{\mathcal{X}}\|^2 \text{ where } \hat{\mathcal{X}} = \sum_{j=1}^{r} \lambda_j \, \mathbf{a}_j^{\otimes d}$$

$$\frac{\partial F}{\partial \mathbf{a}_j} = -d\lambda_j \mathcal{X} \mathbf{a}_j^{d-1} + d\lambda_j \sum_{k=1}^{r} \lambda_k \langle \mathbf{a}_j, \mathbf{a}_k \rangle^{d-1} \mathbf{a}_k$$

$$\frac{\partial F}{\partial \lambda_j} = -\mathcal{X} \mathbf{a}_j^d + \sum_{k=1}^{r} \lambda_k \langle \mathbf{a}_j, \mathbf{a}_k \rangle^d$$

Cost to form moment tensor: $O(pn^d)$

Storage for moment tensor: $O(n^d)$

Example: n = 128, d = 4 ⇒ storage = 2 GB

Example: n = 512, d = 3 ⇒ storage = 1 GB

Every optimization step costs: $O(rn^d)$

Key: Avoid Forming Explicit Moment Tensor by Exploiting Structure!

Sandia National Laboratories

$$\mathbf{Z} \in \mathbb{R}^{n \times p}, \mathbf{A} \in \mathbb{R}^{n \times r}, r \ll p$$

$$\mathcal{X} = \frac{1}{p} \sum_{\ell=1}^{p} \mathbf{z}_\ell^{\otimes d}, \quad \hat{\mathcal{X}} = \sum_{j=1}^{r} \lambda_j \mathbf{a}_j^{\otimes d}$$

Work reduction: $O(pn^d) \rightarrow O(npr)$

Explicit

$$F(\mathcal{X}, \hat{\mathcal{X}}) = \tfrac{1}{2}(\|\mathcal{X}\|_F^2 + \|\hat{\mathcal{X}}\|_F^2) - \langle \mathcal{X}, \hat{\mathcal{X}} \rangle$$

$$\frac{\partial F}{\partial \mathbf{a}_j} = d\lambda_j \left(-\mathcal{X} \mathbf{a}_j^{d-1} + \sum_{k=1}^{r} \lambda_k \langle \mathbf{a}_j, \mathbf{a}_k \rangle^{d-1} \mathbf{a}_k \right)$$

$$\frac{\partial F}{\partial \lambda_j} = -\mathcal{X} \mathbf{a}_j^{d} + \sum_{k=1}^{r} \lambda_k \langle \mathbf{a}_j, \mathbf{a}_k \rangle^{d}$$

Implicit

$$F(\mathcal{X}, \hat{\mathcal{X}}) = \tfrac{1}{2}(\mathbf{e}^T (\mathbf{Z}^T \mathbf{Z})^d \mathbf{e} + \boldsymbol{\lambda}^T (\mathbf{A}^T \mathbf{A})^d \boldsymbol{\lambda}) - \mathbf{e}^T (\mathbf{Z}^T \mathbf{A})^d \boldsymbol{\lambda}$$

$$\frac{\partial F}{\partial \mathbf{a}_j} = d\lambda_j \left(-\mathbf{Z} \cdot (\mathbf{Z}^T \mathbf{a}_j)^{d-1} + \sum_{k=1}^{r} \lambda_k \langle \mathbf{a}_j, \mathbf{a}_k \rangle^{d-1} \mathbf{a}_k \right)$$

$$\frac{\partial F}{\partial \lambda_j} = -\langle (\mathbf{Z}^T \mathbf{a}_j)^{d-1}, \mathbf{Z}^T \mathbf{a}_j \rangle + \sum_{k=1}^{r} \lambda_k \langle \mathbf{a}_j, \mathbf{a}_k \rangle^{d}$$

Key: Avoid Forming Explicit Moment Tensor by Exploiting Structure!

Sandia National Laboratories

$$\mathbf{Z} \in \mathbb{R}^{n \times p}, \mathbf{A} \in \mathbb{R}^{n \times r}, r \ll p$$

$$\mathcal{X} = \frac{1}{p} \sum_{\ell=1}^{p} \mathbf{z}_\ell^{\otimes d}, \quad \hat{\mathcal{X}} = \sum_{j=1}^{r} \lambda_j \mathbf{a}_j^{\otimes d}$$

Work reduction: $O(pn^d) \rightarrow O(npr)$

Explicit

$O(pn^d)$

$$F(\mathcal{X}, \hat{\mathcal{X}}) = \tfrac{1}{2}(\|\mathcal{X}\|_F^2 + \|\hat{\mathcal{X}}\|_F^2) - \langle \mathcal{X}, \hat{\mathcal{X}} \rangle$$

$O(rn^d)$ $O(n^d)$

$$\frac{\partial F}{\partial \mathbf{a}_j} = d\lambda_j \left(-\mathcal{X} \mathbf{a}_j^{d-1} + \sum_{k=1}^{r} \lambda_k \langle \mathbf{a}_j, \mathbf{a}_k \rangle^{d-1} \mathbf{a}_k \right)$$

$O(n^d)$ $O(nr)$

$$\frac{\partial F}{\partial \lambda_j} = -\mathcal{X} \mathbf{a}_j^d + \sum_{k=1}^{r} \lambda_k \langle \mathbf{a}_j, \mathbf{a}_k \rangle^d$$

$O(n^d)$ $O(nr)$

Implicit

$O(np^2)$

$$F(\mathcal{X}, \hat{\mathcal{X}}) = \tfrac{1}{2}(\mathbf{e}^T (\mathbf{Z}^T \mathbf{Z})^d \mathbf{e} + \boldsymbol{\lambda}^T (\mathbf{A}^T \mathbf{A})^d \boldsymbol{\lambda}) - \mathbf{e}^T (\mathbf{Z}^T \mathbf{A})^d \boldsymbol{\lambda}$$

$O(nr^2)$ $O(pnr)$

$$\frac{\partial F}{\partial \mathbf{a}_j} = d\lambda_j \left(-\mathbf{Z} \cdot (\mathbf{Z}^T \mathbf{a}_j)^{d-1} + \sum_{k=1}^{r} \lambda_k \langle \mathbf{a}_j, \mathbf{a}_k \rangle^{d-1} \mathbf{a}_k \right)$$

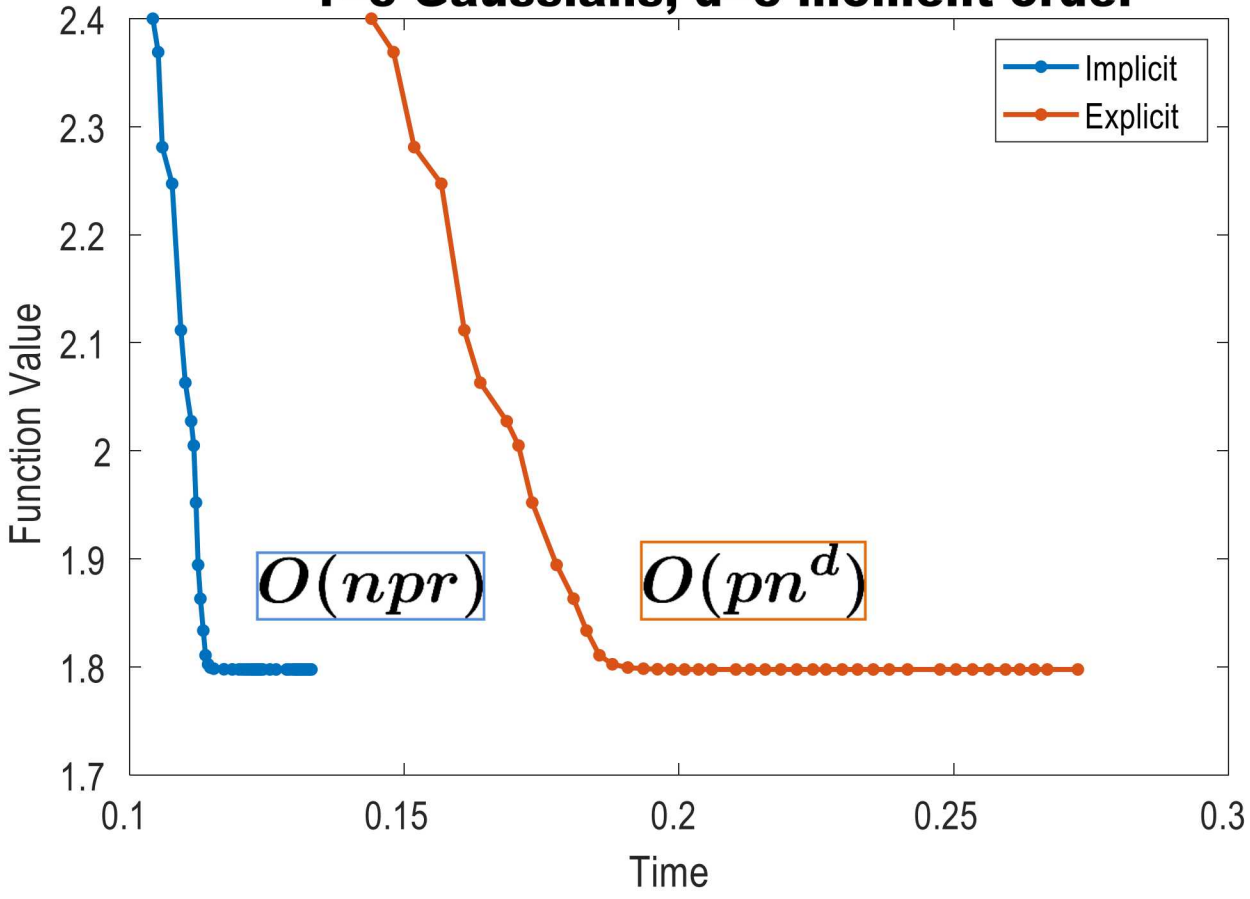
$O(np)$ $O(nr)$

$$\frac{\partial F}{\partial \lambda_j} = -\langle (\mathbf{Z}^T \mathbf{a}_j)^{d-1}, \mathbf{Z}^T \mathbf{a}_j \rangle + \sum_{k=1}^{r} \lambda_k \langle \mathbf{a}_j, \mathbf{a}_k \rangle^d$$

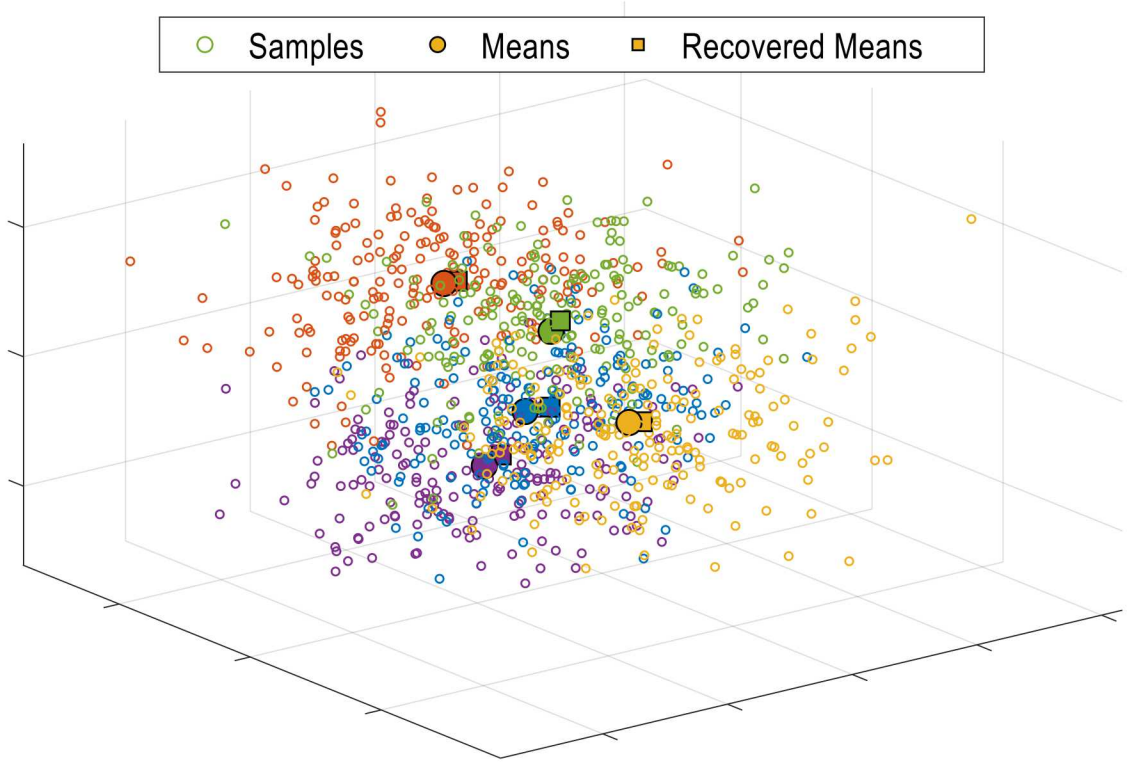
$O(np)$ $O(nr)$

Implicit Method is Much Faster Than Explicit

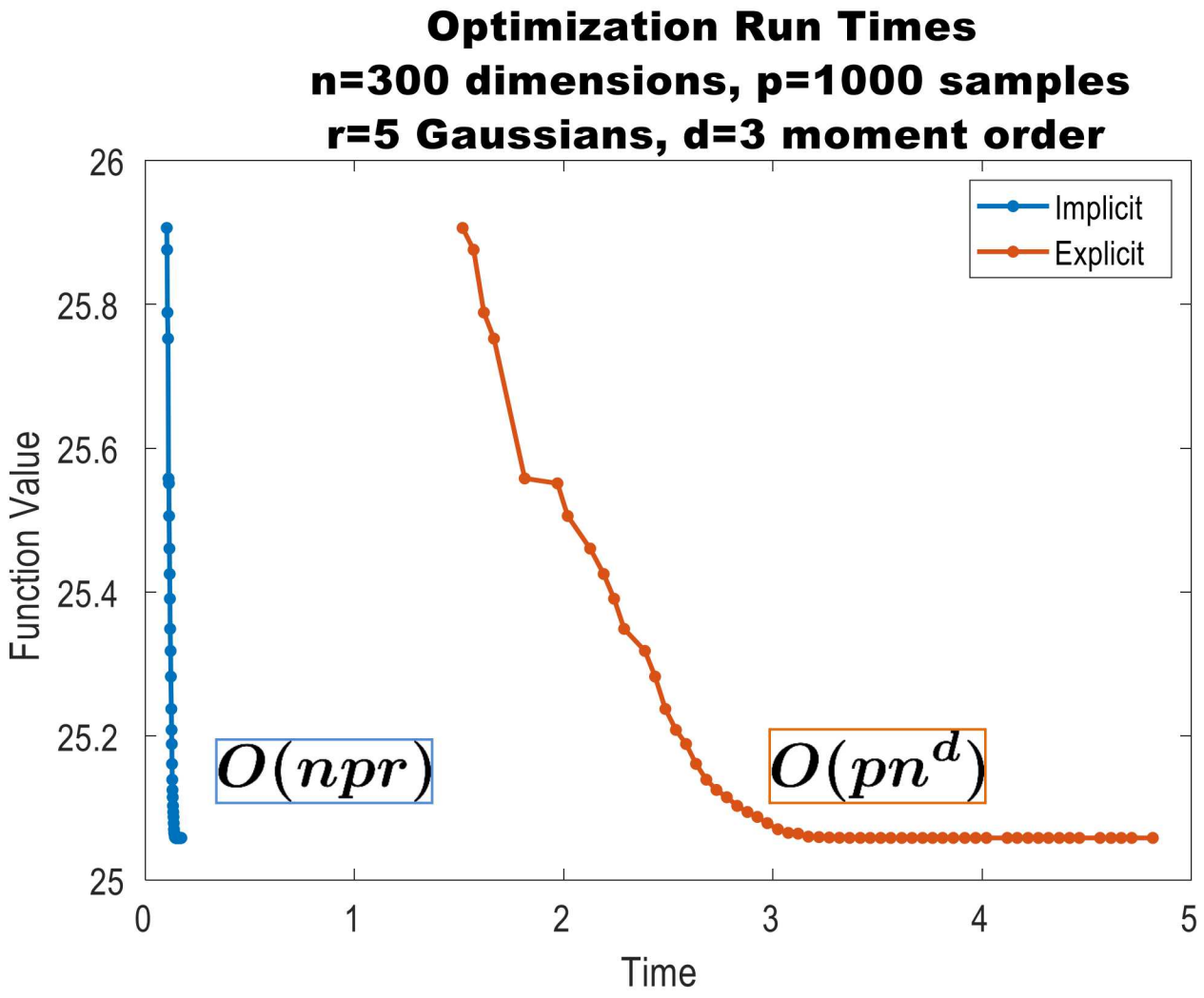
Optimization Run Times
n=100 dimensions, p=1000 samples
r=5 Gaussians, d=3 moment order

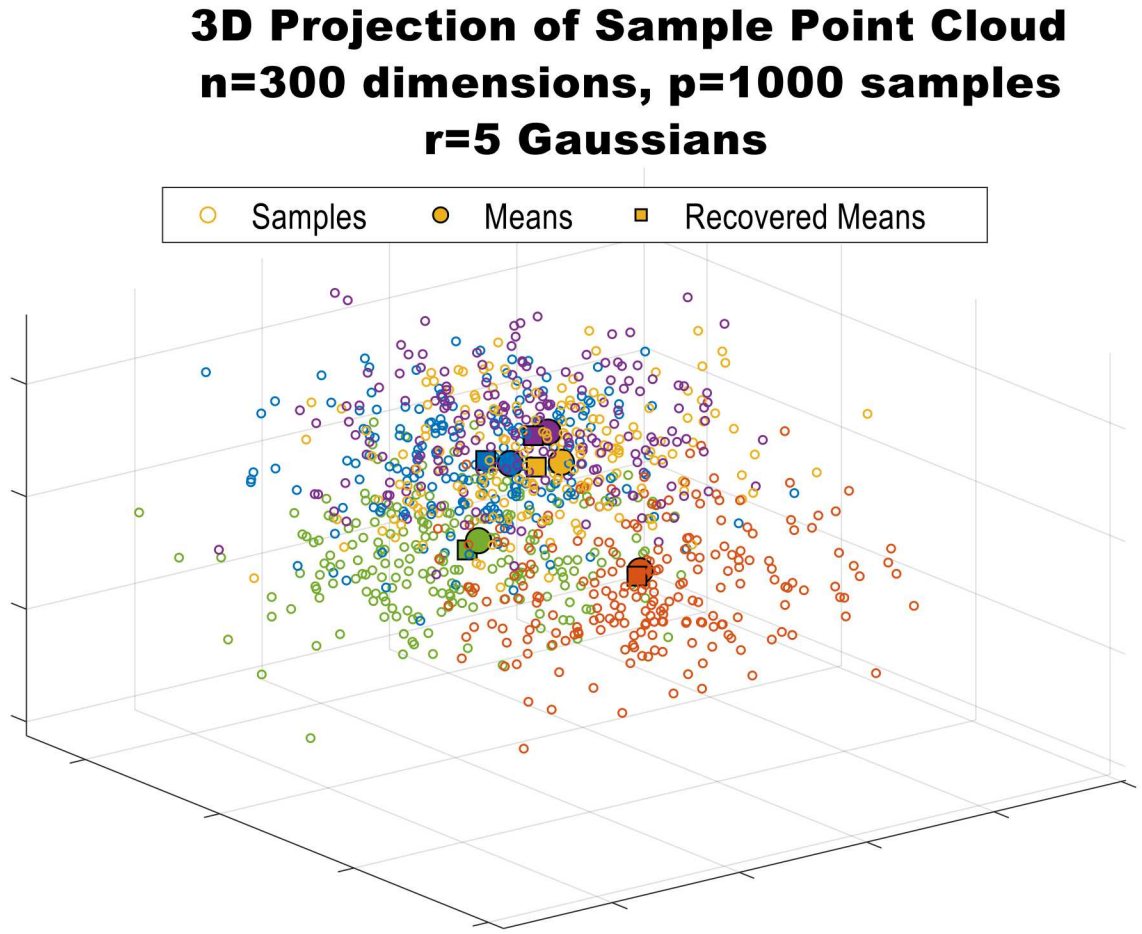


3D Projection of Sample Point Cloud
n=100 dimensions, p=1000 samples
r=5 Gaussians

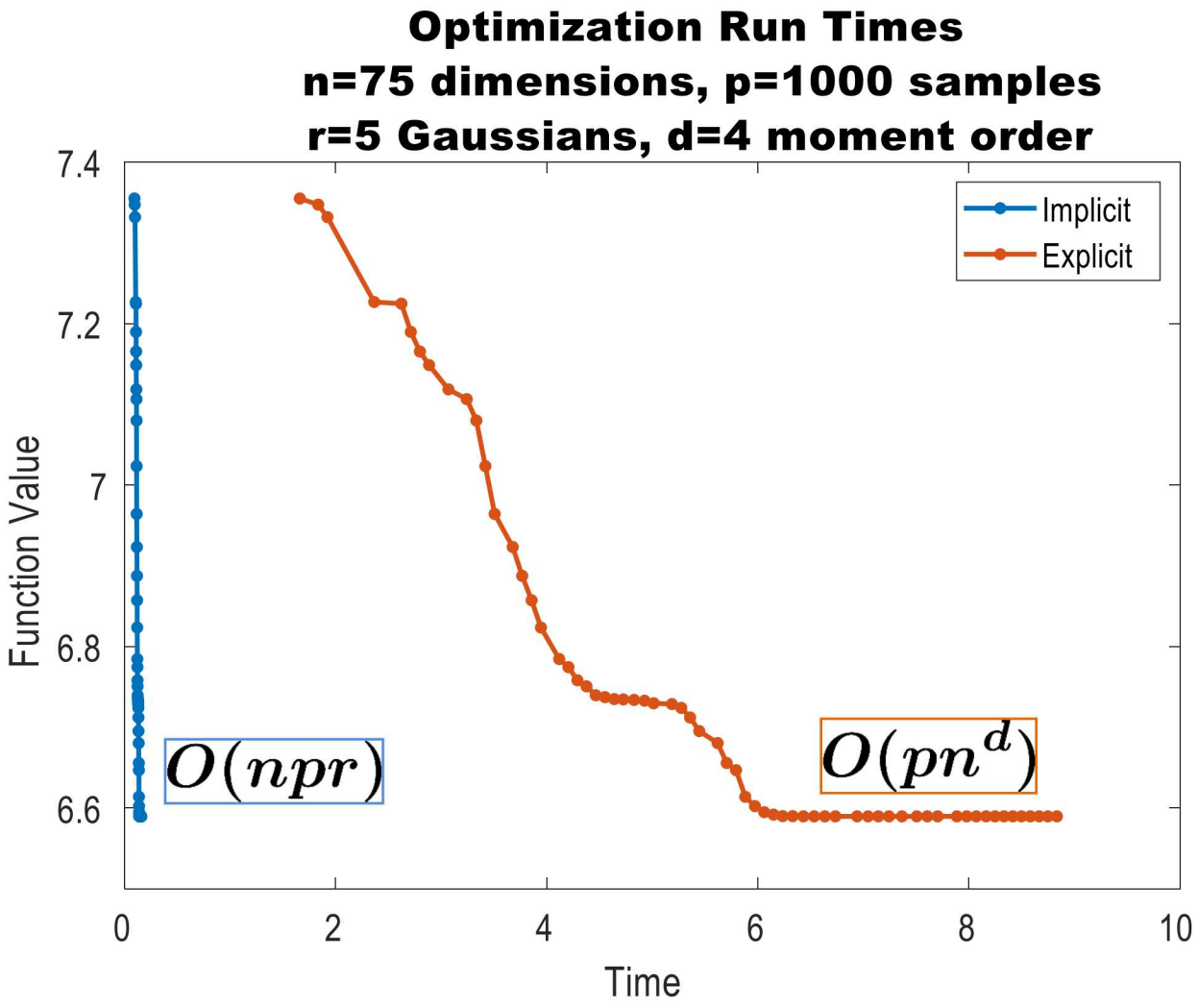


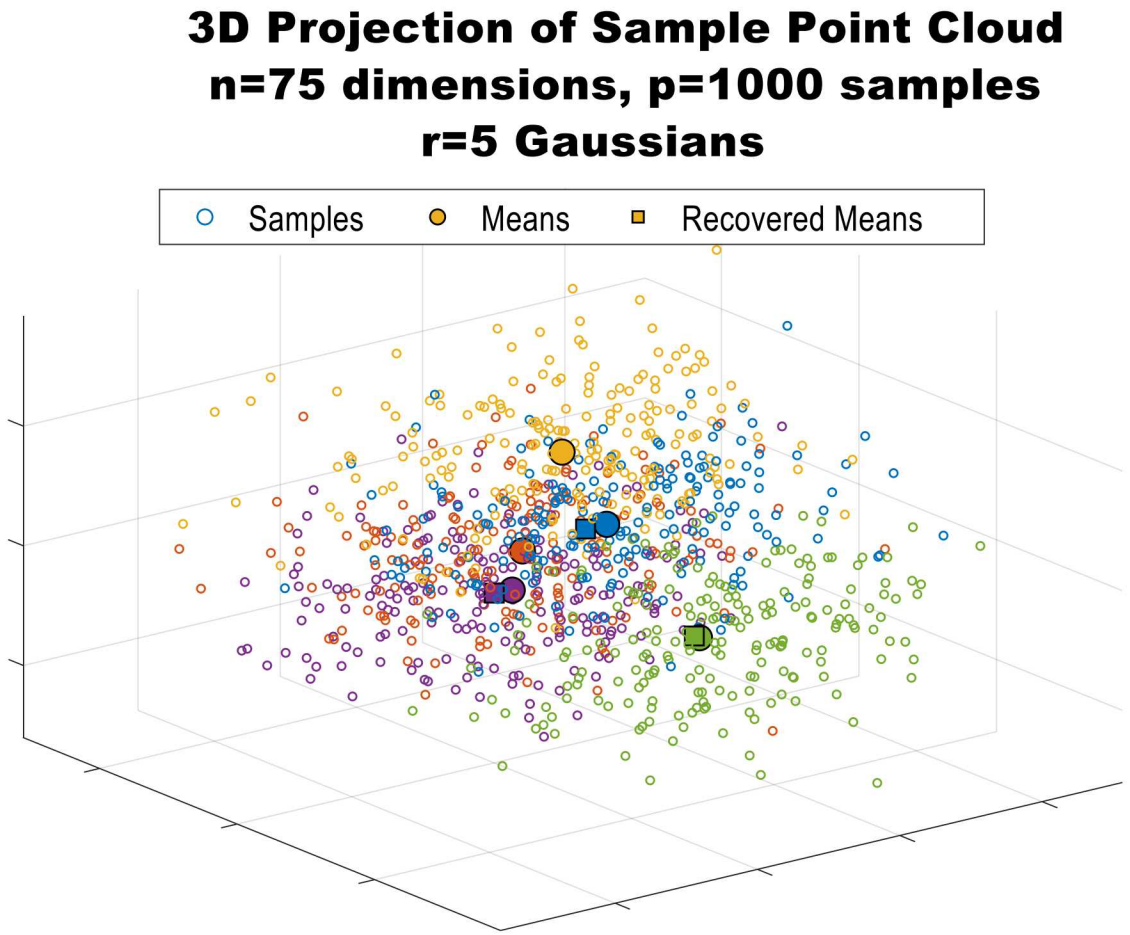
Advantage Becomes More Pronounced as Dimension (n) Grows





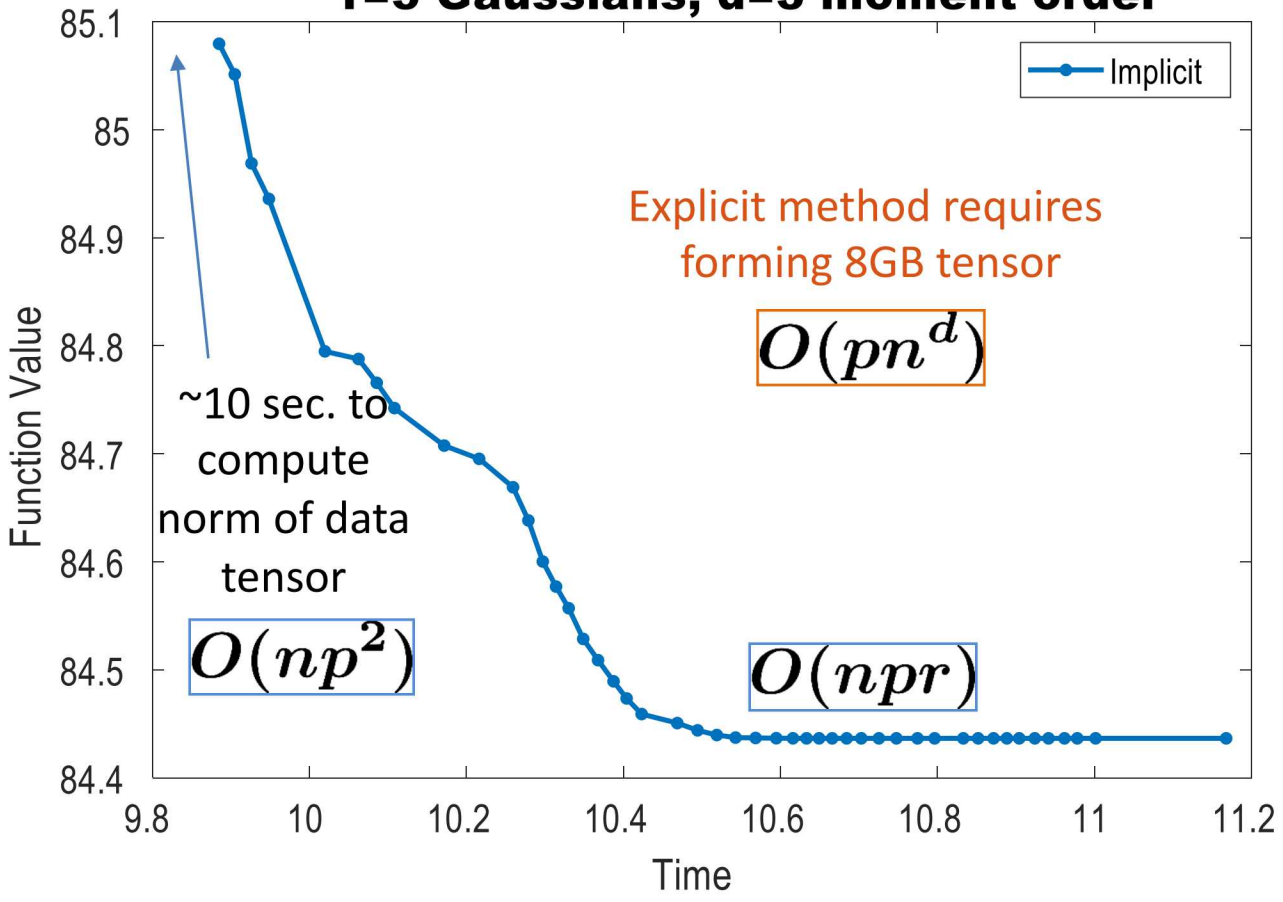
Speed-ups from Implicit Method increase as Order (d) grows



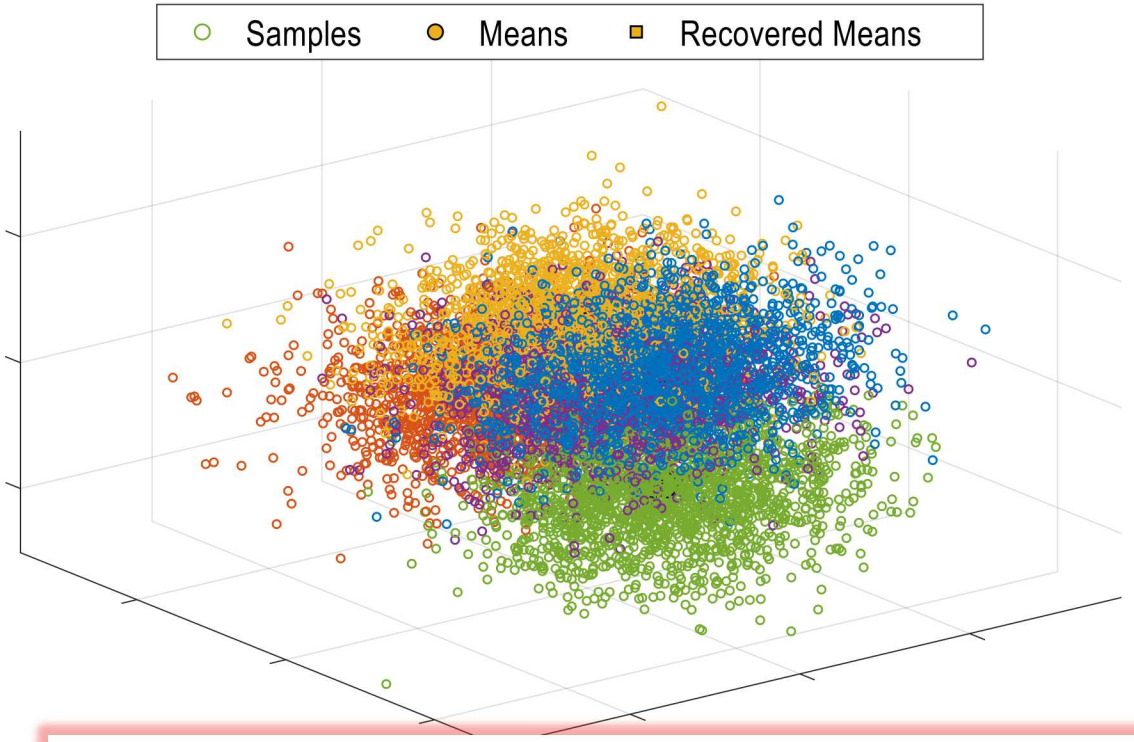


Implicit Symmetric CP Scales to Much Larger Dimension (e.g., n=1000) than Explicit

Optimization Run Times
n=1000 dimensions, p=10000 samples
r=5 Gaussians, d=3 moment order



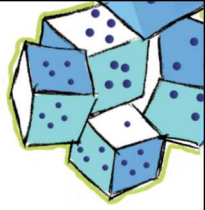
3D Projection of Sample Point Cloud
n=1000 dimensions, p=10000 samples
r=5 Gaussians



Issue: Iteration cost scales linearly with # samples (p), which could be millions or billions

Improve Scalability with Stochastic Gradient Descent for Implicit Symmetric CP

Sandia National Laboratories



30-Second Tutorial on SGD

$$\min F(x)$$

Gradient Descent (GD)

α = learning rate

$$x^{(t+1)} = x^{(t)} - \alpha g^{(t)}$$

Stochastic Gradient Descent (SGD)

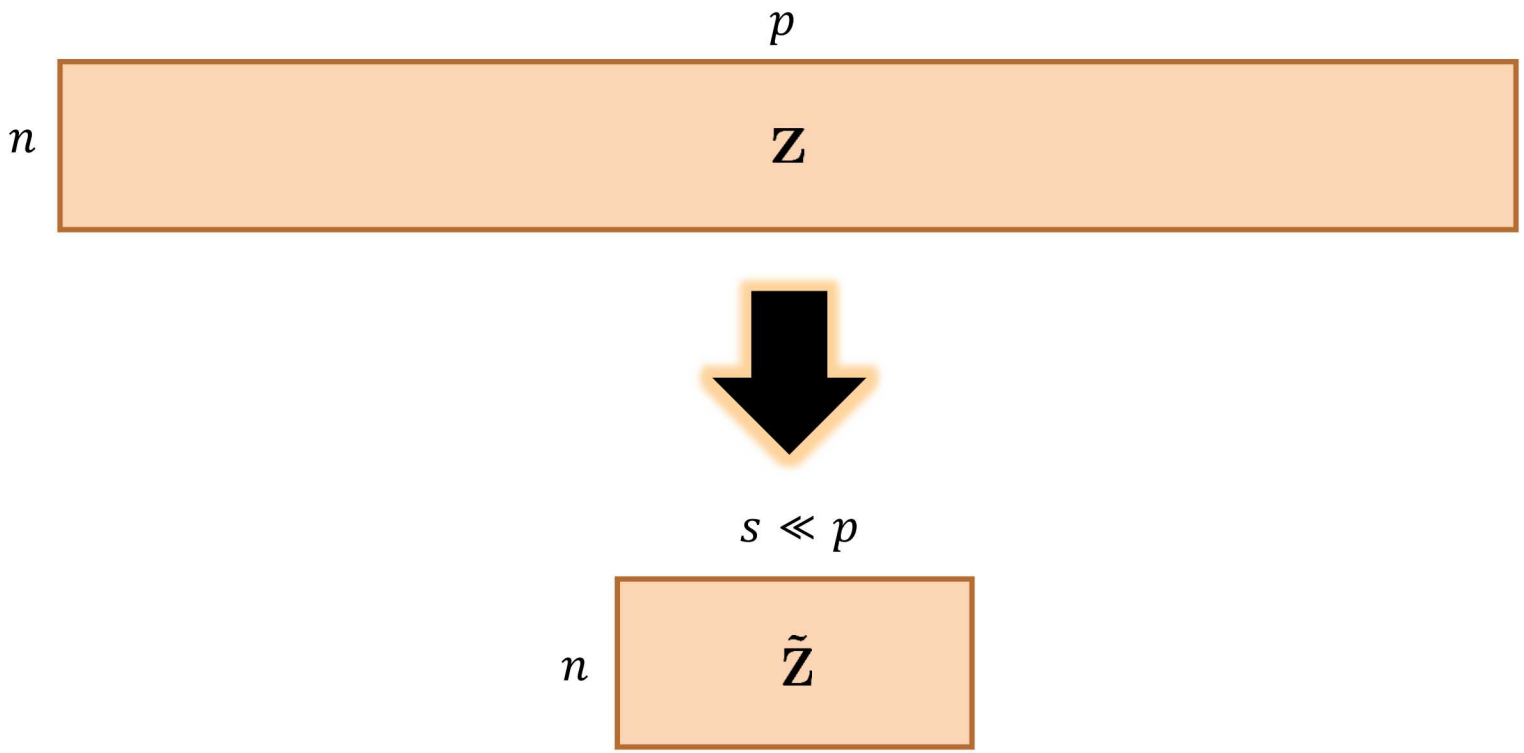
$$x^{(t+1)} = x^{(t)} - \alpha \tilde{g}^{(t)}$$

$$\mathbb{E}[\tilde{g}^{(t)}] = g^{(t)} \equiv \nabla F(x^{(t)})$$

Adam (Kingma & Ba, 2015)

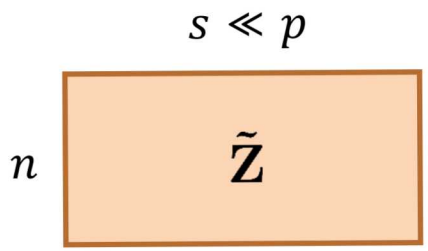
Adaptive momentum SGD

Idea: For stochastic gradient, use just a few random columns of Z at each iteration…



Stochastic Function & Gradient

Work reduction: $O(npr) \rightarrow O(nsr)$

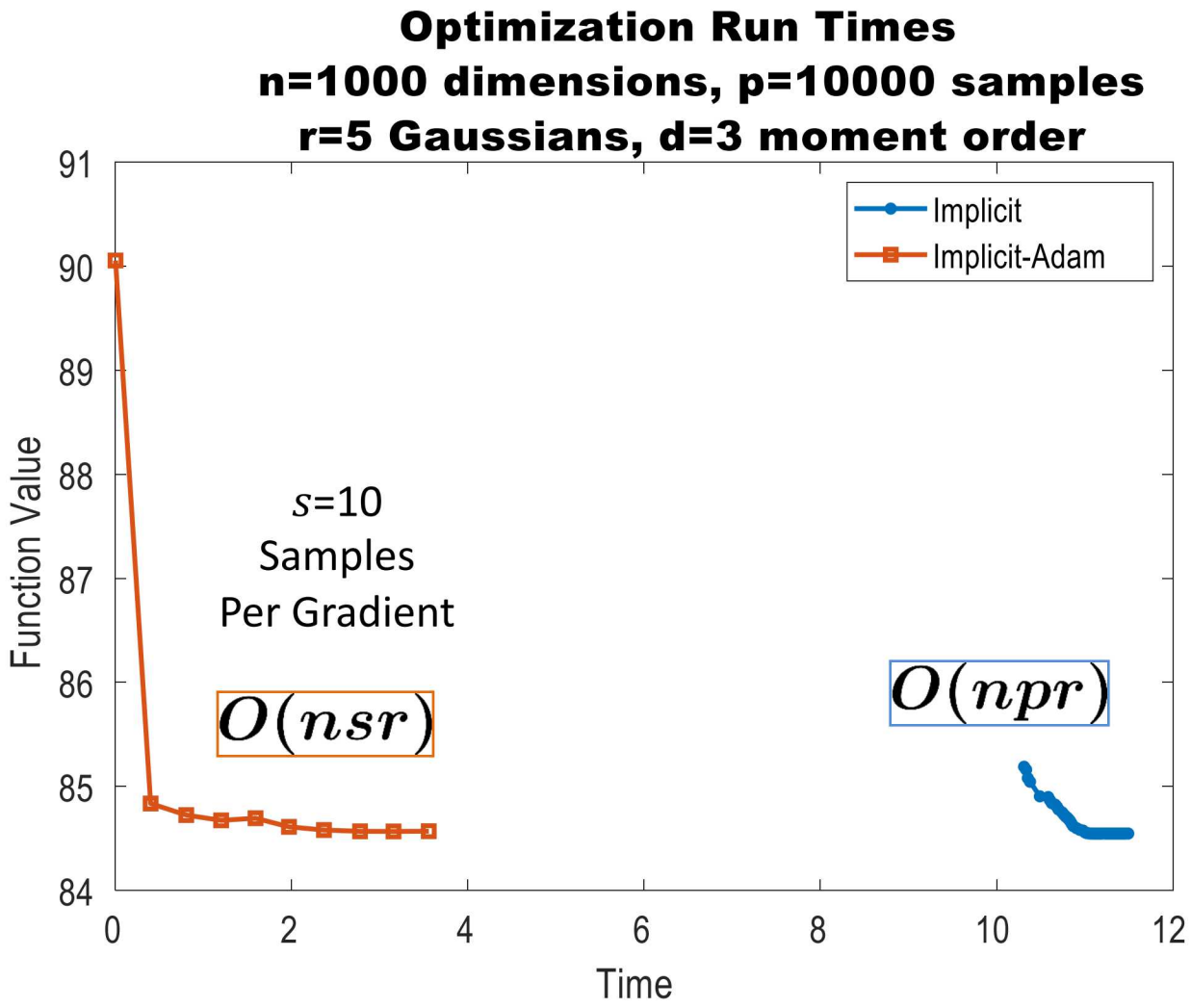


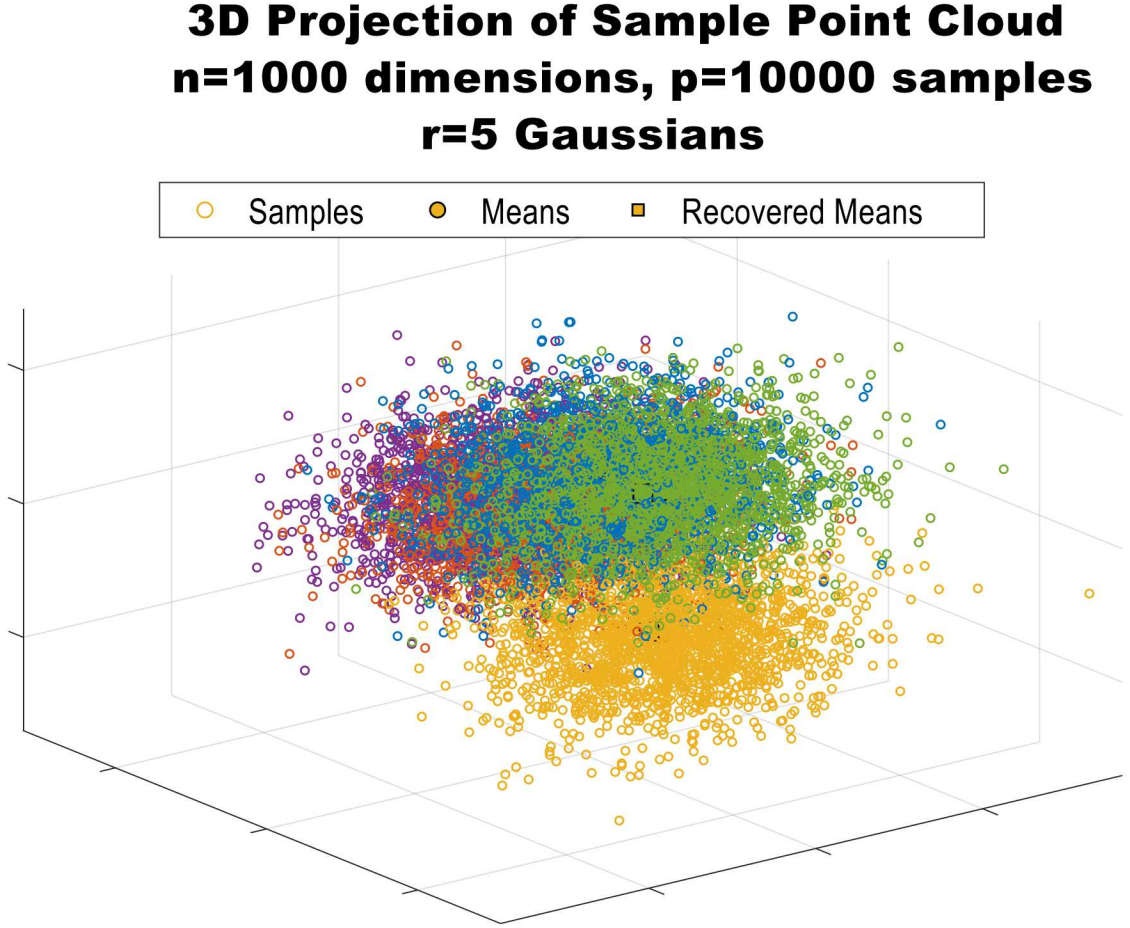
$$F(\mathcal{X}, \hat{\mathcal{X}}) = \tfrac{1}{2}\left(\mathbf{e}^T(\tilde{\mathbf{Z}}^T\tilde{\mathbf{Z}})^d\mathbf{e} + \boldsymbol{\lambda}^T(\mathbf{A}^T\mathbf{A})^d\boldsymbol{\lambda}\right) - \mathbf{e}^T(\tilde{\mathbf{Z}}^T\mathbf{A})^d\boldsymbol{\lambda}$$

$$\frac{\partial F}{\partial \mathbf{a}_j} = d\lambda_j\left(-\tilde{\mathbf{Z}}\cdot(\tilde{\mathbf{Z}}^T\mathbf{a}_j)^{d-1} + \sum_{k=1}^{r}\lambda_k\langle\mathbf{a}_j,\mathbf{a}_k\rangle^{d-1}\mathbf{a}_k\right)$$

$$\frac{\partial F}{\partial \lambda_j} = -\langle(\tilde{\mathbf{Z}}^T\mathbf{a}_j)^{d-1}, \tilde{\mathbf{Z}}^T\mathbf{a}_j\rangle + \sum_{k=1}^{r}\lambda_k\langle\mathbf{a}_j,\mathbf{a}_k\rangle^{d}$$

For Large Sample Size (p) Stochastic Optimization Much Faster, Same Accuracy





Conclusions & Future Work

- **CP tensor decomposition** is an unsupervised learning method that discovers low-rank multilinear structure in data
- **Symmetric CP tensor decomposition** can be used to discover a mixture of Gaussians with analogous higher-order moments
  - n = dimension
  - p = # samples
  - r = # Gaussians in mixture
  - $d \geq 3$ = empirical moment order
- **Implicit CP tensor decomposition** exploits the structure in the empirical moment matrix to reduce the complexity from O(pn^d) to O(pnr)
- **Stochastic implicit CP tensor decomposition** can handle large numbers of observations, reducing the complexity to O(snr) for $s \ll p$
- **Higher-order approach for finding mixtures of Gaussians** versus prior approaches that used on covariance information

Sam Sherman
Notre Dame

References & Acknowledgements

- **Implicit Symmetric CP (this work)** – S. Sherman, T. G. Kolda, **Estimating Higher-Order Moments Using Symmetric Tensor Decomposition**, coming soon!

- Funding for Kolda is from the DOE ASCR Applied Mathematics program, and funding for Sherman is from the NSF Math Sciences Graduate Intern (MSGI) program.

- **Symmetric CP** - T. G. Kolda. **Numerical Optimization for Symmetric Tensor Decomposition**. *Mathematical Programming B*, Vol. 151, No. 1, pp. 225-248, 2015. https://doi.org/10.1007/s10107-015-0895-0
- **Original mouse experiments** - A. H. Williams et al. **Unsupervised Discovery of Demixed, Low-dimensional Neural Dynamics across Multiple Timescales through Tensor Components Analysis**. *Neuron*, 98(6), 2018. https://doi.org/10.1016/j.neuron.2018.05.015
- **Generalized CP (GCP) Tensor Decomposition** - D. Hong, T. G. Kolda, J. A. Duersch. **Generalized Canonical Polyadic Tensor Decomposition**, SIAM Review (to appear). http://arxiv.org/abs/1808.07452

- July 10, 2019
  - 10am-Noon: Theory and Methods for Tensor Decomposition, MS163, Unitobler, F023
  - 5:15-6pm: Orthogonal Tensor Decomposition, Early Career Prize, Elina Robeva, vonRoll, Fabrikstr. 6, 001

Direct Optimization for Symmetric CP

Formulation:

$$\min_{\boldsymbol{\lambda},\mathbf{A}} \frac{1}{2}\|\mathcal{X} - \hat{\mathcal{X}}\|^2 \text{ where } \hat{\mathcal{X}} = \sum_{j=1}^{r} \lambda_j \, \mathbf{a}_j^{\otimes d}$$

Gradients:

$$\frac{\partial f}{\partial \mathbf{a}_j} = -d\lambda_j \underbrace{\mathcal{X}\mathbf{a}_j^{d-1}}_{\text{TTSV}} + d\lambda_j \sum_{k=1}^{r} \lambda_k \langle \mathbf{a}_j, \mathbf{a}_k \rangle^{d-1} \mathbf{a}_k$$

$$\frac{\partial f}{\partial \lambda_j} = -\underbrace{\mathcal{X}\mathbf{a}_j^{d}}_{\text{TTSV}} + \sum_{k=1}^{r} \lambda_k \langle \mathbf{a}_j, \mathbf{a}_k \rangle^{d}$$

Use gradients with standard first-order numerical optimization method such as steepest descent or limited-memory BFGS.

Norm

$$\|\mathcal{X}\|^2 = \sum_{i_1=1}^{n} \cdots \sum_{i_d=1}^{n} x_{i_1 i_2 \ldots i_d}^2$$

Vector Dot Product

$$\langle \mathbf{a}, \mathbf{b} \rangle = \sum_{i=1}^{n} a_i b_i$$

TTSV: Tensor Times Same Vector ($d-1$) times

$$(\mathcal{X}\mathbf{a}^{d-1})_{i_1} = \sum_{i_2=1}^{n} \cdots \sum_{i_d=1}^{n} x_{i_1 i_2 \ldots i_d} \prod_{k=2}^{d} a_{i_k}$$

TTSV: Tensor Times Same Vector d times

$$\mathcal{X}\mathbf{a}^{d} = \sum_{i_1=1}^{n} \cdots \sum_{i_d=1}^{n} x_{i_1 \ldots i_d} \prod_{k=1}^{d} a_{i_k} = \langle \mathcal{X}\mathbf{a}^{(d-1)}, \mathbf{a} \rangle$$