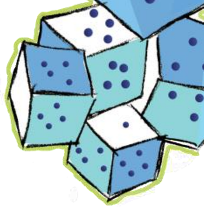
Illustration by Chris Brigman



Sandia National Laboratories

Randomized Methods for Low-Rank Tensor Decomposition

Tamara G. Kolda

Sandia National Labs, Livermore, CA

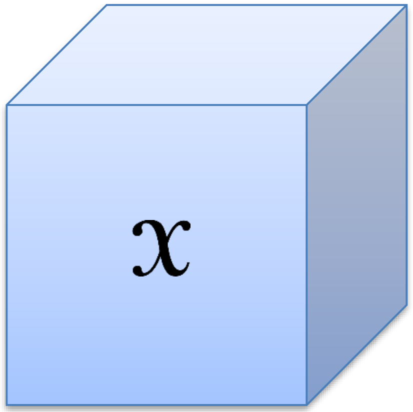
www.kolda.net (slides posted here)

Joint work with Grey Ballard (Wake Forrest), Casey Battaglino (Georgia Tech/ARM), Jed Duersch (Sandia), David Hong (Michigan/Penn), Ruhui Jin (Texas-Austin), Brett Larsen (Stanford), Samantha Sherman (Notre Dame), Rachel Ward (Texas-Austin), Alex Williams (Stanford)

Supported by the DOE Office of Science Advanced Scientific Computing Research (ASCR) Applied Mathematics program and Sandia's Laboratory Directed Research and Development (LDRD program). Sandia National Laboratories is 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.

U.S. DEPARTMENT OF ENERGY

Tensors are Multi-dimensional Arrays



3-way tensor

d = **order** of the tensor (the number of ways or modes)

n_k = **dimension** of mode k, for $k = 1, 2, \dots, d$

For expositional simplicity: $n = n_1 = n_2 \cdots = n_d$

n^d = number of entries for d-way tensor of dimension n

Curse of Dimensionality

Curse of notation…

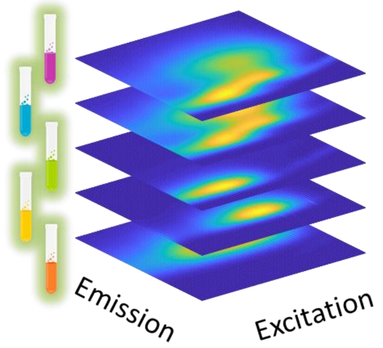
$i \equiv (i_1, i_2, \dots, i_d)$ = **index** into tensor, $i_k \in \{1, \dots, n\}$ for $k = 1, 2, \dots, d$

multi-index shorthand

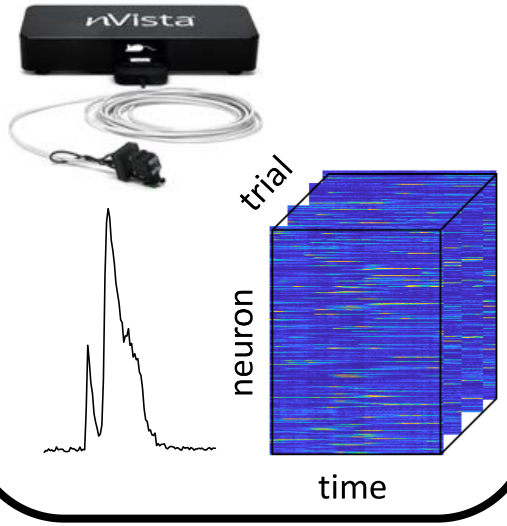
Tensors Come From Many Applications

- **Chemometrics:** Emission x Excitation x Samples (Fluorescence Spectroscopy)
- **Neuroscience:** Neuron x Time x Trial (Calcium Imaging)
- **Criminology:** Day x Hour x Location x Crime (Chicago)

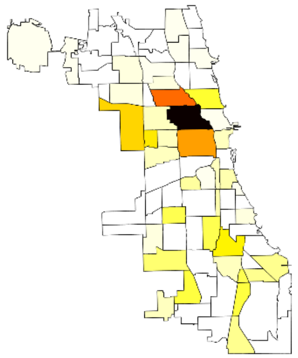
Chemometrics



Neuroscience



Criminology

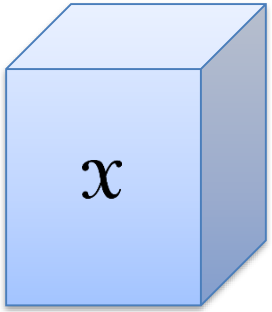


Related Concepts for Matrices

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

Goal is to Decompose Data Tensor

Data



$$\mathcal{X} \in \mathbb{R}^{n \times n \times \cdots \times n}$$

$$x_i = x(i_1, i_2, \ldots, i_d)$$

CANDECOMP/PARAFAC (CP) Model Depends on d Factor Matrices of Size n x r

$\mathcal{X} \in \mathbb{R}^{n \times n \times \cdots \times n}$

$x_i = x(i_1, i_2, \ldots, i_d)$

$\mathbf{A}_k \in \mathbb{R}^{n \times r}$

CP also known as Canonical Polyadic. Hitchcock (1927), Carroll, Chang (1970), Harshman (1970)

"Rank" of Low-Rank Model is the Number of Columns in the Factor Matrices

$\mathcal{X} \in \mathbb{R}^{n \times n \times \cdots \times n}$

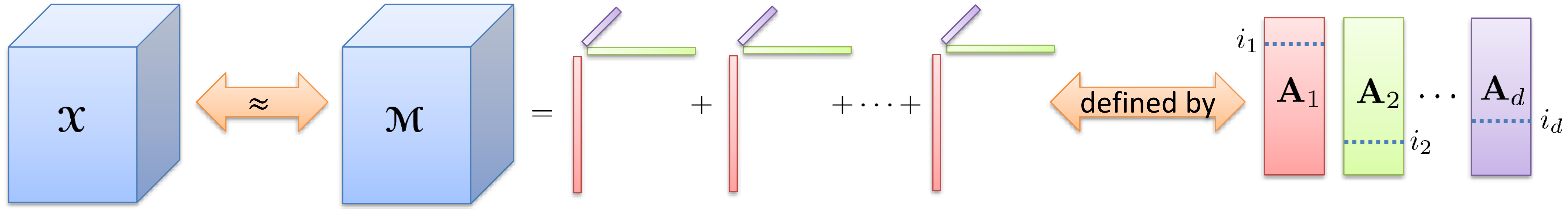
$x_i = x(i_1, i_2, \ldots, i_d)$

$\mathbf{A}_k \in \mathbb{R}^{n \times r}$

$\mathrm{rank}(\mathcal{M}) \leq r$

CP Model: Sum of Outer Products of Columns of Factor Matrices

Data | Model | Sum of r Outer Product Tensors | Factor Matrices



$\mathcal{X} \in \mathbb{R}^{n \times n \times \cdots \times n}$

$x_i = x(i_1, i_2, \ldots, i_d)$

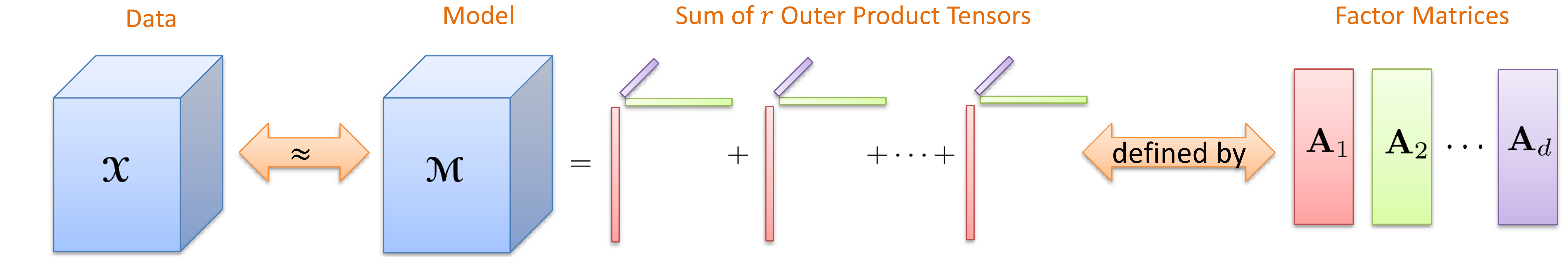
$\mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_d]\!] \in \mathbb{R}^{n \times n \times \cdots \times n}$

$m_i = m(i_1, i_2, \ldots, i_d) = \sum_{j=1}^{r} \prod_{k=1}^{d} a_k(i_k, j)$

$\mathbf{A}_k \in \mathbb{R}^{n \times r}$

$\mathrm{rank}(\mathcal{M}) \leq r$

CP Optimization Problem: Nonconvex Sum of Squared Errors



$\mathcal{X} \in \mathbb{R}^{n \times n \times \cdots \times n}$

$x_i = x(i_1, i_2, \ldots, i_d)$

$\mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_d]\!] \in \mathbb{R}^{n \times n \times \cdots \times n}$

$m_i = m(i_1, i_2, \ldots, i_d) = \sum_{j=1}^{r} \prod_{k=1}^{d} a_k(i_k, j)$

$\mathbf{A}_k \in \mathbb{R}^{n \times r}$

$\text{rank}(\mathcal{M}) \leq r$

CP Optimization

$$\min_{\mathbf{A}_1, \ldots, \mathbf{A}_d} \quad \|\mathcal{X} - \mathcal{M}\|^2 \equiv \sum_{i=1}^{n^d} (x_i - m_i)^2$$

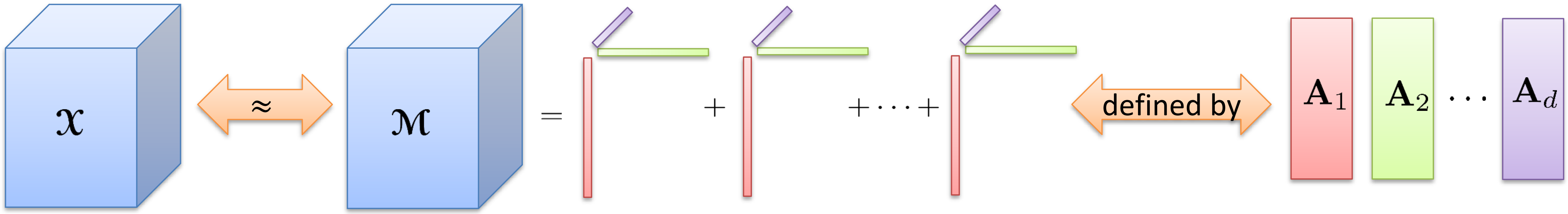
$$\text{s.t.} \quad \mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_d]\!]$$

$$\mathbf{A}_k \in \mathbb{R}^{n \times r} \text{ for } k = 1, \ldots, d$$

CP = CANDECOMP/PARAFAC, also known as Canonical Polyadic. Hitchcock (1927), Carroll, Chang (1970), Harshman (1970)

Low-Dimensional Manifold, Reducing Storage and Increasing Interpretability

Data | Model | Sum of r Outer Product Tensors | Factor Matrices



$\mathcal{X} \in \mathbb{R}^{n \times n \times \cdots \times n}$

$x_i = x(i_1, i_2, \dots, i_d)$

$\mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_d]\!] \in \mathbb{R}^{n \times n \times \cdots \times n}$

$m_i = m(i_1, i_2, \dots, i_d) = \sum_{j=1}^{r} \prod_{k=1}^{d} a_k(i_k, j)$

$\mathbf{A}_k \in \mathbb{R}^{n \times r}$

$\operatorname{rank}(\mathcal{M}) \le r$

Storage for original data: n^d

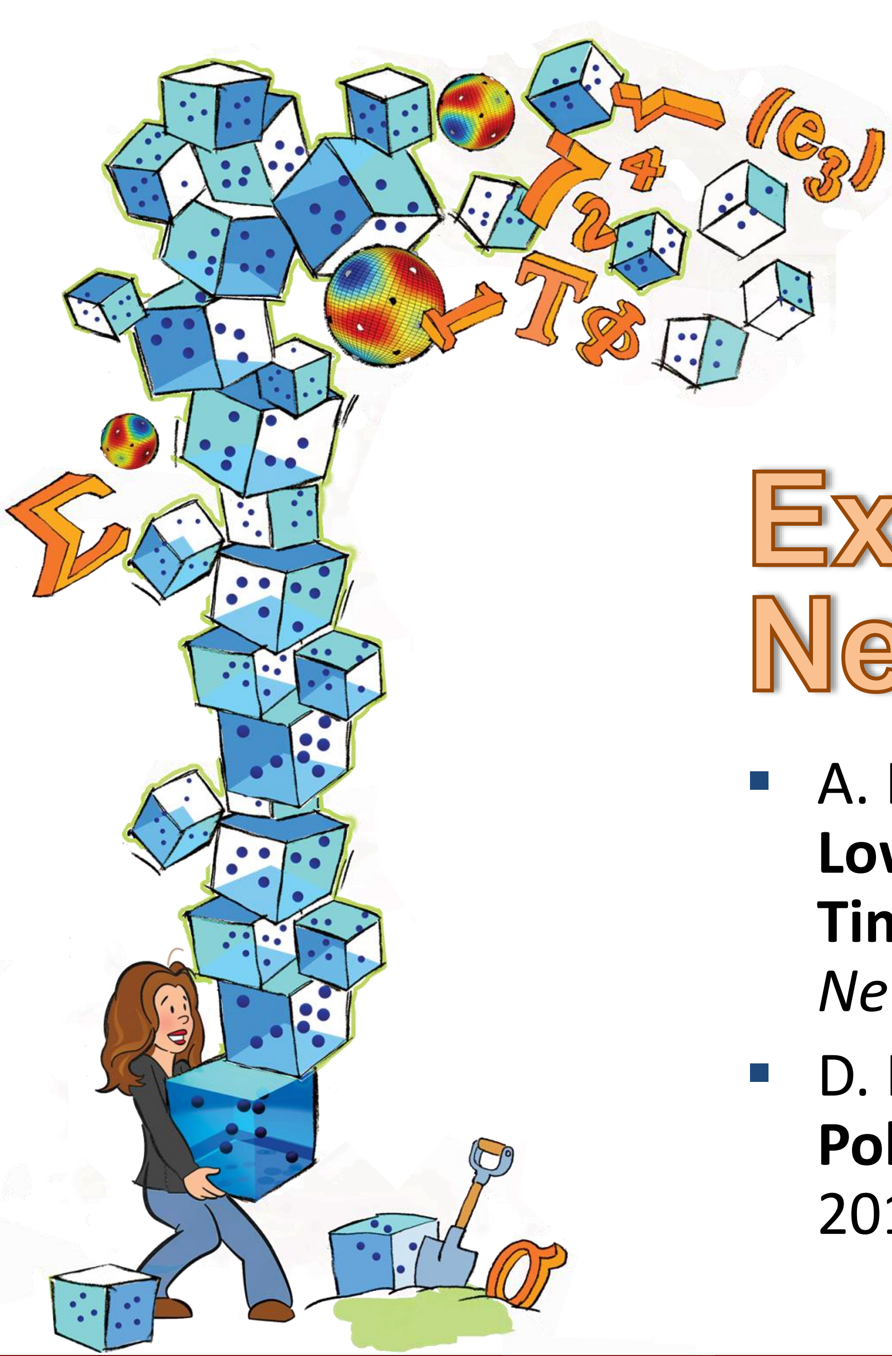
CP Optimization

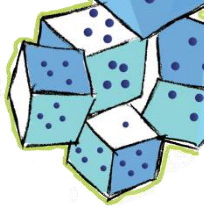
$$\min_{\mathbf{A}_1,\dots,\mathbf{A}_d} \quad \|\mathcal{X} - \mathcal{M}\|^2 \equiv \sum_{i=1}^{n^d} (x_i - m_i)^2$$

$$\text{s.t.} \quad \mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_d]\!]$$

$$\mathbf{A}_k \in \mathbb{R}^{n \times r} \text{ for } k = 1, \dots, d$$

Storage for low-rank model: dnr





Example Tensor from Neuroscience

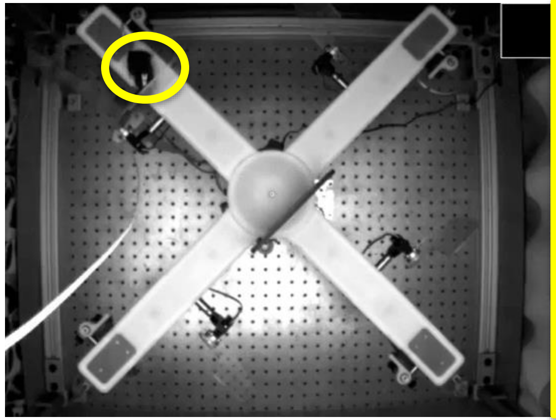
- A. H. Williams et al. **Unsupervised Discovery of Demixed, Low-dimensional Neural Dynamics across Multiple Timescales through Tensor Components Analysis**. *Neuron,* 2018
- D. Hong, T. G. Kolda, J. A. Duersch. **Generalized Canonical Polyadic Tensor Decomposition**. *SIAM Review*, in press, 2019

Activity of Single Neuron Measured Over Time Produces Vector Data

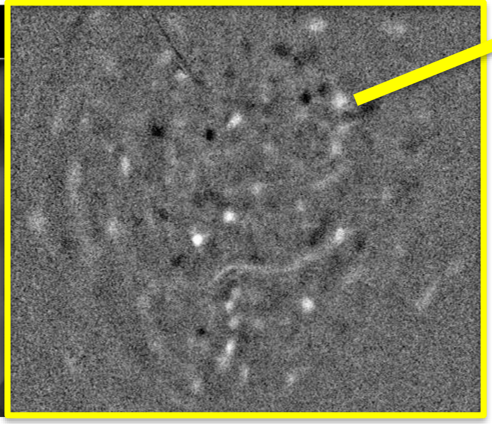
Thanks to Schnitzer Group @ Stanford
Mark Schnitzer, Fori Wang, Tony Kim

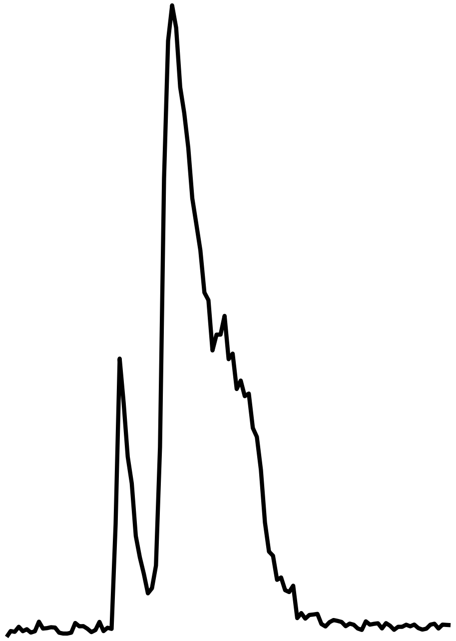
Microscope by Inscopix

mouse
in maze



neural activity via
calcium imaging





Williams et al., Neuron, 2018

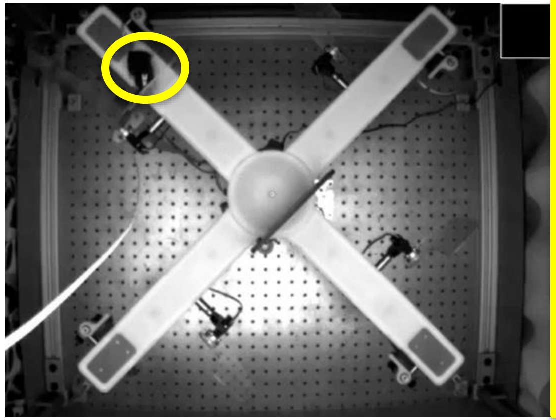
Activity of Single Neuron Measured Over Time Produces Vector Data

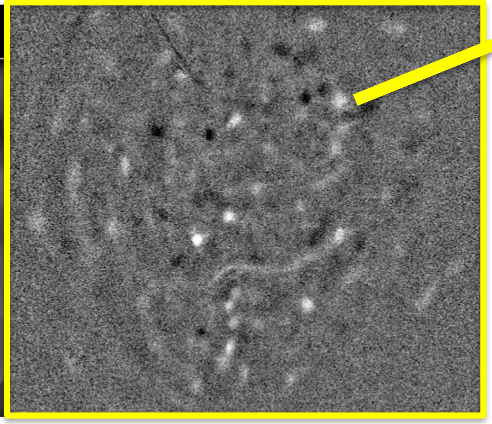
Thanks to Schnitzer Group @ Stanford
Mark Schnitzer, Fori Wang, Tony Kim

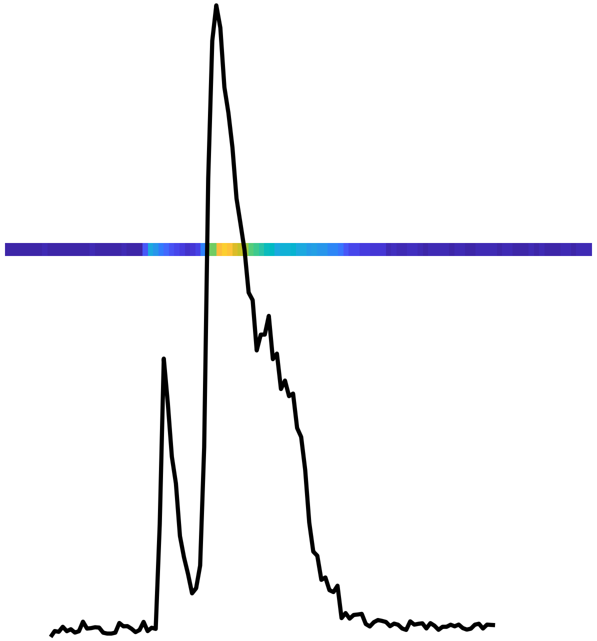
Microscope by Inscopix

mouse in maze

neural activity via calcium imaging







Williams et al., Neuron, 2018

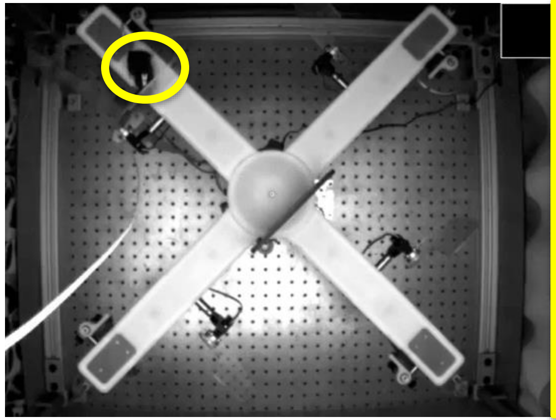
Multiple Neurons Measured Over Time Produces Matrix

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

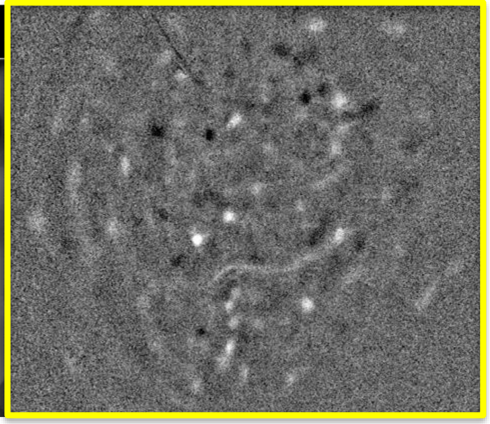
282 neurons × 111 time bins

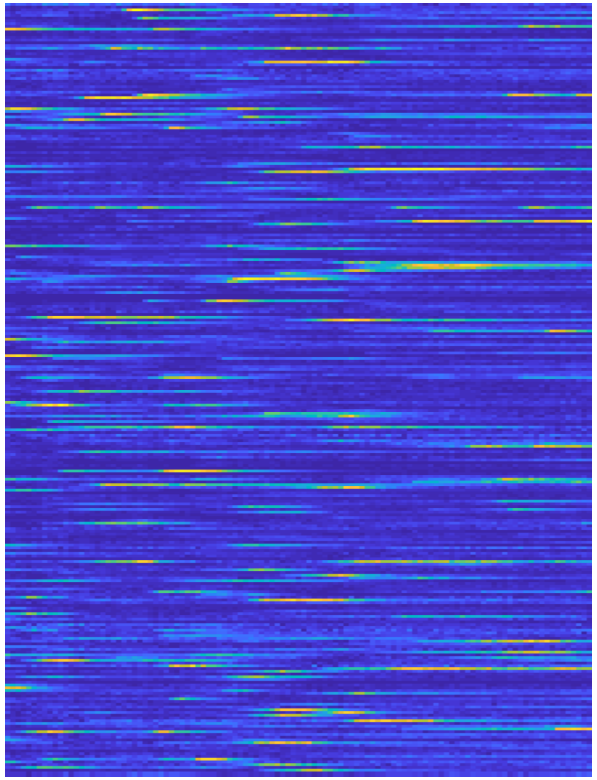
Microscope by Inscopix

mouse in "maze"



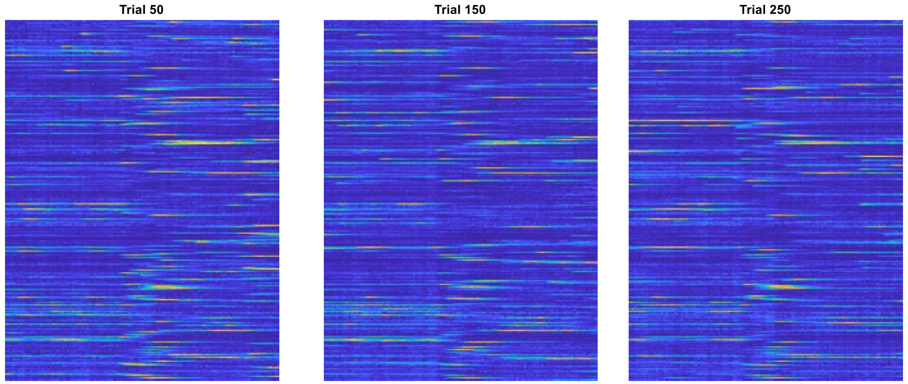
neural activity

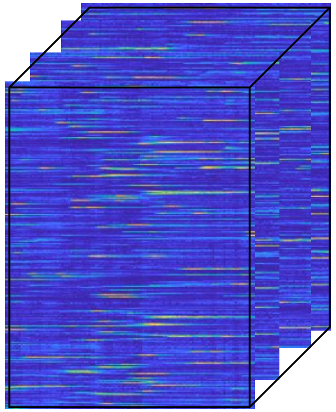


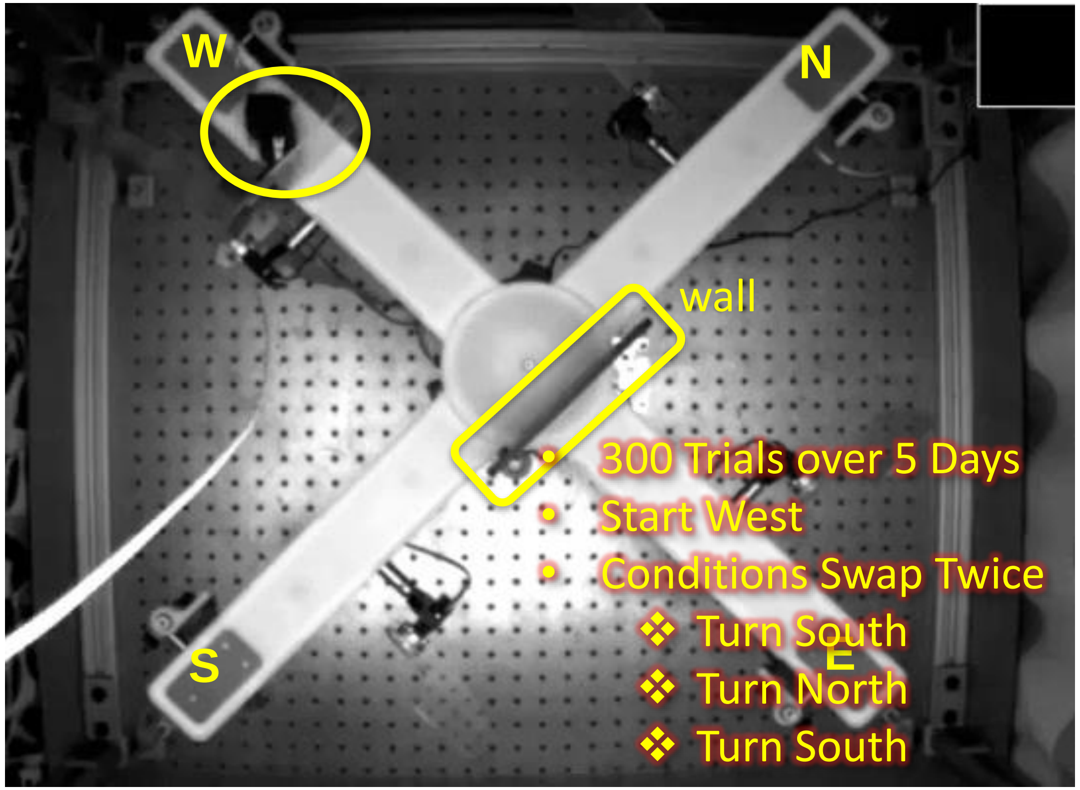


Williams et al., Neuron, 2018

Multiple Trials Produces 3-way Tensor







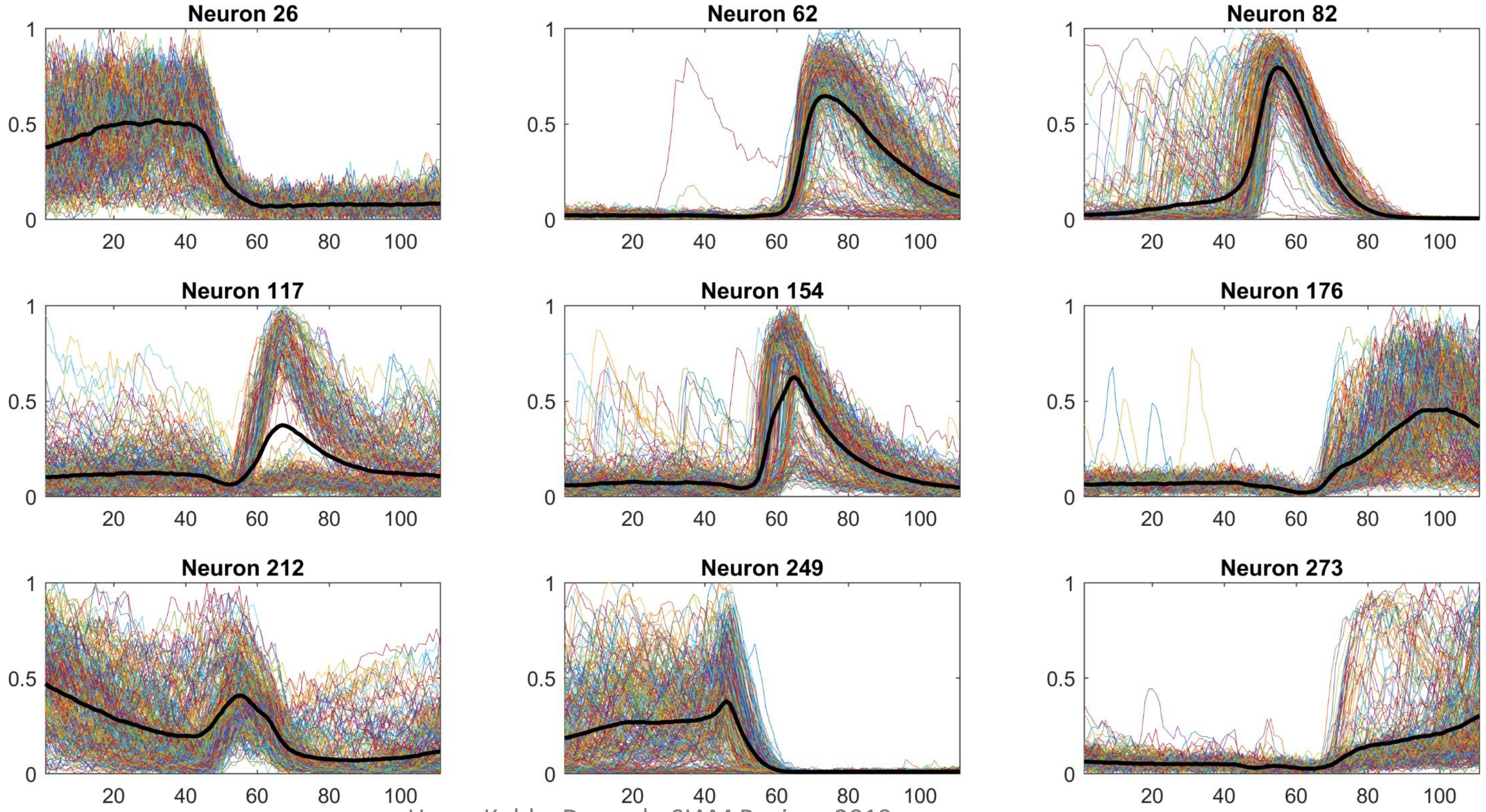
282 neurons × 111 time bins × 300 trials

Williams et al., Neuron, 2018

Example Neuron Activity

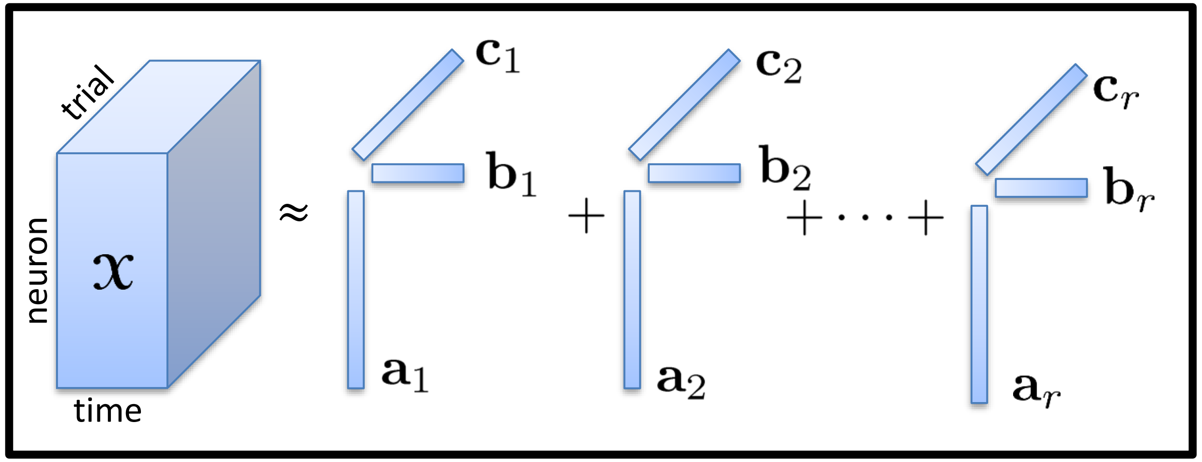
Thin lines show 300 individual trials

Thick line is average



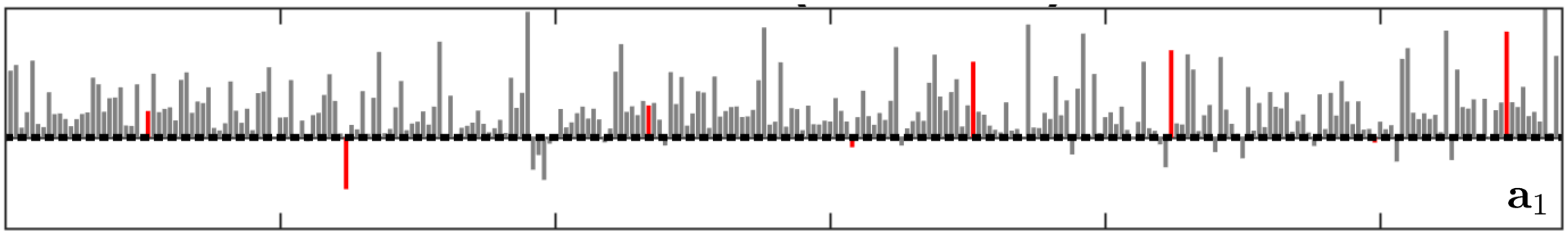
Hong, Kolda, Duersch, SIAM Review, 2019

Neuron Factor Vector Visualized as Bar Chart

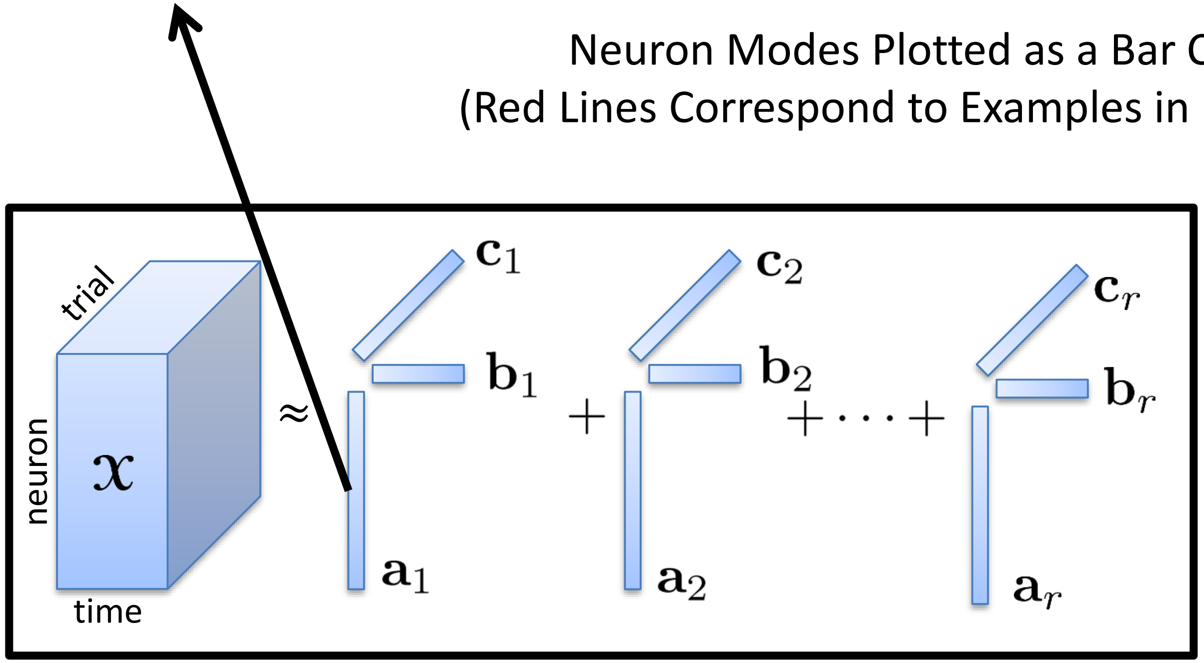


Hong, Kolda, Duersch, SIAM Review, 2019

Neuron Factor Vector Visualized as Bar Chart

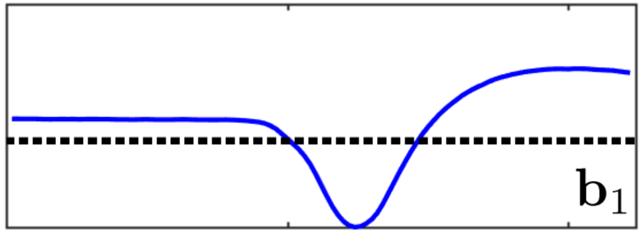


Neuron Modes Plotted as a Bar Chart
(Red Lines Correspond to Examples in Prior Slide)

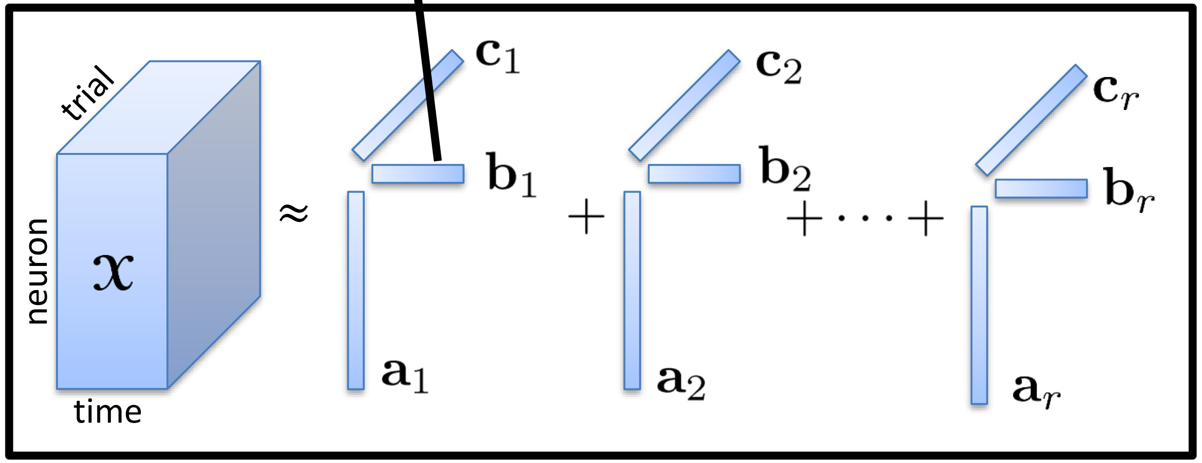


Hong, Kolda, Duersch, SIAM Review, 2019

Time Factor Vector Visualized as Line

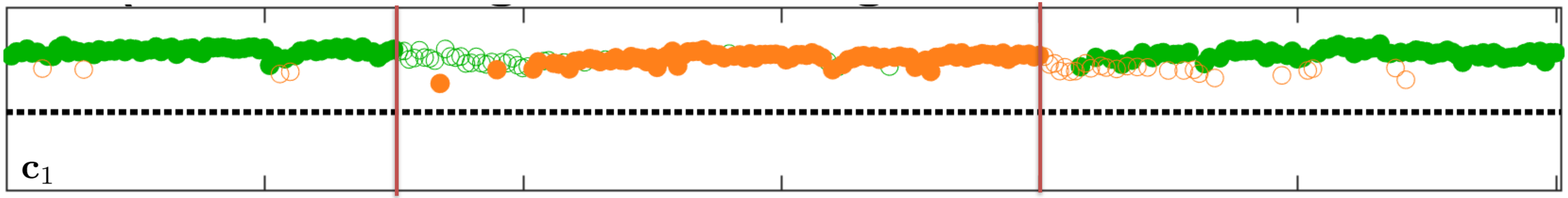


Time (within trial) Plotted as a Line
(Dashed Line is Zero)



Hong, Kolda, Duersch, SIAM Review, 2019

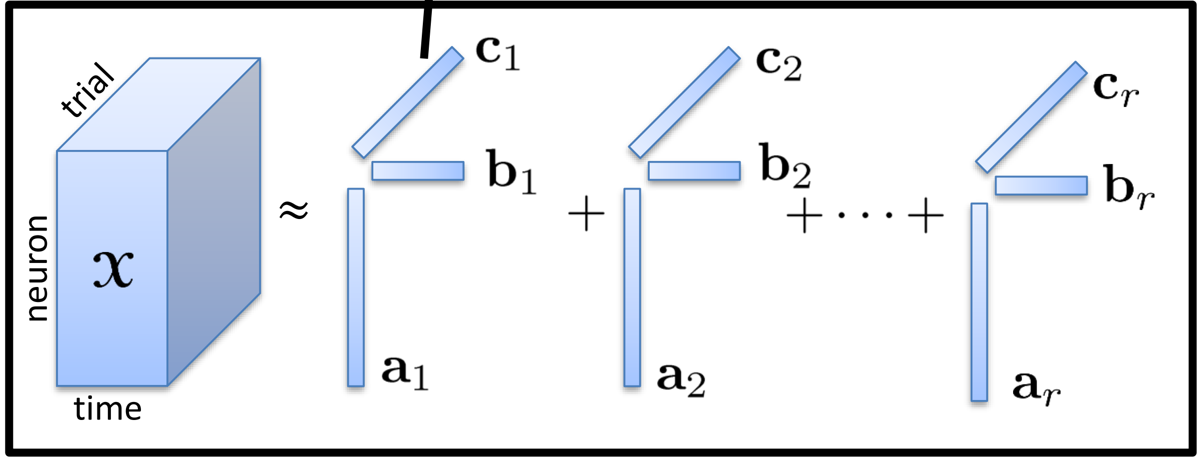
Trial Factor Vector Visualized as Color-Coded Scatter Plot



Rule Change

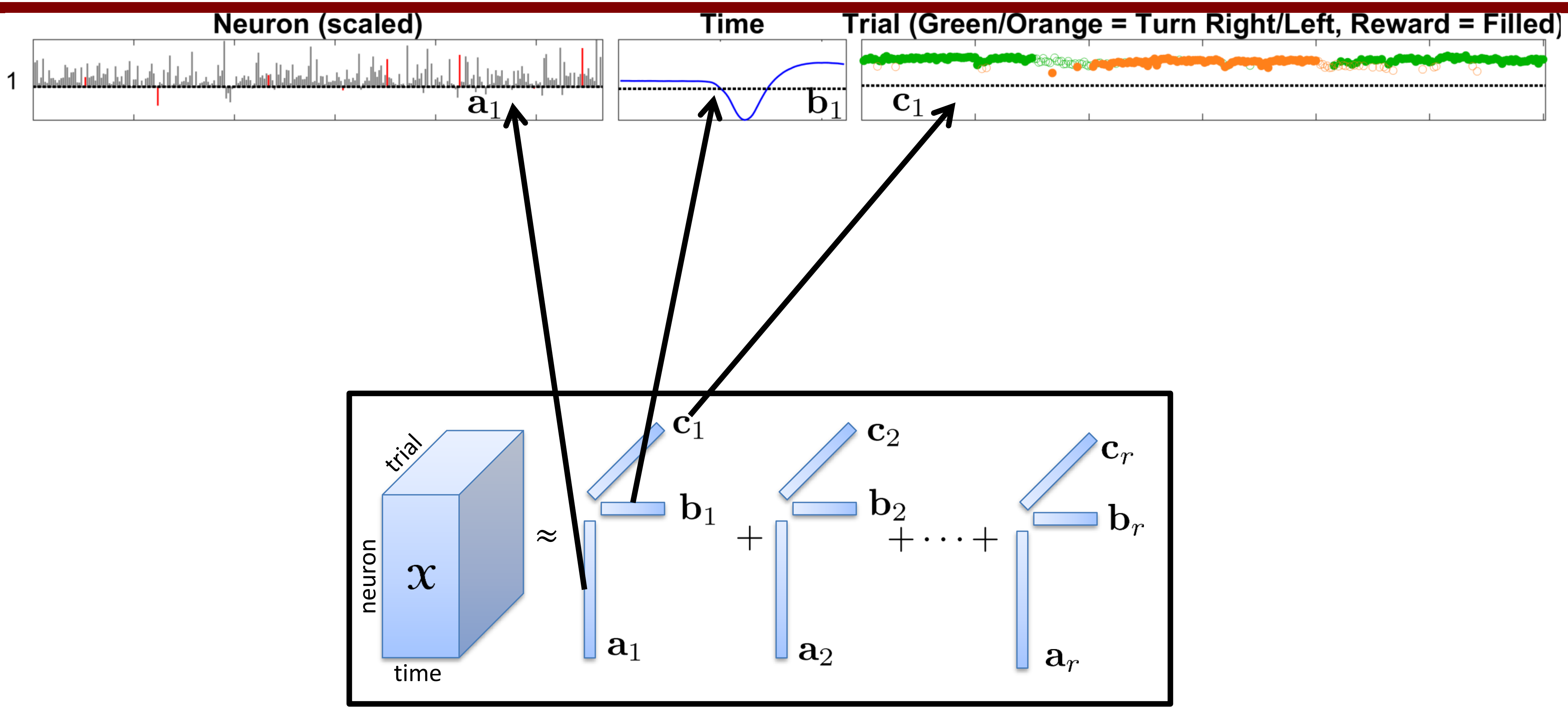
Trial Plotted as Scatter Graph
Right turn = Green
Left turn = Orange
Filled = Reward

Rule Change



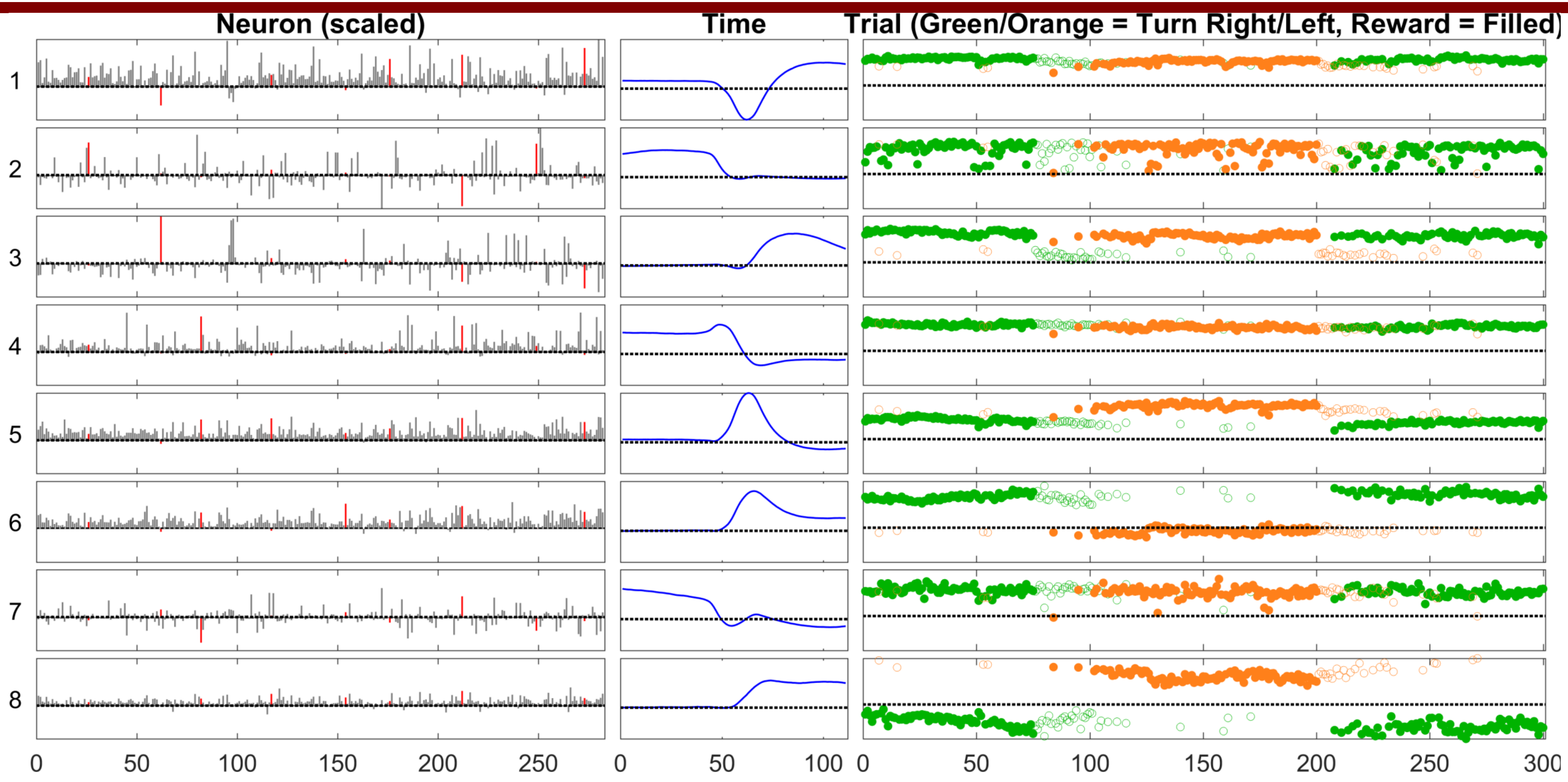
Hong, Kolda, Duersch, SIAM Review, 2019

Visualization of CP Tensor Decomposition Shows the Factors (Vectors)

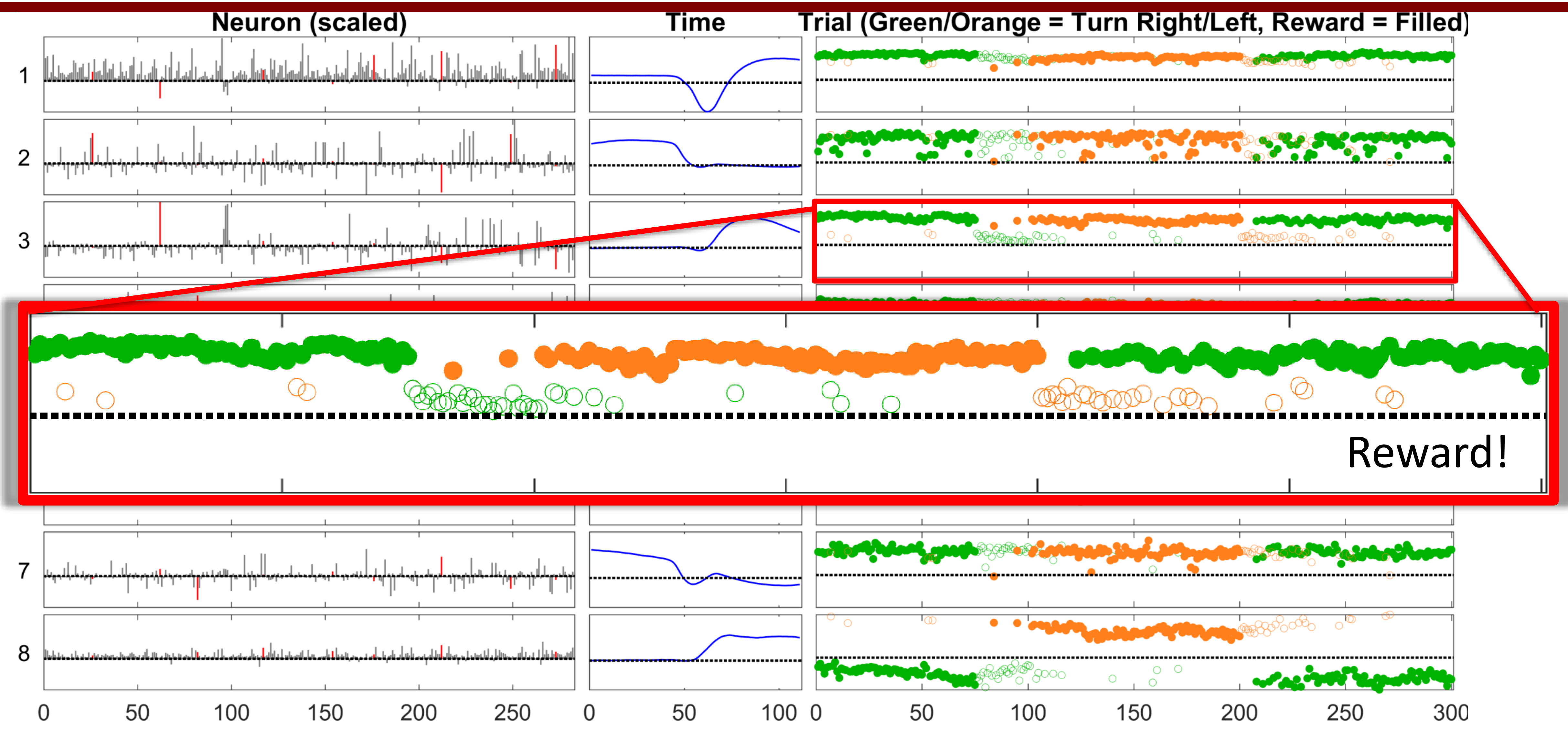


Hong, Kolda, Duersch, SIAM Review, 2019

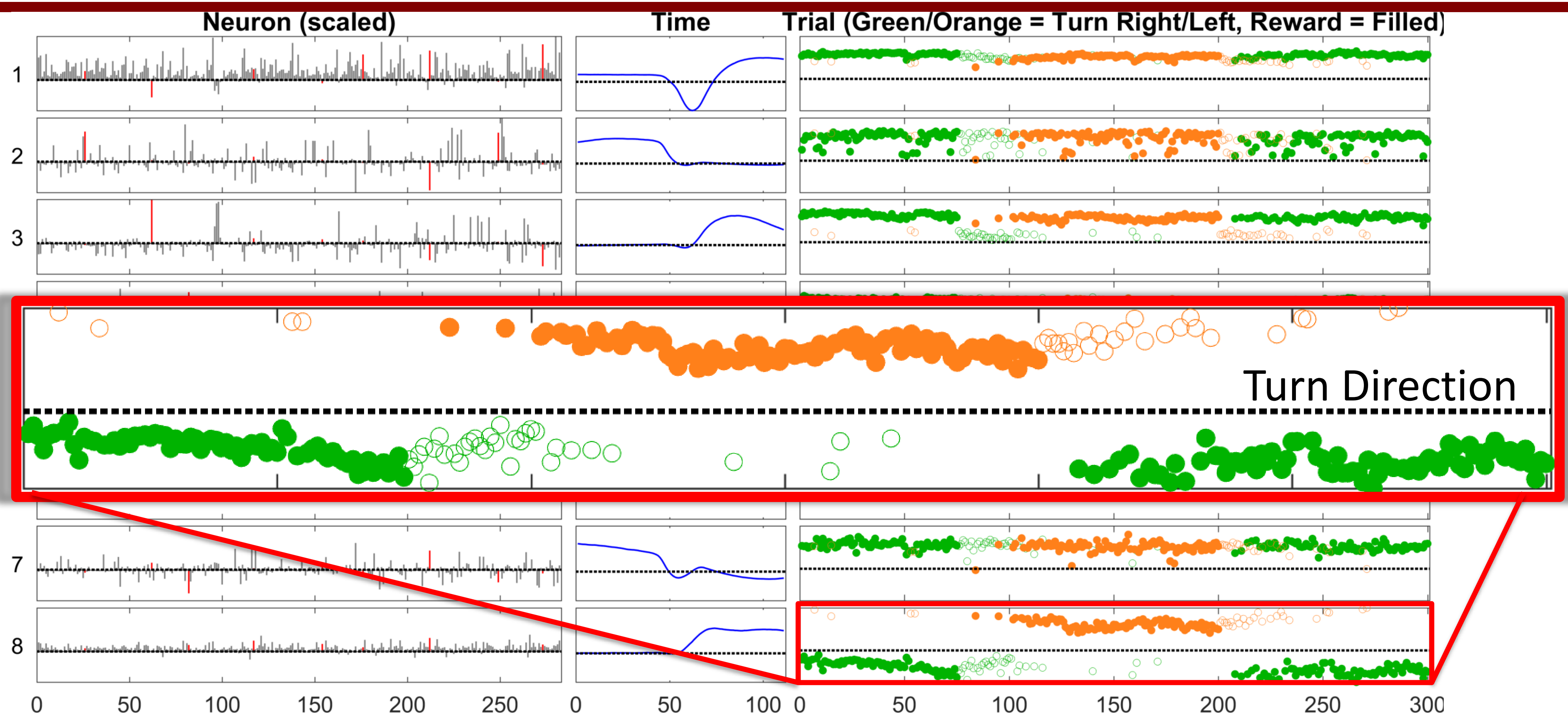
CP Decomposition of Mouse Data



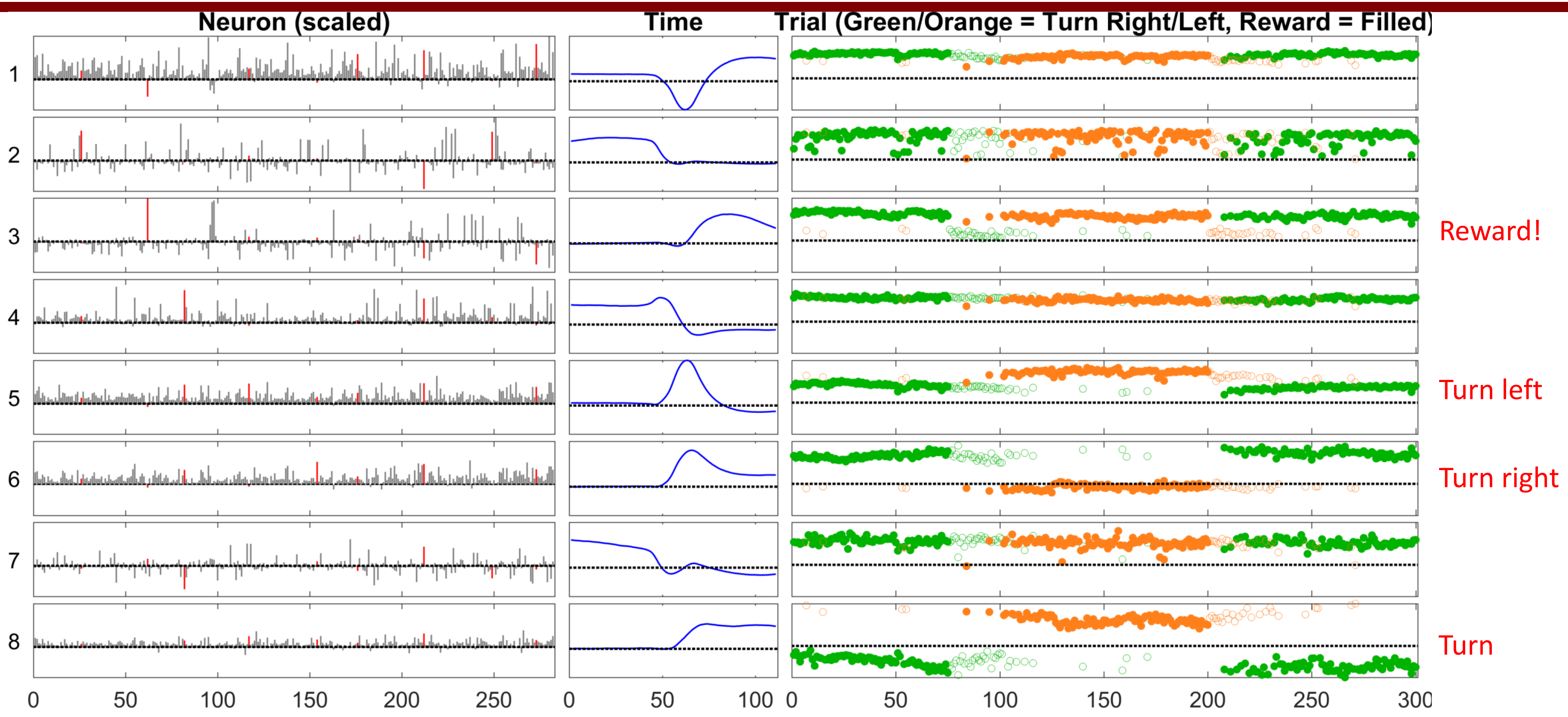
CP Tensor Decomposition “Sees” Reward



CP Tensor Decomposition “Sees” Turn Direction



CP Tensor Decomposition Yields Interpretation of a Complex Dataset



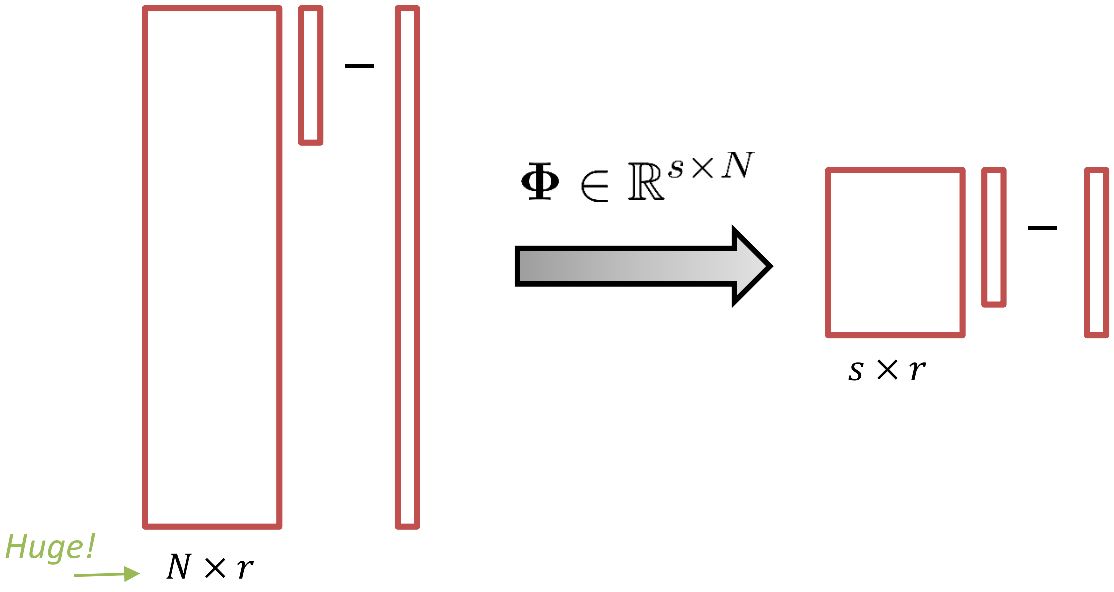
What about huge data (large n)?

Just computing the sum of squared errors objective function is $O(n^d)$ work!

Randomization Offers Powerful Tools, But Doesn't Always Work "Out of the Box"

Matrix Sketching

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 \qquad \hat{\mathbf{x}}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \|\mathbf{\Phi}\mathbf{A}\mathbf{x} - \mathbf{\Phi}\mathbf{b}\|_F^2$$



Under *suitable conditions* on $\mathbf{\Phi}$, achieve ϵ-distortion whp:

$$(1-\epsilon)\|\mathbf{A}\mathbf{x}^* - \mathbf{b}\| \leq \|\mathbf{A}\hat{\mathbf{x}}^* - \mathbf{b}\| \leq (1+\epsilon)\|\mathbf{A}\mathbf{x}^* - \mathbf{b}\|$$

Johnson, Lindenstrauss (1984), Ailon, Chazelle (2006), Woodruff (2014)

Stochastic Gradient Descent

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^d} \frac{1}{N} \sum_{i=1}^{N} f_i(\boldsymbol{\theta})$$

Huge!

Stochastic gradient:

$$\tilde{\mathbf{g}}(\boldsymbol{\theta}) = \frac{1}{|\tilde{\mathcal{S}}|} \sum_{i \in \tilde{\mathcal{S}}} \nabla f_i(\boldsymbol{\theta})$$

$|\tilde{\mathcal{S}}| \ll N$

random subset of indices

True gradient in expectation

$$\mathbb{E}[\tilde{\mathbf{g}}(\boldsymbol{\theta})] = \frac{1}{N} \sum_{i=1}^{N} \nabla f_i(\boldsymbol{\theta})$$

Stochastic gradient descent (SGD):

$$\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} - \alpha\, \tilde{\mathbf{g}}(\boldsymbol{\theta}^{(t)})$$

For *suitable choice(s)* of α, achieve SGD eventually converges to a stationary point.

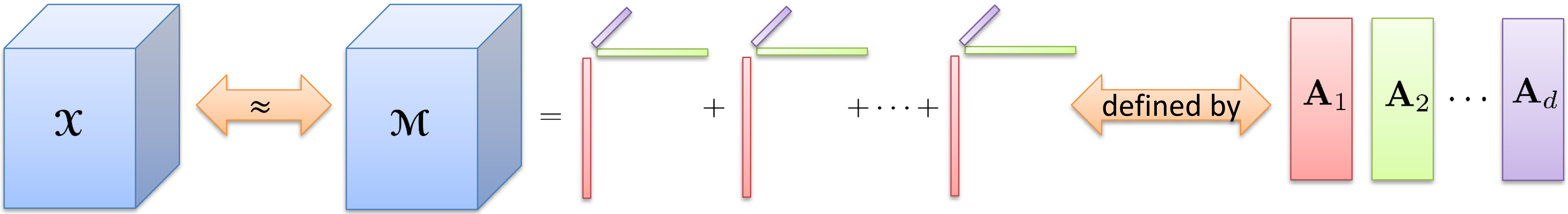
Robbins, Monro (1951), Bottou, Curtis, Nocedal (2018)

Matrix Sketching for CP Tensor Decomposition

- C. Battaglino, G. Ballard, T. G. Kolda. **A Practical Randomized CP Tensor Decomposition**. *SIAM Journal on Matrix Analysis and Applications*, 2018
- R. Jin, T. G. Kolda, R. Ward. **Faster Johnson-Lindenstrauss Transforms via Kronecker Product.** Coming soon, 2019
- T. G. Kolda, B. Larsen. **Leverage Score Sampling for Randomized CP Tensor Decomposition.** Coming soon, 2019

Recall the Optimization Problem

Data | Model | Sum of r Outer Product Tensors | Factor Matrices



$\mathcal{X} \in \mathbb{R}^{n \times n \times \cdots \times n}$

$x_i = x(i_1, i_2, \ldots, i_d)$

$\mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_d]\!] \in \mathbb{R}^{n \times n \times \cdots \times n}$

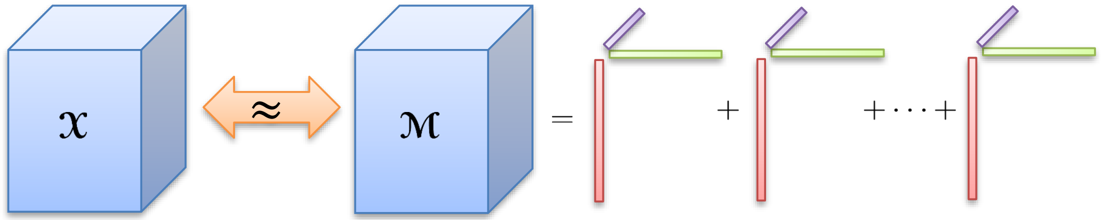
$m_i = m(i_1, i_2, \ldots, i_d) = \sum_{j=1}^{r} \prod_{k=1}^{d} a_k(i_k, j)$

$\mathbf{A}_k \in \mathbb{R}^{n \times r}$

$\mathrm{rank}(\mathcal{M}) \leq r$

$$
\begin{aligned}
\min_{\mathbf{A}_1, \ldots, \mathbf{A}_d} \quad & \|\mathcal{X} - \mathcal{M}\|^2 \equiv \sum_{i=1}^{n^d} (x_i - m_i)^2 \\
\text{s.t.} \quad & \mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_d]\!] \\
& \mathbf{A}_k \in \mathbb{R}^{n \times r} \text{ for } k = 1, \ldots, d
\end{aligned}
$$

We Can Rewrite the Tensor Optimization in Terms of Matrices



$$\min_{\mathbf{A}_1,\ldots,\mathbf{A}_d} \quad \|\mathcal{X}-\mathcal{M}\|^2 \equiv \sum_{i=1}^{n^d}(x_i - m_i)^2$$

$$\text{s.t.} \quad \mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_d]\!]$$

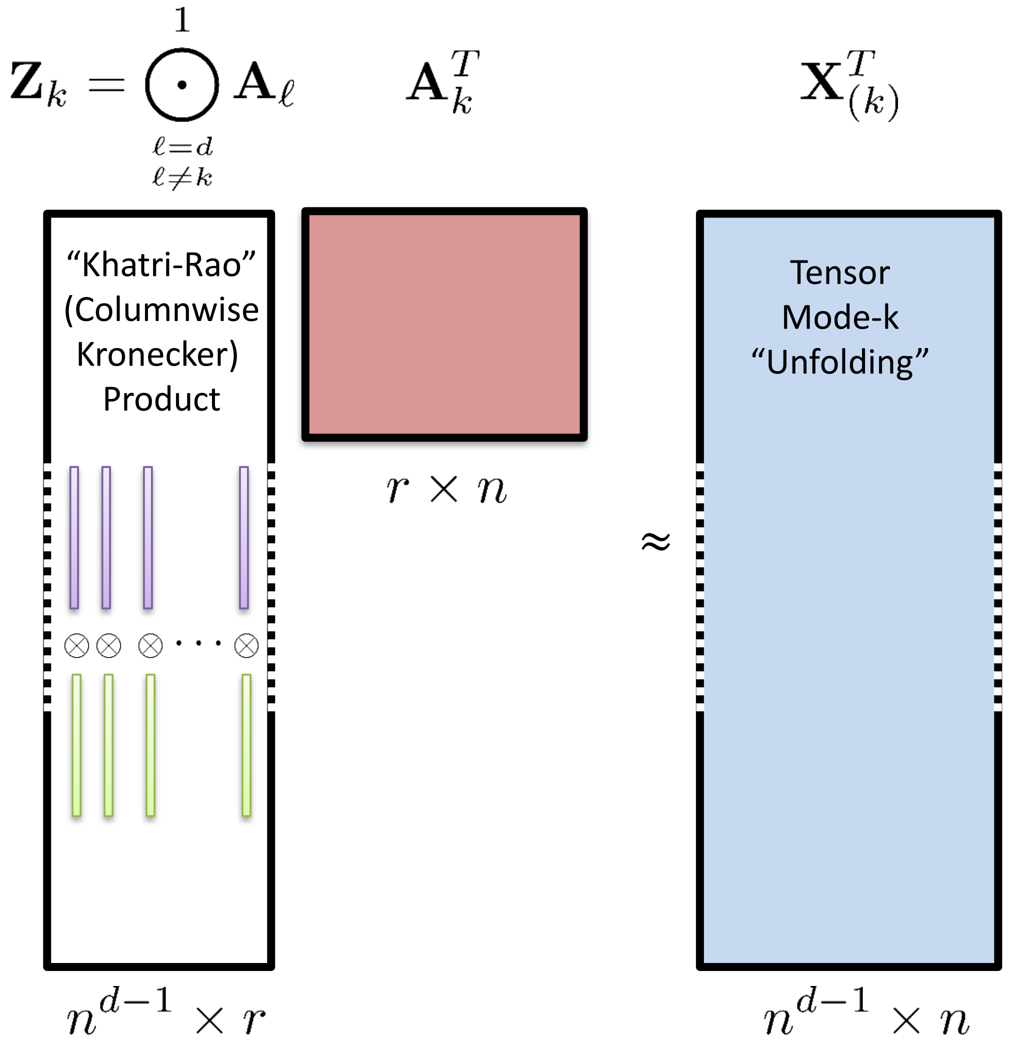
$$\mathbf{A}_k \in \mathbb{R}^{n\times r} \ \text{ for } k = 1,\ldots,d$$

$$\Downarrow$$

$$\min_{\mathbf{A}_1,\ldots,\mathbf{A}_d} \quad \|\mathcal{X}-\mathcal{M}\|^2 = \|\mathbf{X}_{(k)} - \mathbf{A}_k\mathbf{Z}_k^T\|^2$$

$$\text{s.t.} \quad \mathbf{Z}_k = \bigodot_{\ell\neq k} \mathbf{A}_\ell$$

$$\mathbf{A}_k \in \mathbb{R}^{n\times r} \ \text{ for } k = 1,\ldots,d$$



Matrix Version Leads To Alternating Least Squares (ALS) Optimization Algorithm

Alternating Least Squares (CP-ALS)

> **while** not converged **do**
>   **for** $k = 1, \ldots, d$ **do**
> $\mathbf{A}_k \leftarrow \arg\min_{\mathbf{A}_k} \|\mathbf{X}_{(k)} - \mathbf{A}_k \mathbf{Z}_k^T\|^2$
>   **end for**
> **end while**

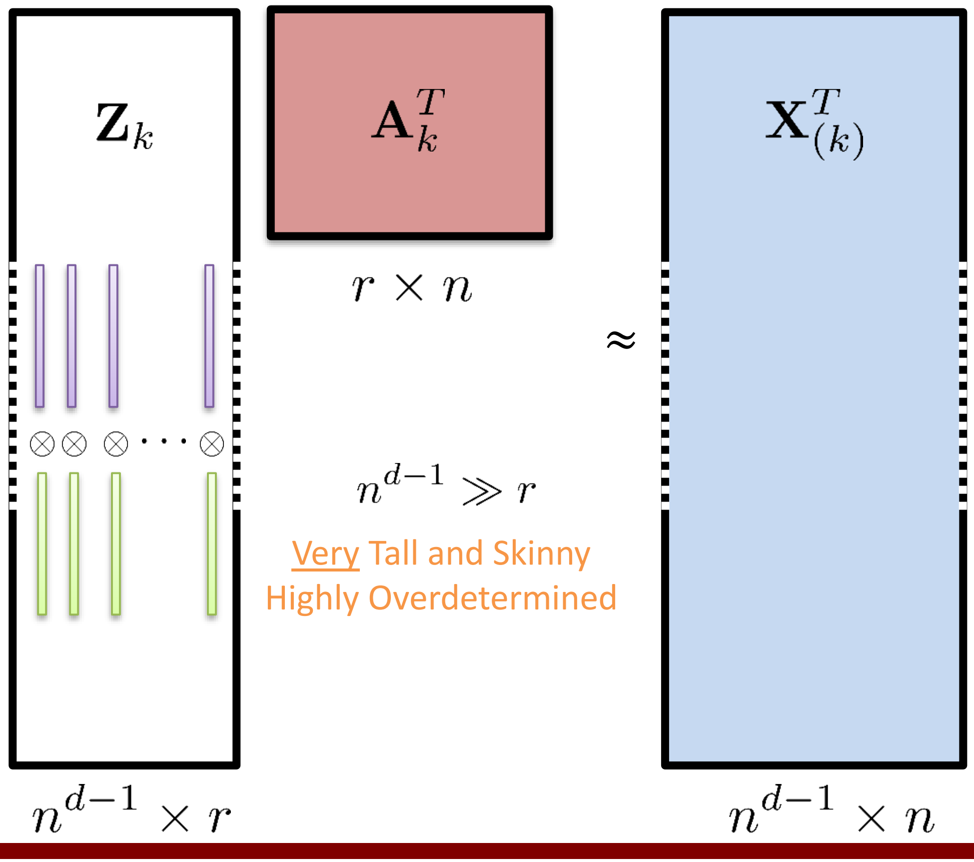
Constructing $\mathbf{Z}_k : O(rn^{d-1})$

Computational complexity per least squares solve: $O(r^2 n^d)$

Idea: Use matrix sketching to solve the highly overdetermined systems

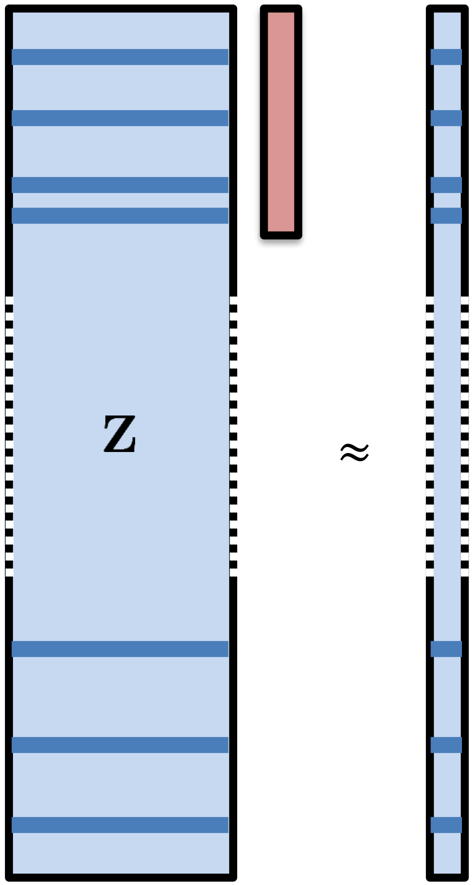
$$\min_{\mathbf{A}_k} \quad \|\mathcal{X} - \mathcal{M}\|^2 = \|\mathbf{X}_{(k)} - \mathbf{A}_k \mathbf{Z}_k^T\|^2$$

$$\text{s.t.} \quad \mathbf{Z}_k = \bigodot_{\ell \neq k} \mathbf{A}_\ell$$



Option 1: Johnson-Lindenstraus Transform (Mixing and Sampling)

$$\min_{\mathbf{a}} \|\mathbf{Z}\mathbf{a} - \mathbf{x}\|^2$$



$N \times r$

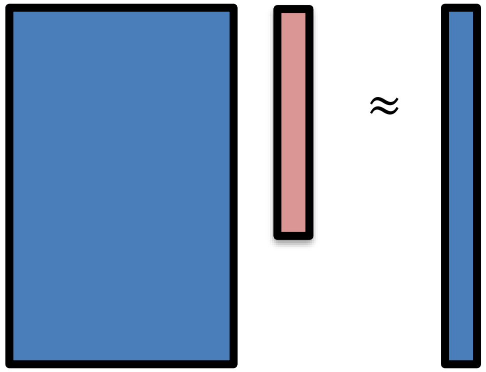
Computational complexity: $O(Nr^2)$ versus $O(rsN + sr^2)$

Sample Mixed Rows

$\mathbf{S} = s$ random rows of $\mathbf{I}_N$

$\mathbf{\Phi} = N \times N$, entries $\mathcal{N}(0,1)$

$$\min_{\mathbf{a}} \|\mathbf{S\Phi Z a} - \mathbf{S\Phi x}\|^2$$



$s \times r$

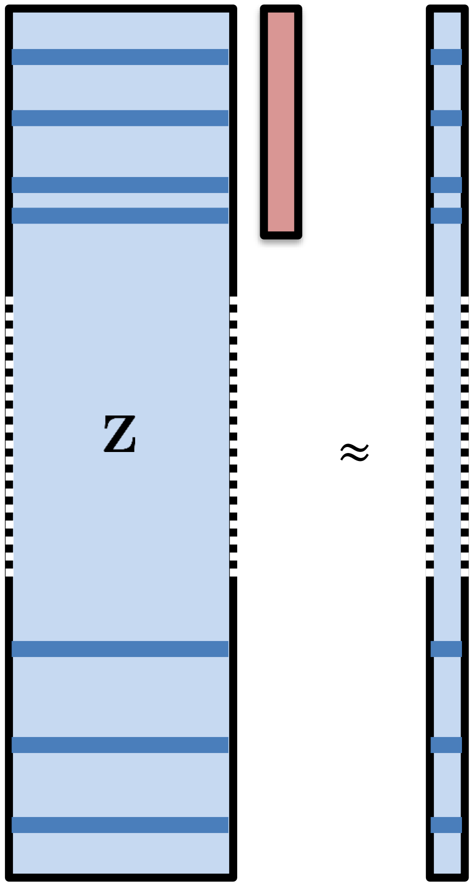
Theoretical sample size

$s = O(\epsilon^{-2} \log r)$

No significant reduction in computational complexity due to cost in applying $\mathbf{\Phi}$*!*

Option 2: Fast JLT (FJLT)

$$\min_{\mathbf{a}} \|\mathbf{Z}\mathbf{a} - \mathbf{x}\|^2$$



$N \times r$

Computational complexity: $O(Nr^2)$ versus $O(rN\log N + sr^2)$

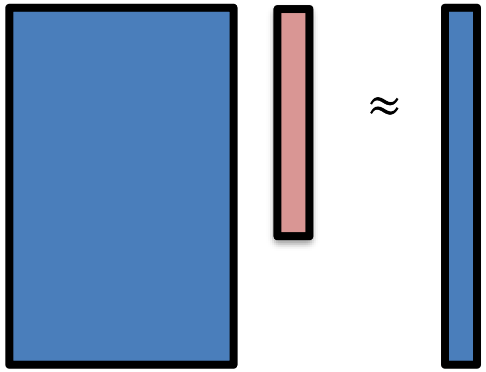
Sample Mixed Rows

$\mathbf{S} = s$ random rows of $\mathbf{I}_N$

$\mathcal{F} =$ FFT of size N

$\mathbf{D} =$ diagonal random ± 1

$$\min_{\mathbf{a}} \|\mathbf{S}\mathcal{F}\mathbf{D}\mathbf{Z}\mathbf{a} - \mathbf{S}\mathcal{F}\mathbf{D}\mathbf{x}\|^2$$



$s \times r$

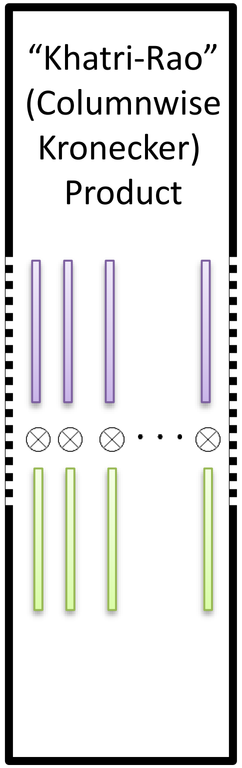
Theoretical sample size

$$s = O(\epsilon^{-2}\log r\,\log N)$$

FFT helps, but still dependence on N!

Use Kronecker Structure to Reduce Computational Complexity

$$\mathbf{Z}_k = \bigodot_{\substack{\ell=d \\ \ell\neq k}}^{1} \mathbf{A}_\ell$$



$N \times r$

Recall $N = n^{(d-1)}$

FJLT mixing: $\mathbf{S}\mathcal{F}_N\mathbf{D}_N\mathbf{Z}_k$

Complexity: $O(rN \log N + sr^2)$

$$= O(rn^{(d-1)} \log n + sr^2)$$

$$\mathbf{S} = s \text{ random rows of } \mathbf{I}_N$$

$$\mathcal{F}_N = \text{ FFT of size } N$$

$$\mathbf{D}_N = \text{ diagonal random } \pm 1$$

Kronecker FJLT mixing: $\mathbf{S}\left(\bigotimes_{\substack{\ell=d \\ \ell\neq k}}^{1} \mathcal{F}_n\mathbf{D}_n\right)\mathbf{Z}_k$

$$\left(\bigotimes_{\substack{\ell=d \\ \ell\neq k}}^{1} \mathcal{F}_n\mathbf{D}_n\right)\left(\bigodot_{\substack{\ell=d \\ \ell\neq k}}^{1} \mathbf{A}_\ell\right) = \bigodot_{\substack{\ell=d \\ \ell\neq k}}^{1} \mathcal{F}_n\mathbf{D}_n\mathbf{A}_\ell$$

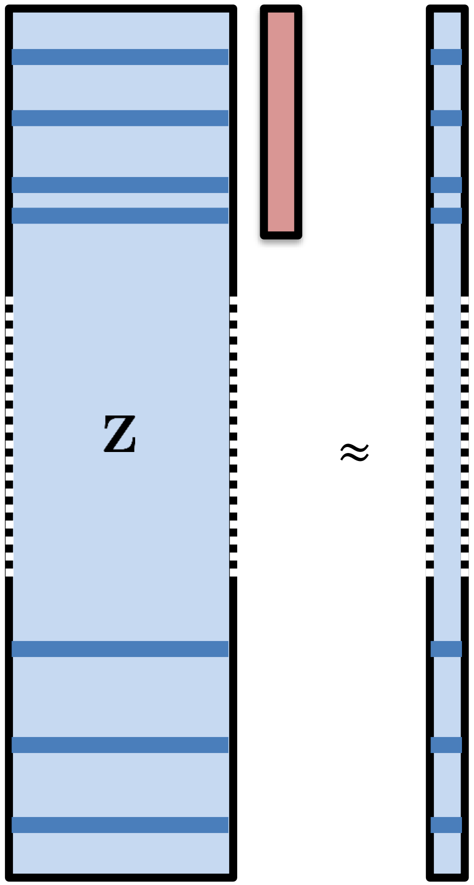
We can *mix first* and then sample – avoid every forming Z!

Is this still a JLT? *Yes (Jin-Kolda-Ward)!*

Complexity: $O(rn \log n + sr^2)$

Option 4: Kronecker FJLT

$$\min_{\mathbf{a}} \|\mathbf{Z}\mathbf{a} - \mathbf{x}\|^2$$



$N \times r$

Computational complexity: $O(Nr^2)$ versus $O(rn \log n + sr^2)$

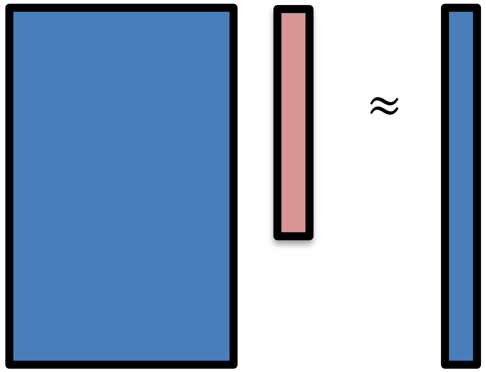
Sample Mixed Rows

$\mathbf{S} = s$ random rows of $\mathbf{I}_N$

$\mathcal{F}_n =$ FFT of size n

$\mathbf{D}_n =$ diagonal random ± 1

$$\min_{\mathbf{a}} \left\|\mathbf{S}\left(\otimes_{k=1}^{d-1}\mathcal{F}_n\mathbf{D}_n\right)\mathbf{Z}\mathbf{a} - \mathbf{S}\left(\otimes_{k=1}^{d-1}\mathcal{F}_n\mathbf{D}_n\right)\mathbf{x}\right\|^2$$



$s \times r$

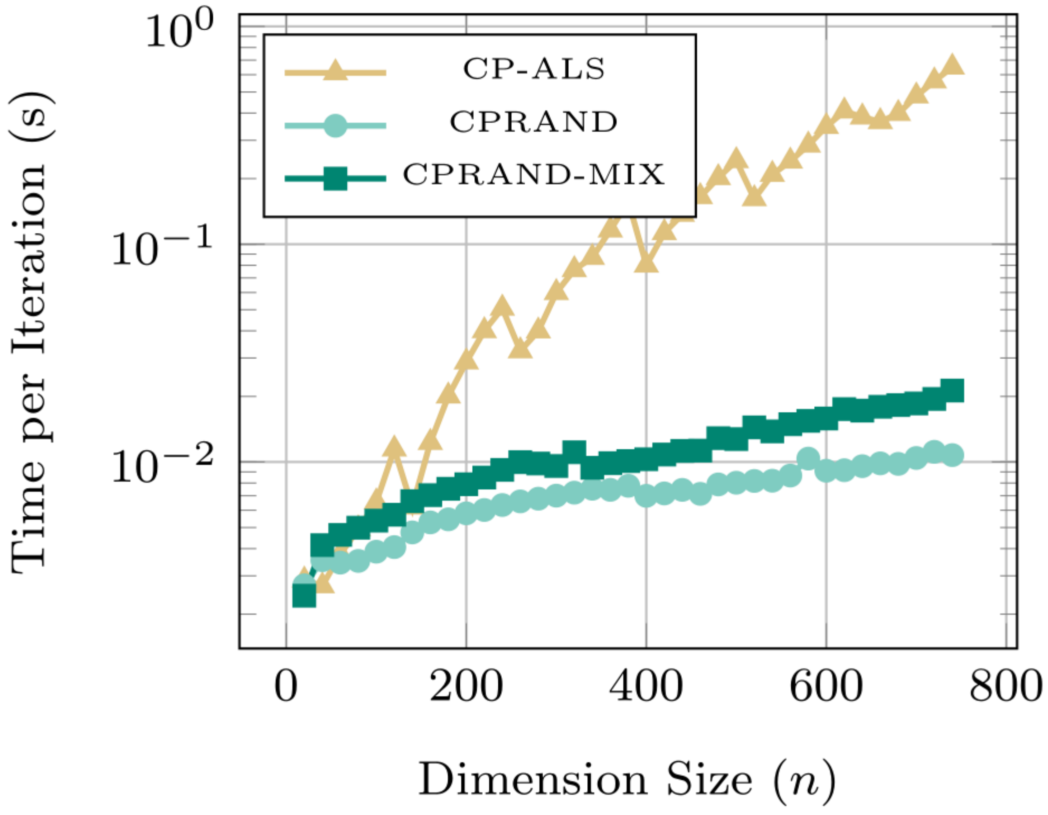
Avoid $O(Nr)$ cost to form $\mathbf{Z}$ by only computing the sampled rows!

Special effort needed to avoid cost of mixing right-hand side…

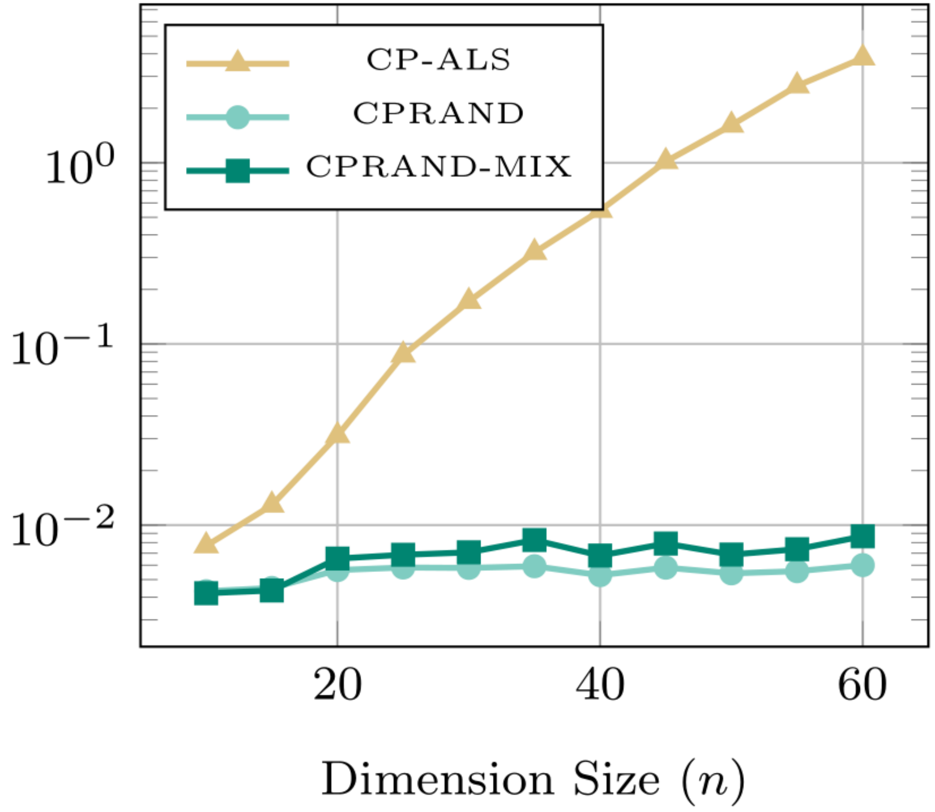
Randomized CP-ALS (CPRAND) Yields Speed-Ups as Problem Size Grows!

Per-iteration Timing Comparison with $r = 5$ (number of components) and $s = 90$ (number of samples)

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



5-way tensor of size $n \times n \times n \times n \times n$

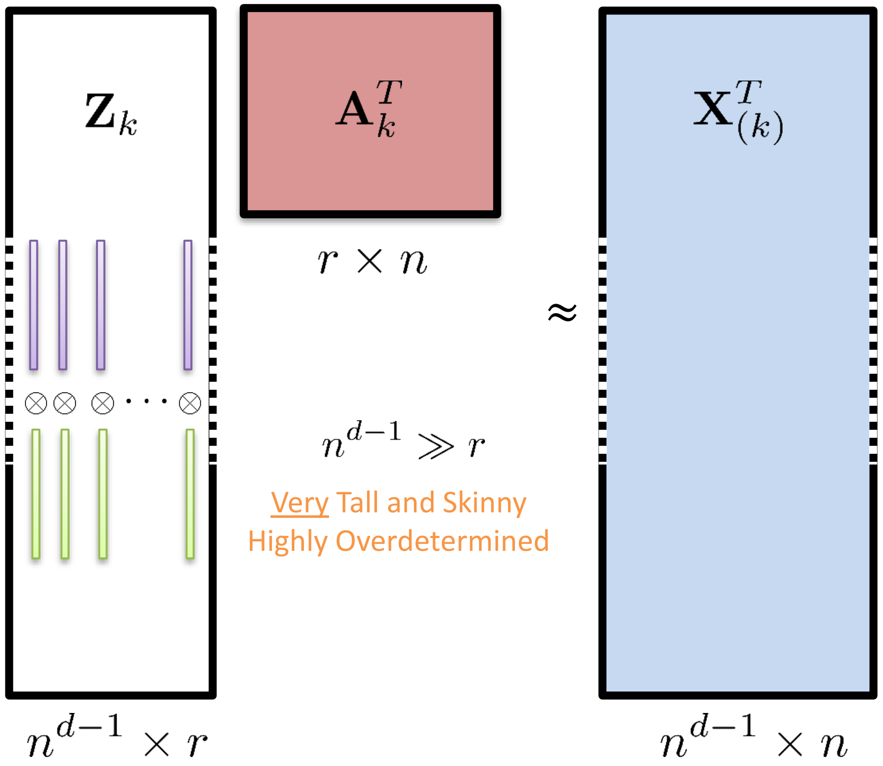


Note there is almost no change in number of iterations, so per-iteration speed-ups relevant

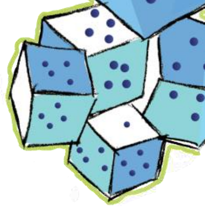
Matrix Sketching Only Worthwhile If Structure Exploited

$$\min_{\mathbf{A}_k} \quad \|\mathcal{X} - \mathcal{M}\|^2 = \|\mathbf{X}_{(k)} - \mathbf{A}_k \mathbf{Z}_k^T\|^2$$

$$\text{s.t.} \quad \mathbf{Z}_k = \bigodot_{\ell \neq k} \mathbf{A}_\ell$$



- CP-ALS is standard method for fitting tensor decomposition with tall + skinny least-squares methods at its heart
- Matrix sketching used *within* larger ALS algorithm – called many times!
- Naïve application of FJLT would be less efficient than direct solution
- Adapted principals for fast mixing to special Kronecker product structure – resulting in huge complexity reduction
- Proved "Kronecker FJLT" is a low-distortion embedding
- Working on leverage-score sampling as another alternative – requires clever crafting of sampling strategy

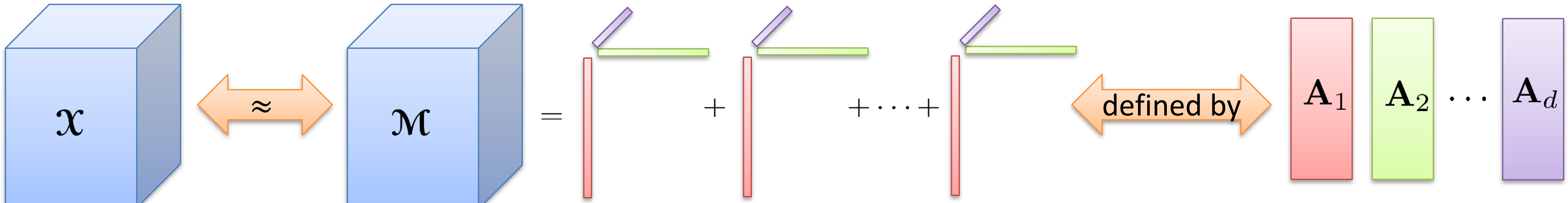


Stochastic Gradients for Tensor Decomposition

- D. Hong, T. G. Kolda, J. A. Duersch. **Generalized Canonical Polyadic Tensor Decomposition**. *SIAM Review*, in press, 2019
- T. G. Kolda, D. Hong. **Stochastic Gradients for Large-Scale Tensor Decomposition**. arXiv:1906.01687, 2019

Recall the Optimization Problem Uses Sum of Squares Error (SSE)

Data | Model | Sum of r Outer Product Tensors | Factor Matrices



$\mathcal{X} \in \mathbb{R}^{n \times n \times \cdots \times n}$

$x_i = x(i_1, i_2, \dots, i_d)$

$\mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_d]\!] \in \mathbb{R}^{n \times n \times \cdots \times n}$

$m_i = m(i_1, i_2, \dots, i_d) = \sum_{j=1}^{r} \prod_{k=1}^{d} a_k(i_k, j)$

$\mathbf{A}_k \in \mathbb{R}^{n \times r}$

$\text{rank}(\mathcal{M}) \leq r$

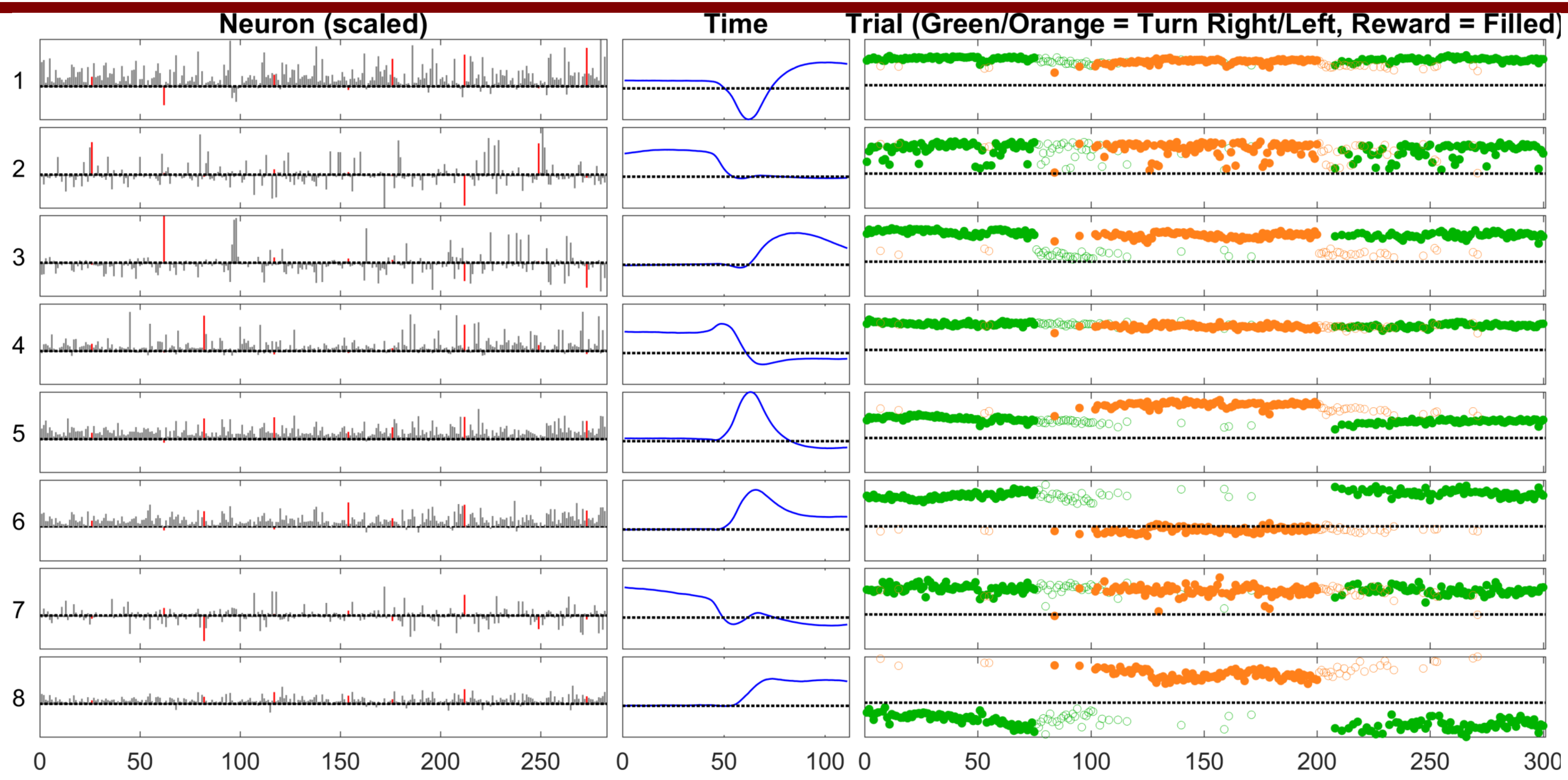
Standard CP

$$\min_{\mathbf{A}_1,\dots,\mathbf{A}_d} \quad \|\mathcal{X} - \mathcal{M}\|^2 \equiv \sum_{i=1}^{n^d} (x_i - m_i)^2$$

$$\text{s.t.} \quad \mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_d]\!]$$

$$\mathbf{A}_k \in \mathbb{R}^{n \times r} \text{ for } k = 1, \dots, d$$

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



Reward!

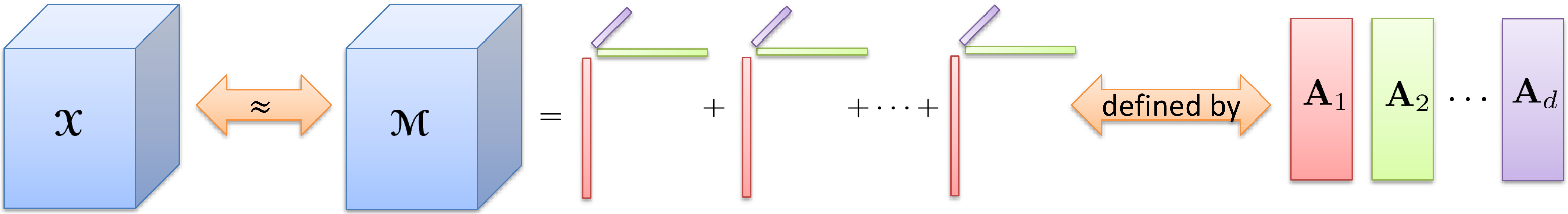
Turn left

Turn right

Turn

Generalized CP (GCP) Allows for Different Loss Functions

Data | Model | Sum of r Outer Product Tensors | Factor Matrices



$\mathcal{X} \in \mathbb{R}^{n \times n \times \cdots \times n}$

$x_i = x(i_1, i_2, \ldots, i_d)$

$\mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_d]\!] \in \mathbb{R}^{n \times n \times \cdots \times n}$

$m_i = m(i_1, i_2, \ldots, i_d) = \sum_{j=1}^{r} \prod_{k=1}^{d} a_k(i_k, j)$

$\mathbf{A}_k \in \mathbb{R}^{n \times r}$

$\text{rank}(\mathcal{M}) \leq r$

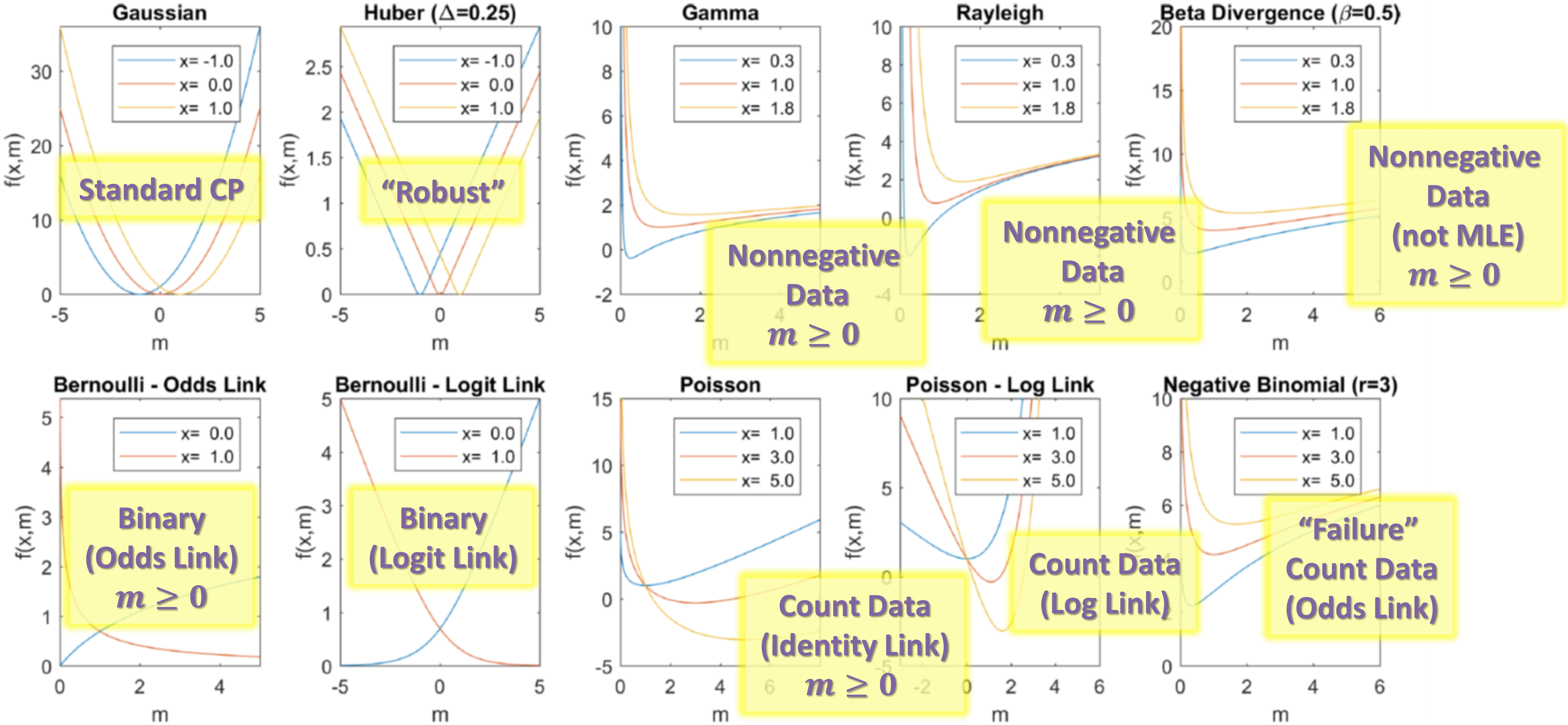
Generalized CP

$$\min_{\mathbf{A}_1, \ldots, \mathbf{A}_d} \quad F(\mathcal{X}, \mathcal{M}) = \sum_{i=1}^{n^d} f(x_i, m_i)$$

$$\text{s.t.} \quad \mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_d]\!]$$

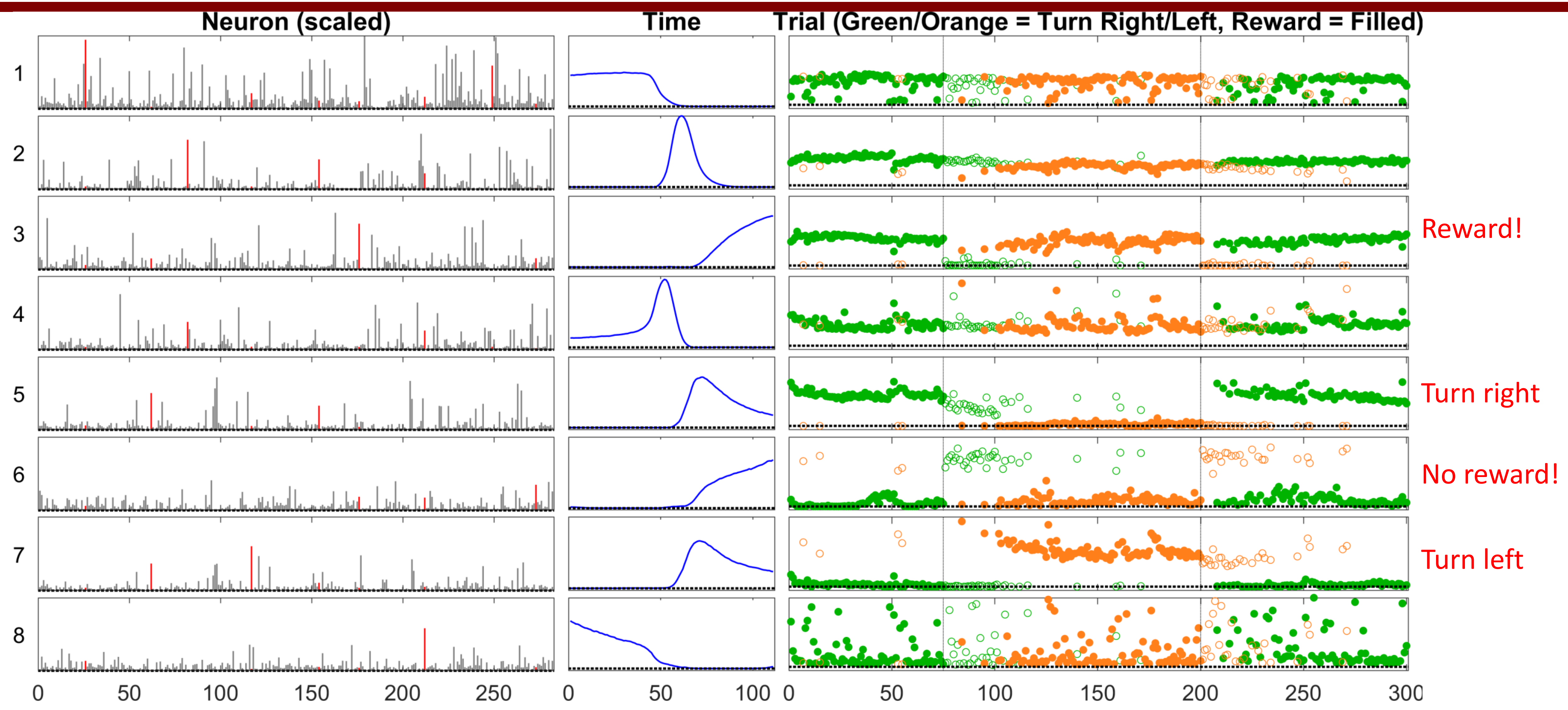
$$\mathbf{A}_k \in \mathbb{R}^{n \times r} \text{ for } k = 1, \ldots, d$$

Alternative Loss Functions Allow Binary, Count, Nonnegative Data



Welling & Webber, 2001; Cichocki & Phan, 2009; Chi & Kolda, 2009; Hong, Kolda, Duersch, SIAM Review, 2019

GCP Decomposition with Beta Divergence ($\beta = 0.5, f(x,m) = \sqrt{m} + x/\sqrt{m}$)

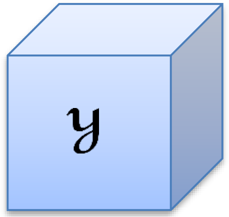


Gradient-based Optimization Can Be Used for Fitting the GCP Model

Generalized CP

$$\min_{\mathbf{A}_1,\dots,\mathbf{A}_d} \quad F(\mathcal{X},\mathcal{M}) = \sum_{i=1}^{n^d} f(x_i, m_i)$$
$$\text{s.t.} \quad \mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_d]\!]$$
$$\mathbf{A}_k \in \mathbb{R}^{n \times r} \text{ for } k = 1,\dots,d$$

Define: Elementwise partial gradient tensor, same size as data tensor = n^d



$$y_i = \frac{\partial f}{\partial m}(x_i, m_i)$$

Define: Khatri-Rao product in all modes but one of size $n^{d-1} \times r$

$$\mathbf{Z}_k = \bigodot_{\substack{\ell=d \\ \ell \neq k}}^{1} \mathbf{A}_\ell$$

Gradients computed via a sequence of matricized-tensor times Khatri-Rao product (MTTKRPs):

$$\mathbf{G}_k \equiv \frac{\partial F}{\partial \mathbf{A}_k} = \mathbf{Y}_{(k)} \mathbf{Z}_k \text{ for } k = 1,\dots,d$$

gradient for mode-k factor matrix

tensor unfolded in mode k into matrix

MTTKRP

MTTKRPs can be computed efficiently…

- Bader & Kolda, SISC, 2007 – Dense and sparse
- Phan, Tichavsky, Cichocki, 2013 – Sequence
- Smith et al., IPDPS 2015 – Sparse
- Kaya & Ucar, SC 2015 – Sparse
- Li et al., IPDPS 2017 – Sparse
- Hayashi et al., 2017 – Dense
- Ballard, Knight, Rouse, 2017 – Dense

Stochastic Gradient Descent (SGD) for GCP Chooses Sparse Stochastic Y-Tensor

Sandia National Laboratories

Generalized CP

$$\min_{\mathbf{A}_1,\ldots,\mathbf{A}_d} \quad F(\mathcal{X},\mathcal{M}) = \sum_{i=1}^{n^d} f(x_i, m_i)$$

$$\text{s.t.} \quad \mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_d]\!]$$

$$\mathbf{A}_k \in \mathbb{R}^{n \times r} \text{ for } k = 1, \ldots, d$$

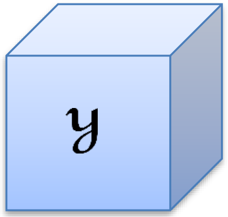
Mode-k unfolding:

$$\mathbf{Y}_{(k)} \in \mathbb{R}^{n \times n^{(d-1)}}$$

Khatri-Rao product of all factor matrices but one:

$$\mathbf{Z}_k = \bigodot_{\substack{\ell=d \\ \ell \neq k}}^{1} \mathbf{A}_\ell \in \mathbb{R}^{n^{(d-1)} \times r}$$

Standard gradient

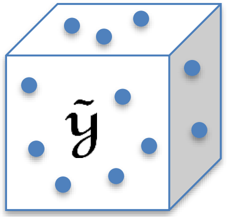


$$\mathbf{G}_k = \mathbf{Y}_{(k)} \mathbf{Z}_k$$

Cost: $O(rn^d)$ flops

$$y_i = \frac{\partial f}{\partial m}(x_i, m_i)$$

Stochastic gradient



$$\tilde{\mathbf{G}}_k = \tilde{\mathbf{Y}}_{(k)} \mathbf{Z}_k$$

Cost: $O(rs)$ flops

Choose stochastic *sparse* Y-tensor

$$\mathbb{E}[\tilde{\mathcal{Y}}] = \mathcal{Y}$$

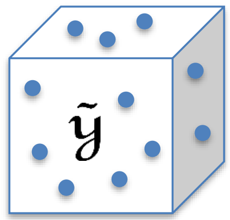
such that

$$\text{nnz}(\tilde{\mathcal{Y}}) \leq s \ll n^d$$

By linearity of expectation: $\mathbb{E}[\tilde{\mathbf{G}}_k] = \mathbf{G}_k$

Uniform Sampling with Appropriate Weights Yields GCP Stochastic Gradient

$$y_i = \frac{\partial f}{\partial m}(x_i, m_i)$$



$$\mathbb{E}[\tilde{\mathcal{Y}}] = \mathcal{Y}$$

$$\text{nnz}(\tilde{\mathcal{Y}}) \leq s \ll n^d$$

Sample $s \ll n^d$ random tensor entries (with replacement)

$$\tilde{s}_i = \#\text{ times } i \text{ sampled}$$

$$\tilde{y}_i = \tilde{s}_i \cdot \frac{n^d}{s} \cdot y_i$$

Choosing s, the number of sampled elements…

- Choose $s = O(n)$
- Gradient = $O(rs) = O(rn)$ versus $O(rn^d)$

Downside…

- If data tensor is sparse, few entries corresponding to nonzeros will be chosen

Theory

Claim: $\mathbb{E}[\tilde{\mathcal{Y}}] = \mathcal{Y}$

Proof: $\mathbb{E}[\tilde{s}_i] = \dfrac{s}{n^d}$

$$\mathbb{E}[\tilde{y}_i] = \mathbb{E}[\tilde{s}_i] \cdot \frac{n^d}{s} \cdot y_i = y_i$$

Nonzeros Needed to Reduce Variance in Stochastic Gradient

Biased sampling toward *functionals* with higher Lipschitz smoothness constants reduces variance in stochastic gradient (Needell, Srebro, & Ward, 2013)

For tensors, *functionals equate to tensor entries, i.e.,* $f_i = f(x_i, m_i)$

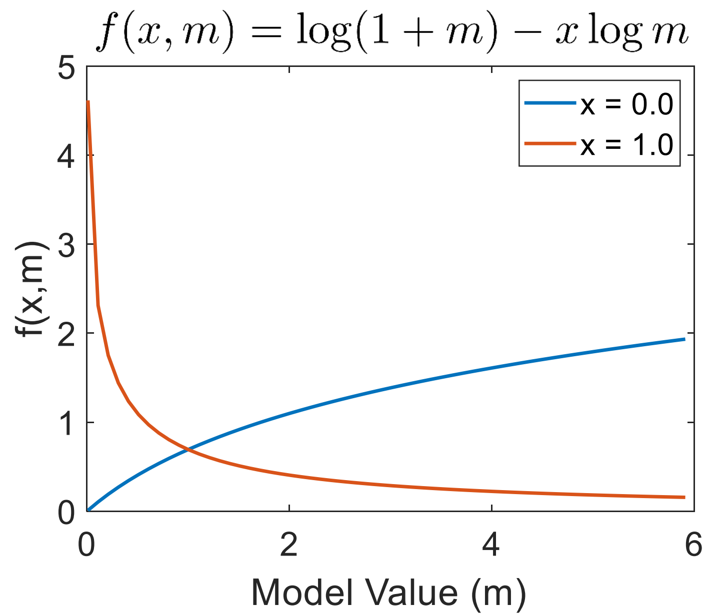
Consider Bernoulli with odds link: $f(x,m) = \log(1+m) - x \log m$

$$\frac{\partial f}{\partial m}(0,m) = \frac{1}{m+1} \quad \Rightarrow \quad L \leq 1$$

$$\frac{\partial f}{\partial m}(1,m) = \frac{-1}{m^2+m} \quad \Rightarrow \quad L \text{ unbounded as } m \downarrow 0$$

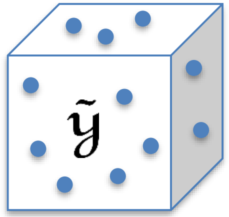
Need to bias sampling to select more nonzeros in sparse tensors

Sparse Binary (0/1) Tensor



Stratified Zero/Nonzero Sampling

$$y_i = \frac{\partial f}{\partial m}(x_i, m_i)$$



$$\mathbb{E}[\tilde{\mathcal{Y}}] = \mathcal{Y}$$

$$\mathrm{nnz}(\tilde{\mathcal{Y}}) \leq s \ll n^d$$

Sample p nonzeros and q zeros.

$$\tilde{p}_i = \#\text{ times nonzero } i \text{ sampled} \qquad \eta = \#\text{ nonzeros}$$

$$\tilde{q}_i = \#\text{ times zero } i \text{ sampled} \qquad \zeta = \#\text{ zeros}$$

$$\tilde{y}_i = \left(\tilde{p}_i \cdot \frac{\eta}{p} + \tilde{q}_i \cdot \frac{\zeta}{q}\right) \cdot y_i \qquad y_i = \frac{\partial f}{\partial m}(x_i, m_i)$$

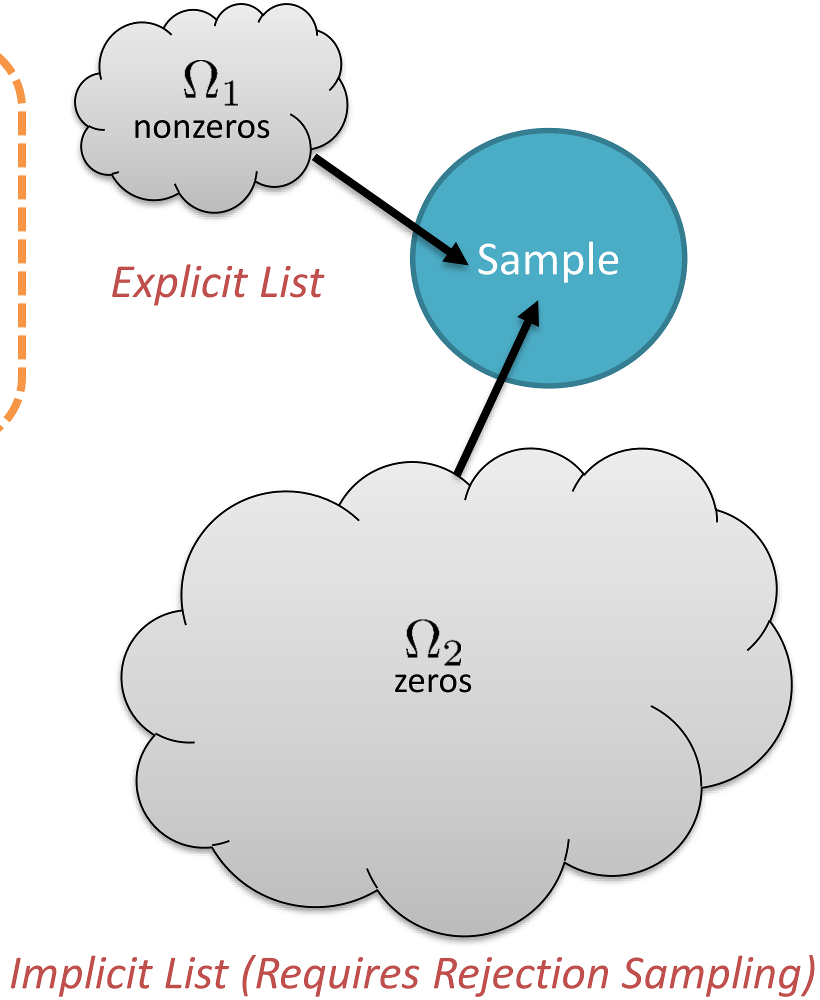
Theory

Claim: $\mathbb{E}[\tilde{\mathcal{Y}}] = \mathcal{Y}$

Proof: $\mathbb{E}[\tilde{p}_i] = \dfrac{p}{\eta}, \; \mathbb{E}[\tilde{q}_i] = \dfrac{q}{\zeta}$

$$x_i = 1 \Rightarrow \mathbb{E}[\tilde{y}_i] = \mathbb{E}[\tilde{p}_i] \cdot \frac{\eta}{p} \cdot y_i = y_i$$

$$x_i = 0 \Rightarrow \mathbb{E}[\tilde{y}_i] = \mathbb{E}[\tilde{q}_i] \cdot \frac{\zeta}{q} \cdot y_i = y_i$$



Semi-Stratified Zero/Nonzero Sampling

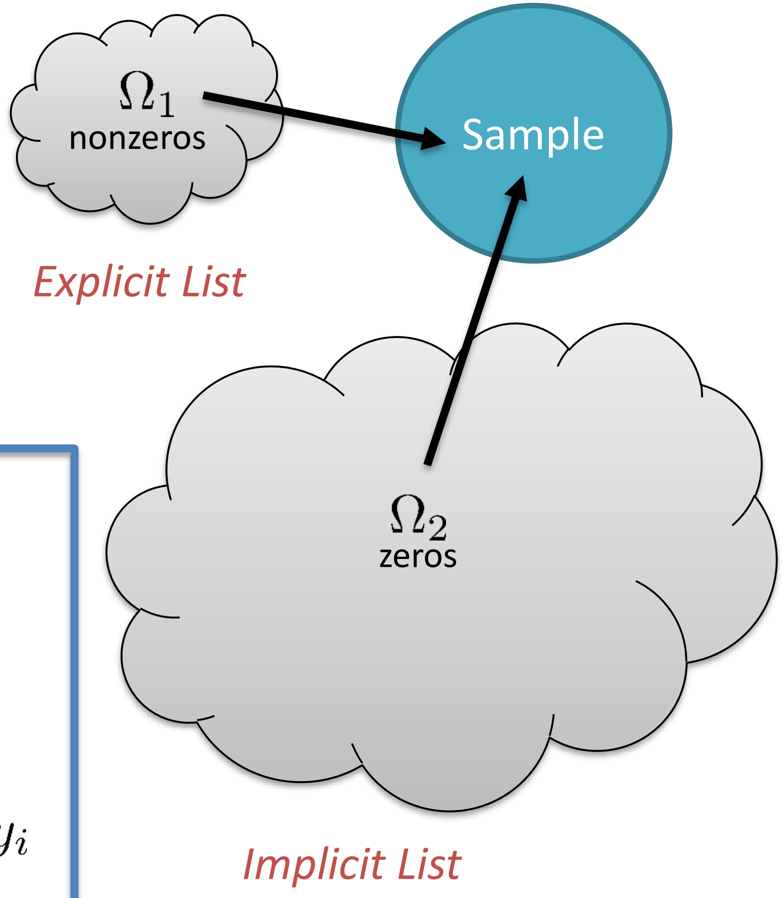
Idea: Sample "assumed zeros" from all indices and *correct* in nonzero samples.

Sample p nonzeros and q assumed zeros.

$\tilde{p}_i = \#$ times nonzero i sampled $\qquad \eta = \#$ nonzeros

$\tilde{q}_i = \#$ times "zero" i sampled $\qquad \zeta = \#$ zeros

$$\tilde{y}_i = \tilde{p}_i \cdot \frac{\eta}{p} \cdot (y_i - c_i) + \tilde{q}_i \cdot \frac{(\eta + \zeta)}{q} \cdot c_i \text{ with } c_i \equiv \frac{\partial f}{\partial m}(0, m_i)$$



Theory

Claim: $\mathbb{E}[\tilde{\mathcal{Y}}] = \mathcal{Y}$

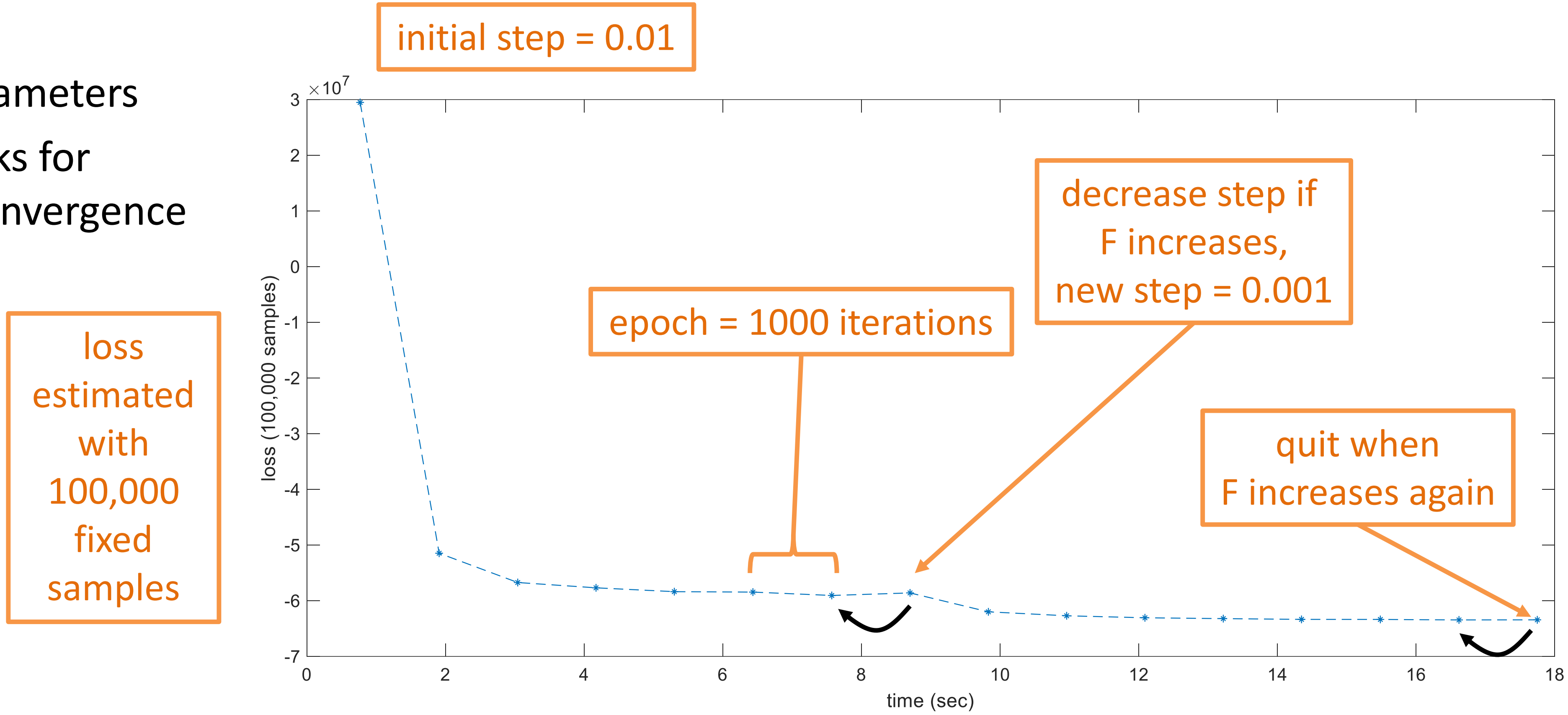
Proof: $\mathbb{E}[\tilde{p}_i] = \dfrac{p}{\eta}, \; \mathbb{E}[\tilde{q}_i] = \dfrac{q}{(\zeta + \eta)}$

$$x_i = 0 \Rightarrow \mathbb{E}[\tilde{y}_i] = \mathbb{E}[\tilde{q}_i] \cdot \frac{(\eta + \zeta)}{q} \cdot y_i = y_i$$

$$x_i = 1 \Rightarrow \mathbb{E}[\tilde{y}_i] = \mathbb{E}[\tilde{p}_i] \cdot \frac{\eta}{p} \cdot (y_i - c_i) + \mathbb{E}[\tilde{q}_i] \cdot \frac{\eta + \zeta}{q} \cdot c_i = y_i$$

GCP with Stochastic Optimization

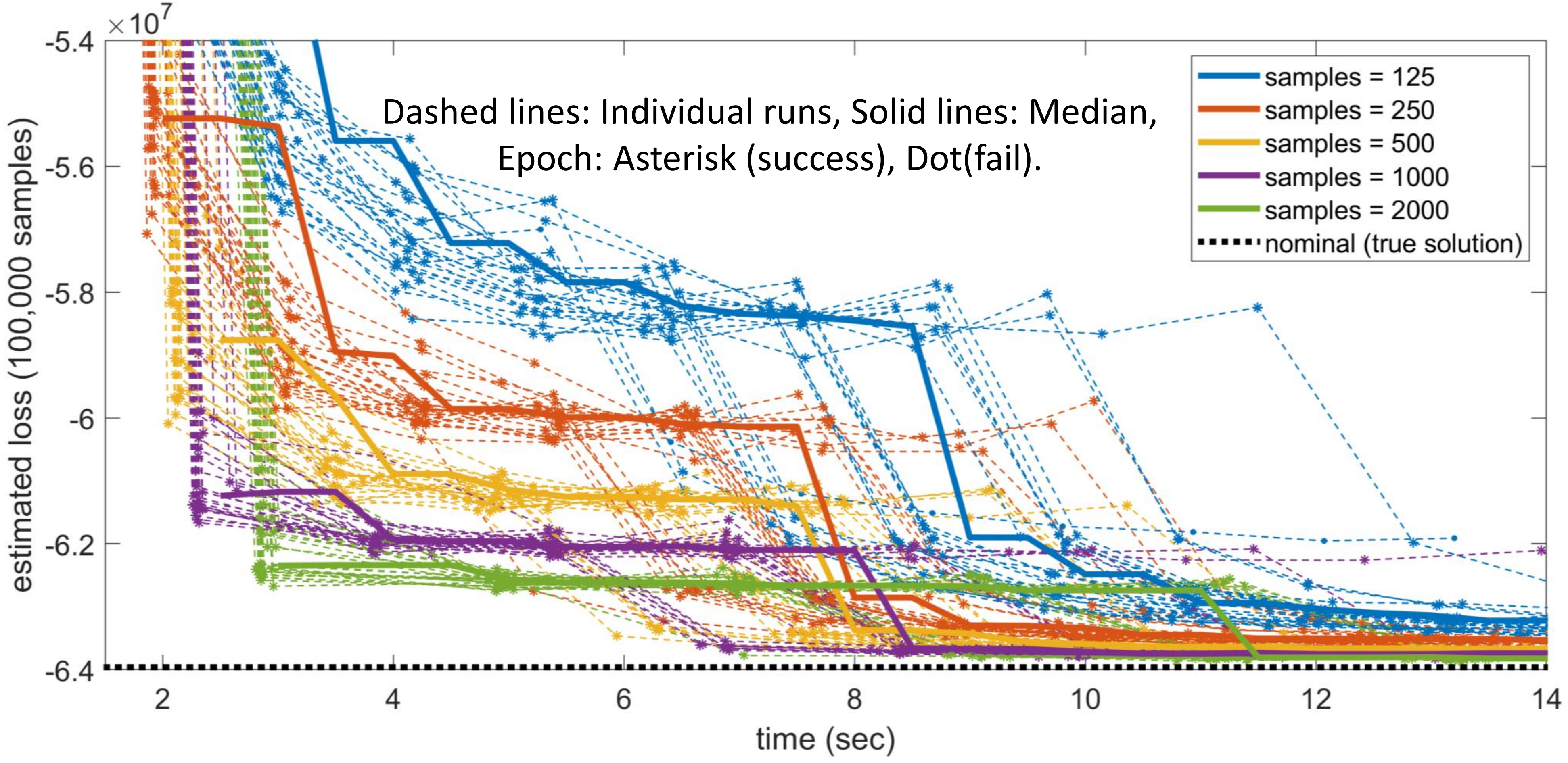
- Using Adam (Kingma & Ba, 2015)
  - Default parameters
  - Some tweaks for checking convergence



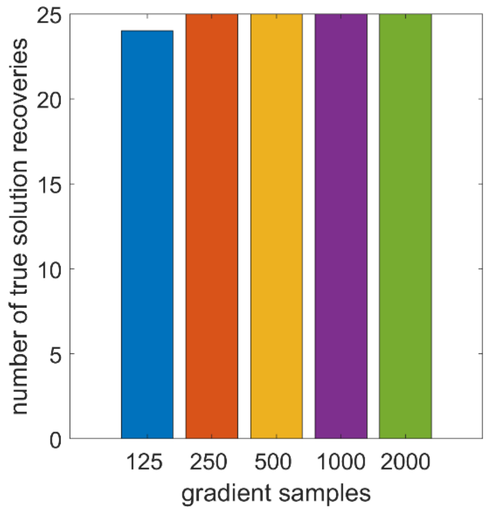
Roughly O(n) Samples Needed Per Stochastic Gradient

$200 \times 150 \times 100 \times 50$ Tensor (150M entries) with rank $r = 5$. Gamma loss: $f(x, m) = \frac{x}{m} + \log m$.

Running Adam with 25 random starts and varying numbers of samples.



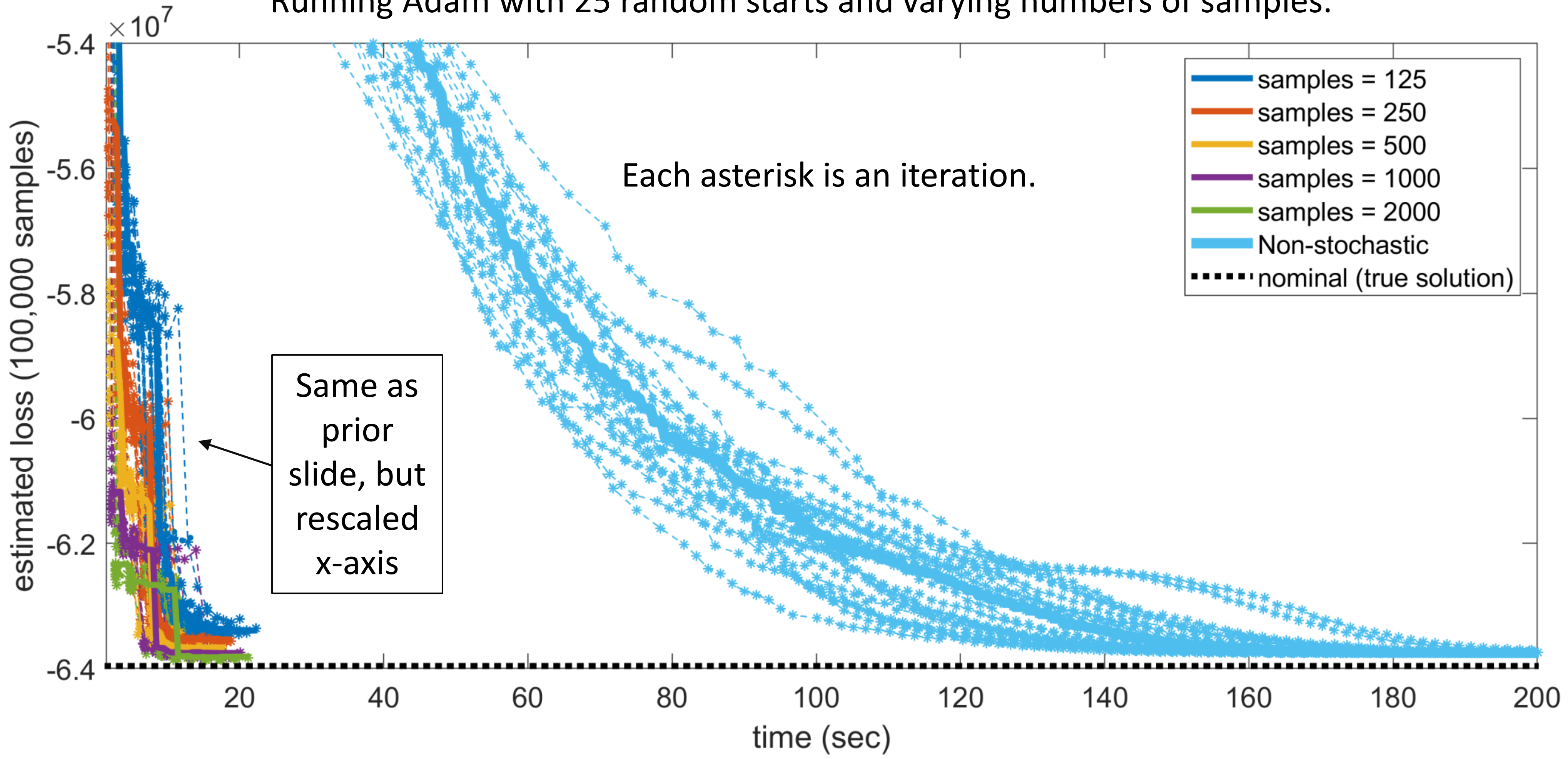
Recovery of Factor Matrices



Zooming Out: Stochastic Much Faster Than Non-Stochastic

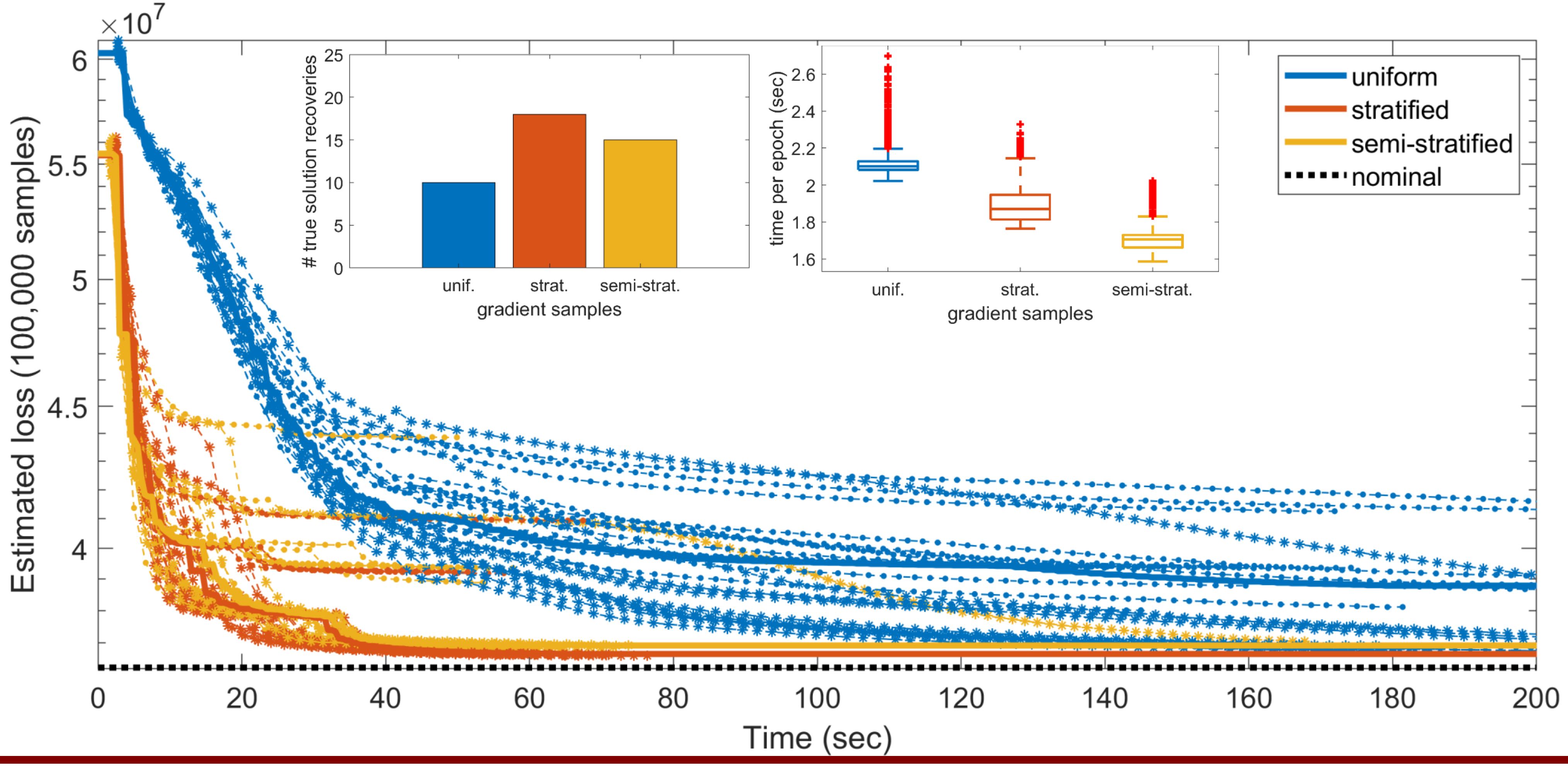
$200 \times 150 \times 100 \times 50$ Tensor (150M entries) with rank $r = 5$. Gamma loss: $f(x, m) = \frac{x}{m} + \log m$.

Running Adam with 25 random starts and varying numbers of samples.



Uniform Sampling is Worse than Stratified for Sparse Tensors

$400 \times 300 \times 200 \times 100$ Tensor (2.4B entries, 9M nonzeros, 0.4% dense) with rank $r = 5$.
Bernoulli loss: $f(x, m) = \log(m + 1) - x \log m$. Using $s = 1000$ samples, evenly divided for stratified and semi-stratified.

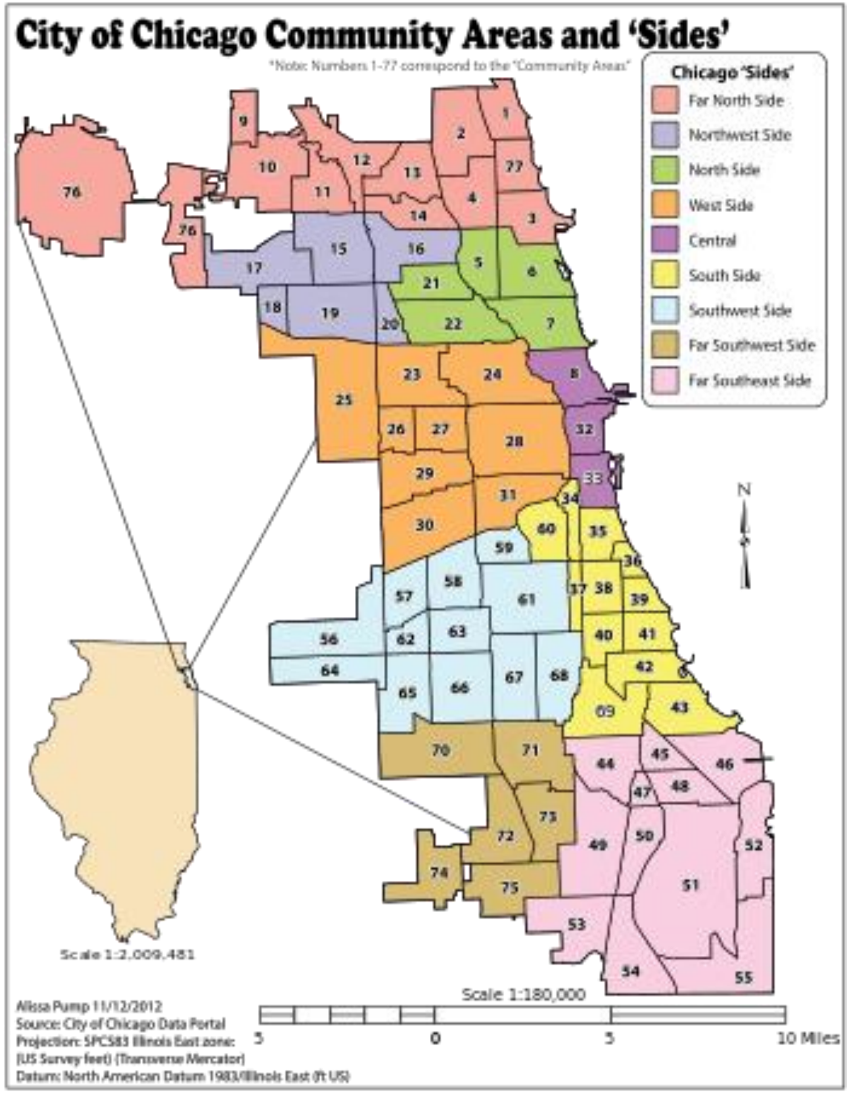


Chicago Crime Data

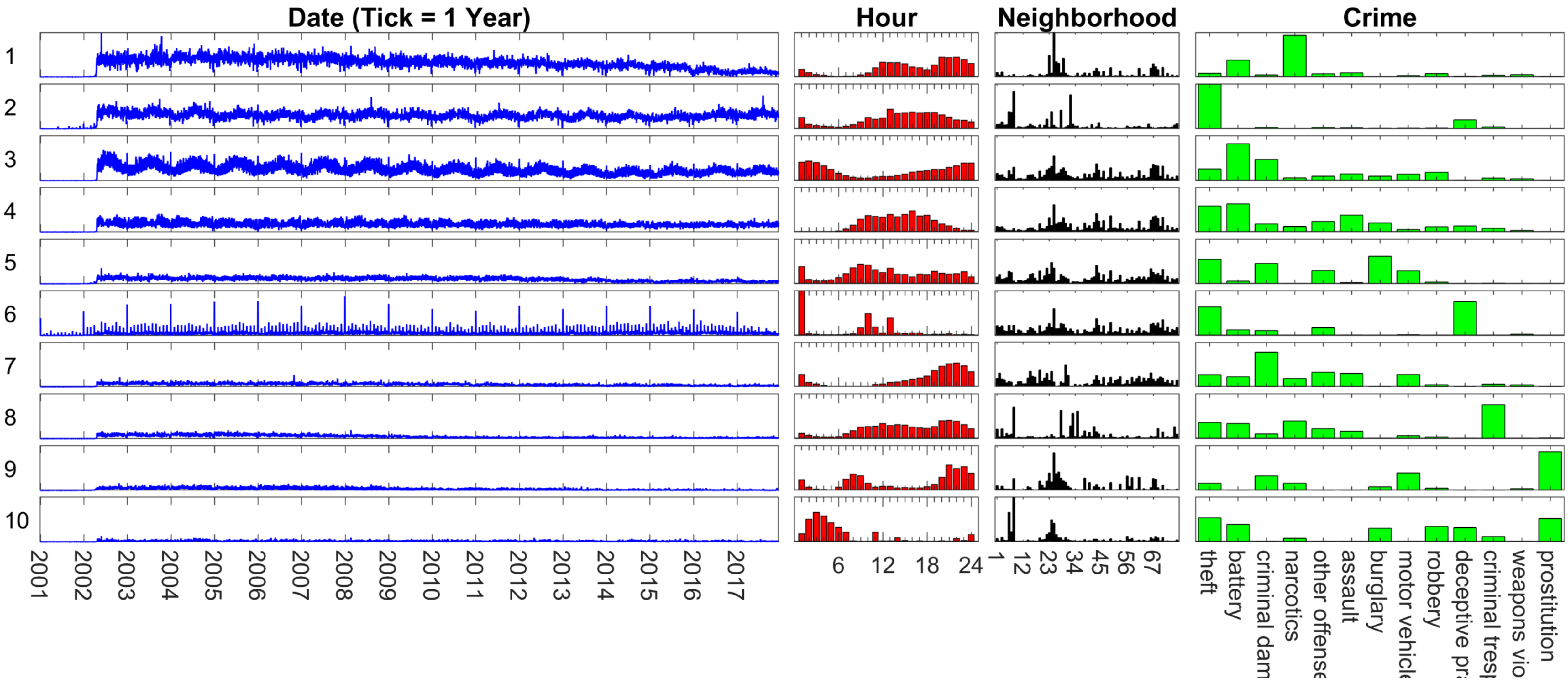
- 4-way count tensor
  - 6,186 Days
  - 24 Hours of the Day
  - 77 Community Areas
  - 32 Crime Types
- Non-zeros: 5,330,673
  - 0.21GB for sparse tensor
- Distribution of entries
  - 0: 98.54%
  - 1: 1.33%
  - ≥ 2: 0.12%
- Obtained from FROSTT (http://frostt.io/tensors/chicago-crime/)
- Data originally from Chicago Data Portal (https://data.cityofchicago.org/Public-Safety/Crimes-2001-to-present/ijzp-q8t2)

GCP-Count
Rank = 10
Samples $s = 6{,}319$

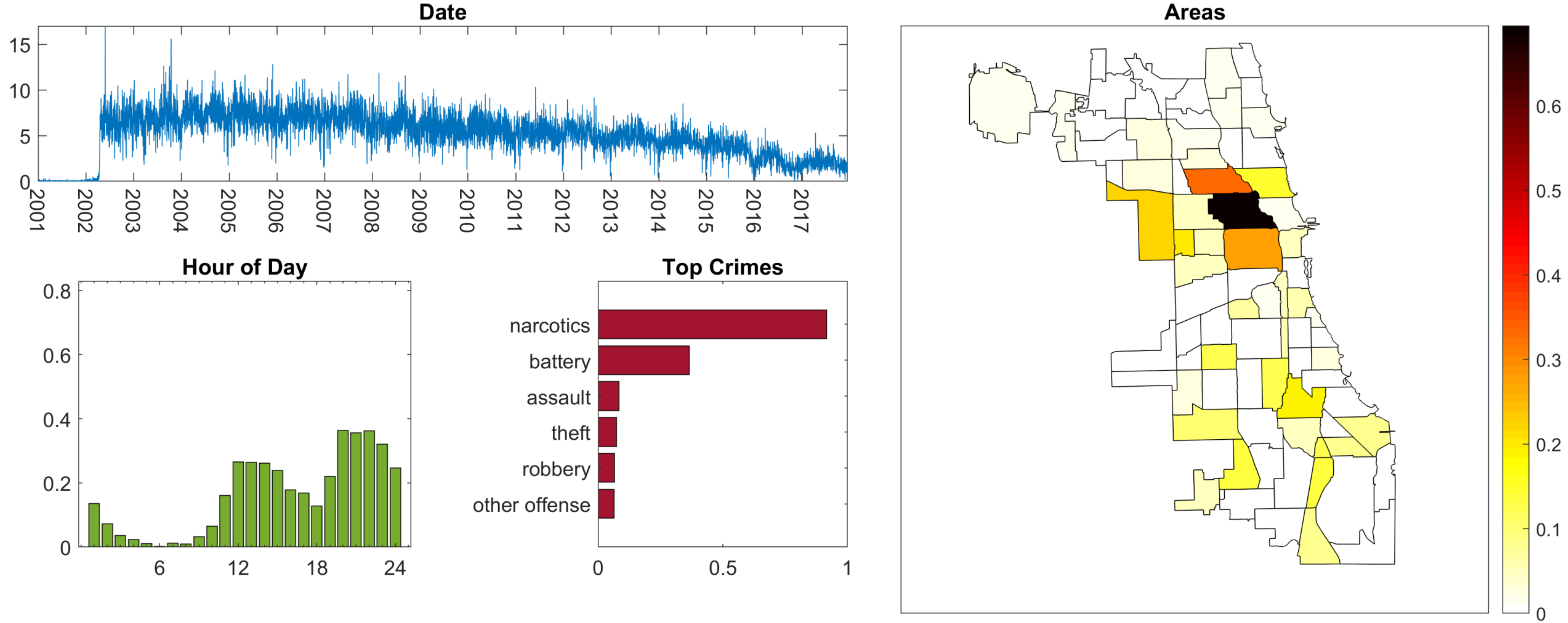
$$f(x,m) = m - x \log m$$



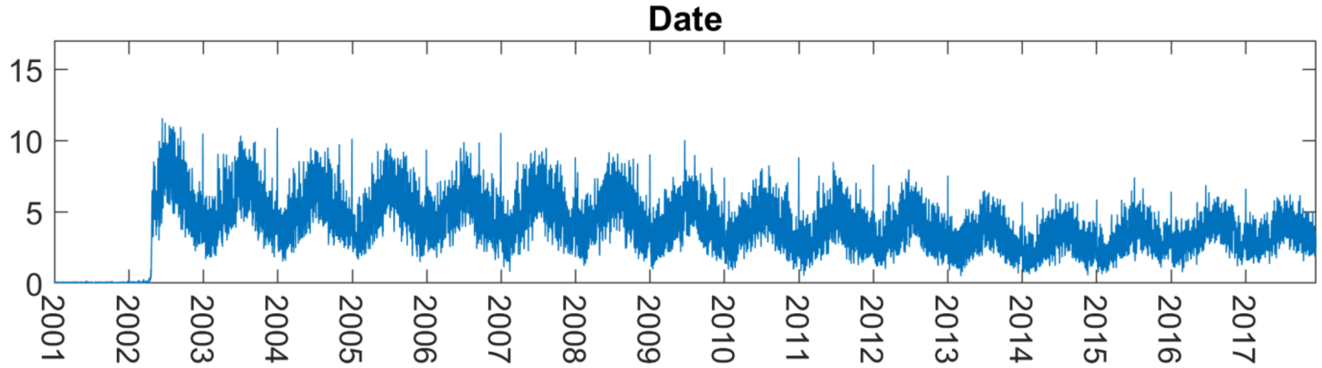
Application to Sparse Crime Binary Tensor (Semi-stratified results)



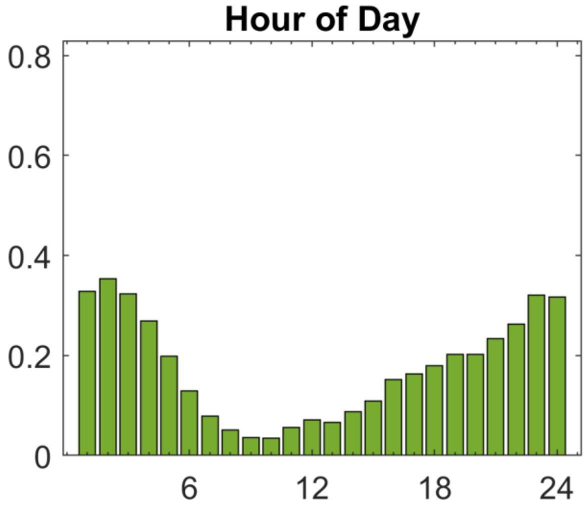
Component #1


Date
15
10
5
0
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
Hour of Day
0.8
0.6
0.4
0.2
0
6
12
18
24
Top Crimes
narcotics
battery
assault
theft
robbery
other offense
0
0.5
1
Areas
0.6
0.5
0.4
0.3
0.2
0.1
0

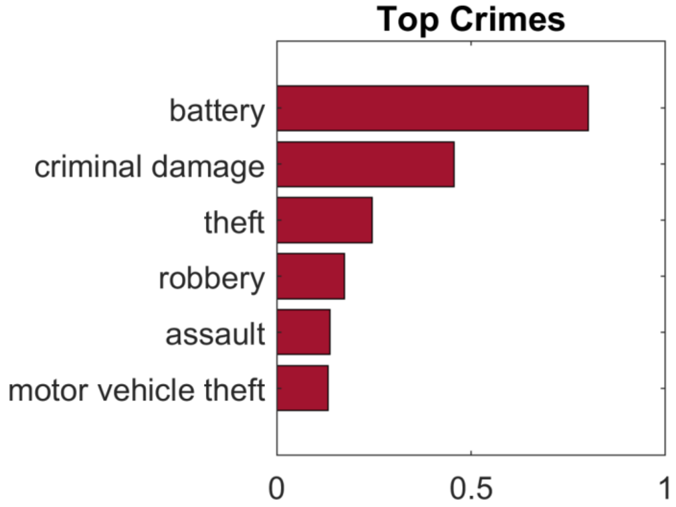
Component #3



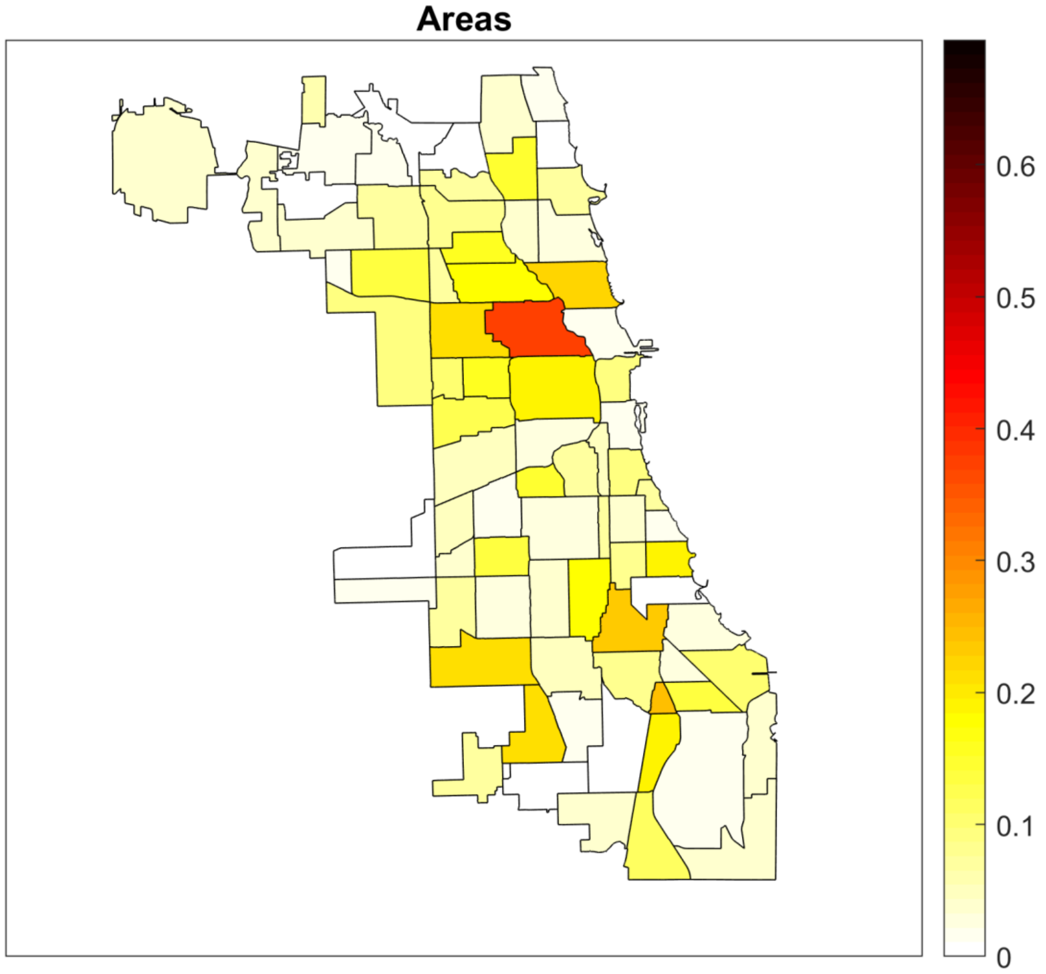
Date
15
10
5
0
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017



Hour of Day
0.8
0.6
0.4
0.2
0
6
12
18
24

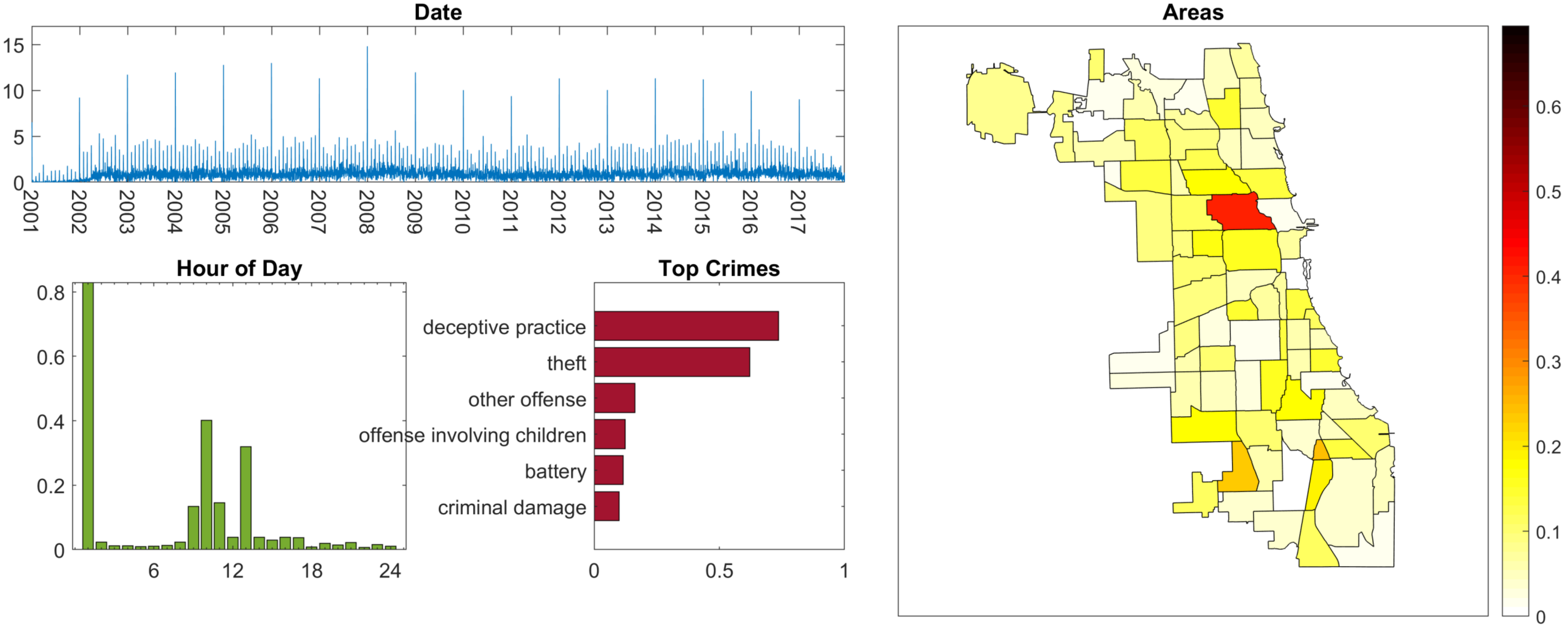


Top Crimes
battery
criminal damage
theft
robbery
assault
motor vehicle theft
0
0.5
1



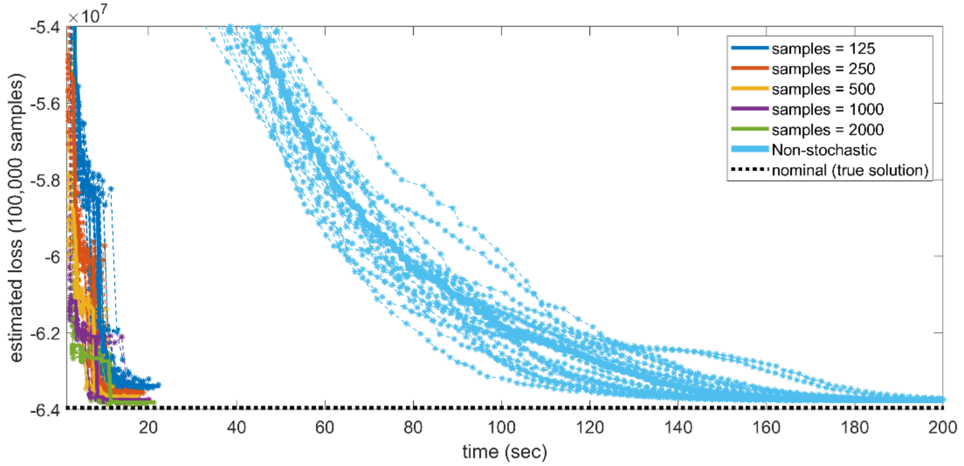
Areas
0.6
0.5
0.4
0.3
0.2
0.1
0

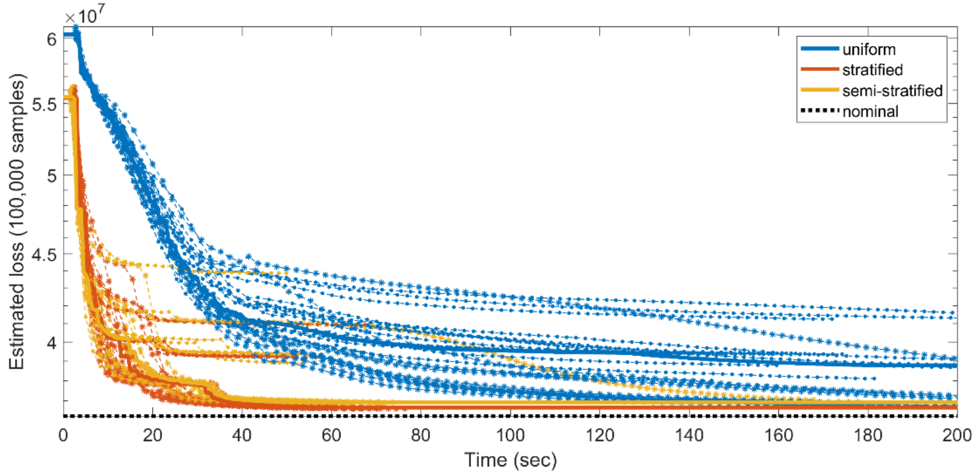
Component #6



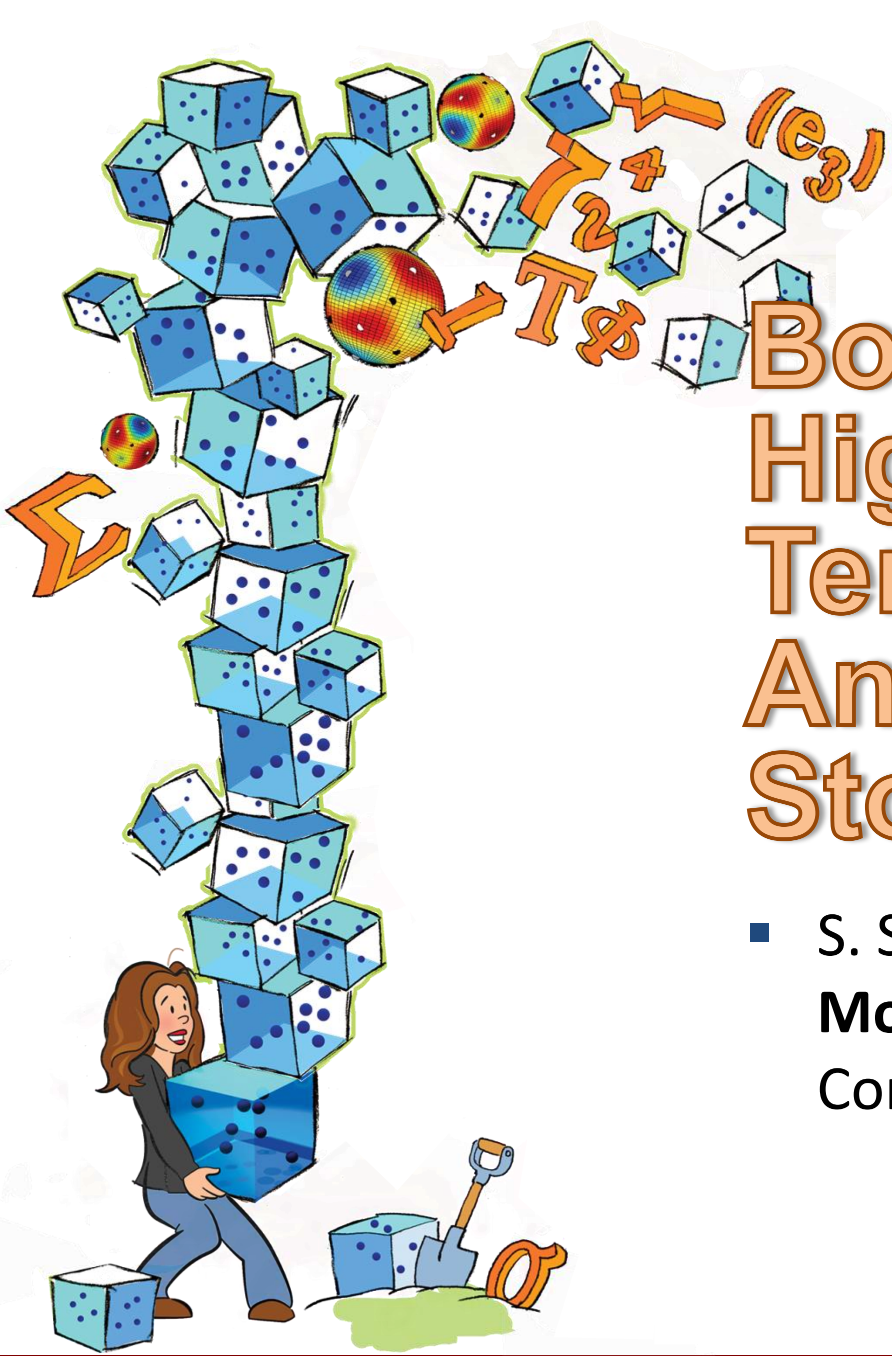
Date
15
10
5
0
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
Hour of Day
0.8
0.6
0.4
0.2
0
6
12
18
24
Top Crimes
deceptive practice
theft
other offense
offense involving children
battery
criminal damage
0
0.5
1
Areas
0.6
0.5
0.4
0.3
0.2
0.1
0

Stochastic Gradients Enables Significant Speed-Ups, But Need Smart Sampling





- GCP is tensor decomposition with modified objective function
- Stochastic version significantly faster
- Stratified sampling important for sparse problems
- Specialized semi-stratified yields greater computational efficiency
- Very few samples needed per iteration
- Stochastic methods (like Adam) need more robust foundations



Bonus Material: Higher-order Moments, Tensor Decomposition, and Another Way of Doing Stochastic Gradients

- S. Sherman, T. G. Kolda, **Estimating Higher-Order Moments Using Symmetric Tensor Decomposition**. Coming soon, 2019.

Empirical Higher-order Moments Measure Higher-Order Interactions

Let $\mathbf{v}_\ell \in \mathbb{R}^n$ for $i = 1, \ldots, p$ be the observations of the random variable V

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

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

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

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

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

Applications: Gaussian mixture models (GMMs), skewness/kurtosis estimation, moment matching, detecting outliers, etc.

Low-rank Symmetric Tensor Decomposition for Moment Tensors Exploits Structure

Observation Tensor

$$\mathbf{V} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_p \end{bmatrix} \in \mathbb{R}^{n \times p}$$

Empirical dth-order Moment Tensor

$$\mathcal{X} \propto \sum_{\ell=1}^{p} \mathbf{v}_\ell^{\otimes d} \in \mathbb{R}^{n^d}$$

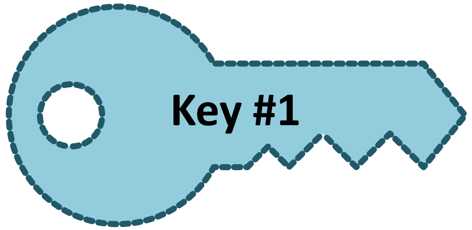
Symmetric CP Decomposition with Rank $r \ll p$

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

Storage/computation of O(n^d) may be intractable

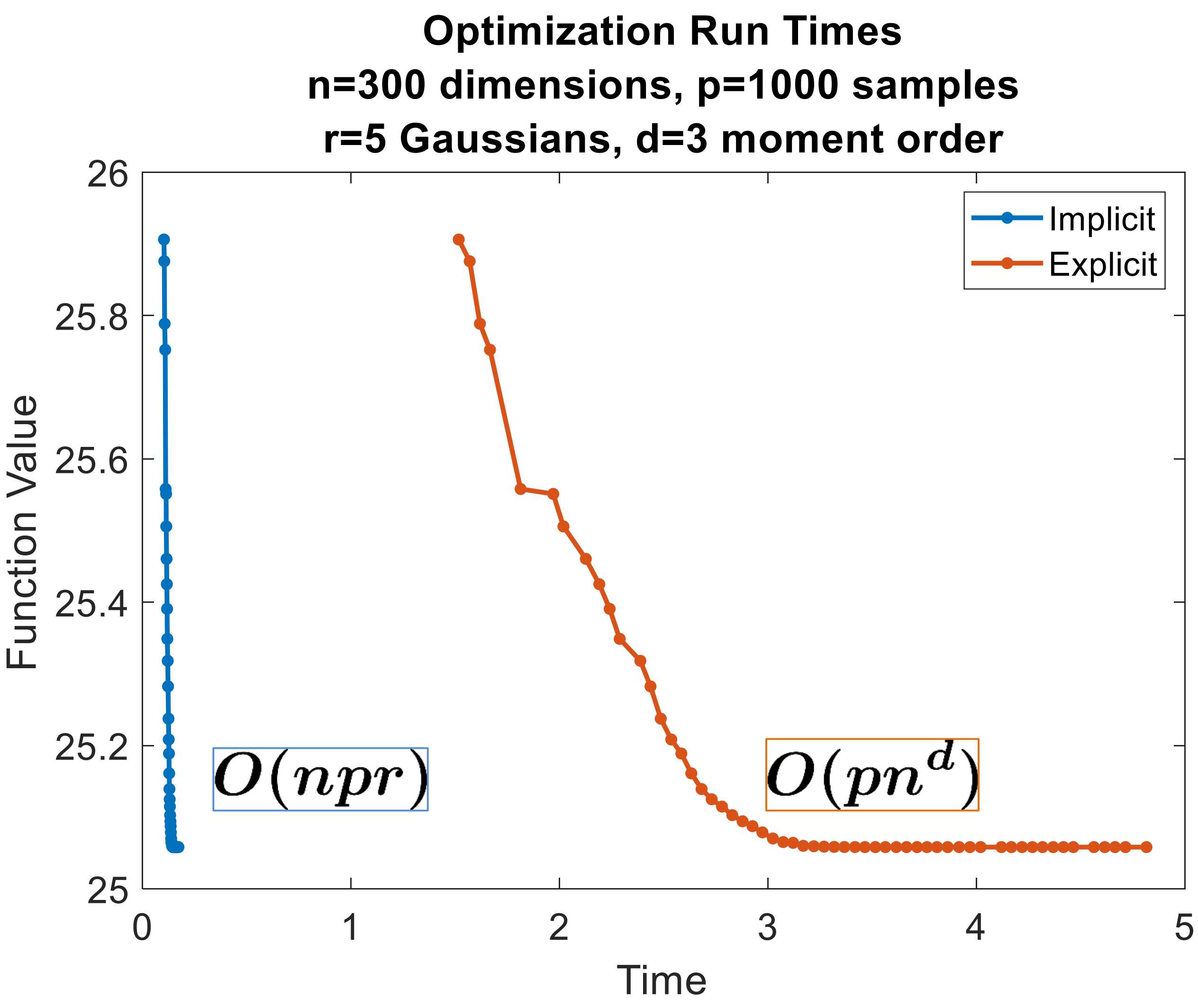
n = 2000, d = 3 ⇒ storage = 64 GB

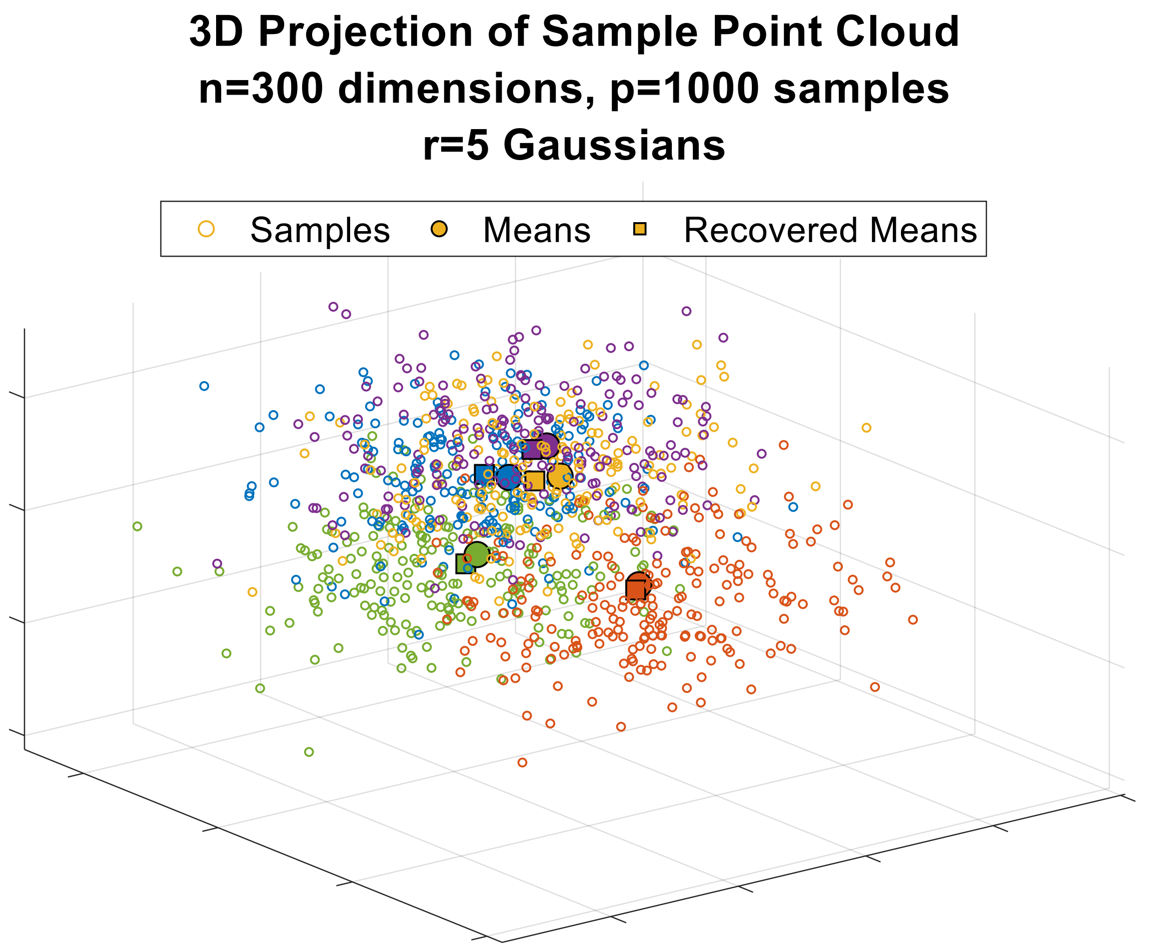
n = 500, d = 4 ⇒ storage = 500 GB



Avoid forming moment tensor explicitly, reducing work from O(pn^d) to O(pnr)

Implicit Method Much Faster For Gaussian Mixture Model Mean Identification





For Large Number of Observations (p), Use Stochastic Moment Tensor

Sandia National Laboratories

Observation Tensor

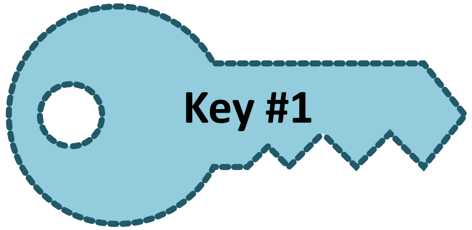
$$\tilde{\mathbf{V}} = \begin{bmatrix} \tilde{\mathbf{v}}_1 & \tilde{\mathbf{v}}_2 & \cdots & \tilde{\mathbf{v}}_s \end{bmatrix} \in \mathbb{R}^{n \times s}$$

Empirical dth-order Moment Tensor

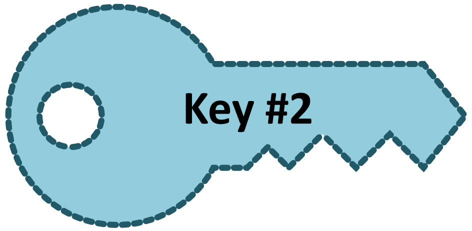
$$\tilde{\mathfrak{X}} \propto \sum_{\ell=1}^{s} \tilde{\mathbf{v}}_\ell^{\otimes d} \in \mathbb{R}^{n^d}$$

Symmetric CP Decomposition with Rank $r \ll p$

$$\hat{\mathfrak{X}} = \sum_{j=1}^{r} \mathbf{a}_j^{\otimes d} \in \mathbb{R}^{n^d}$$

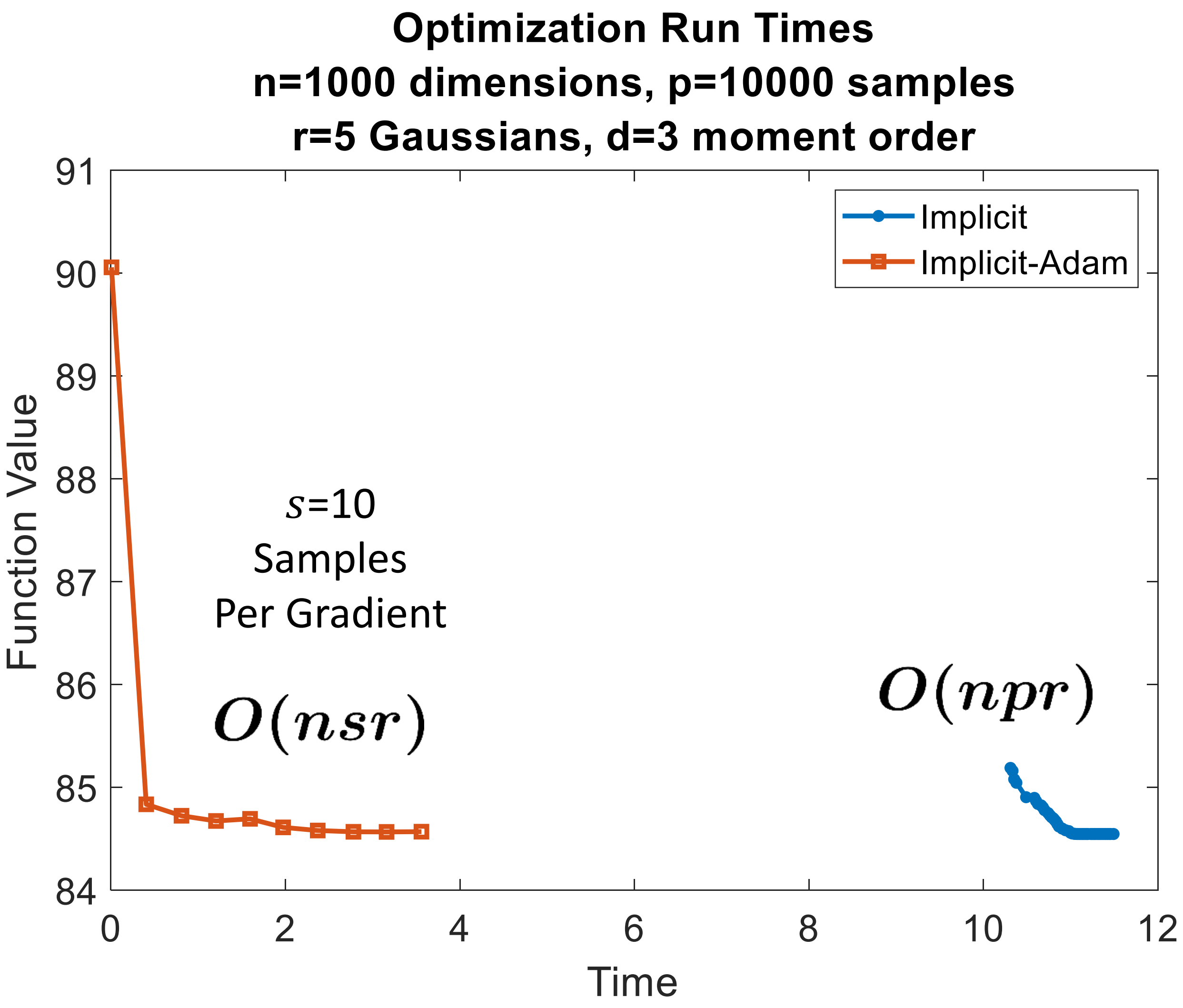


Avoid forming moment tensor explicitly, reducing work from O(pn^d) to O(pnr)

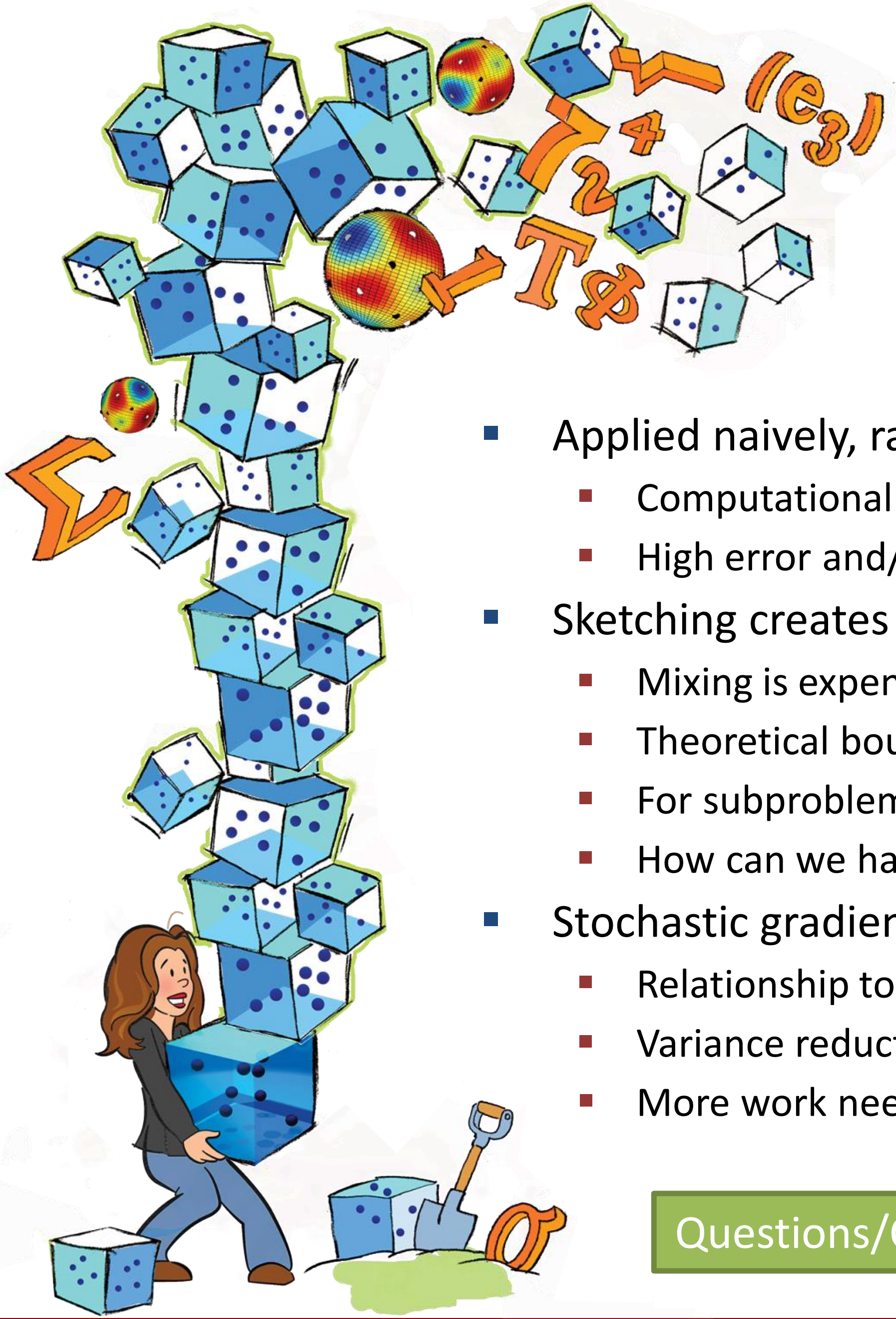


Use stochastic moment tensor, reducing work from O(pnr) to O(snr) with $s \ll p$

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



- Fitting CP to tensors with structure much cheaper
  - Moment tensors
  - Also sparse tensors
- Even still, there is opportunity for improvements using randomized methods
- Yet another example of computing a stochastic gradient

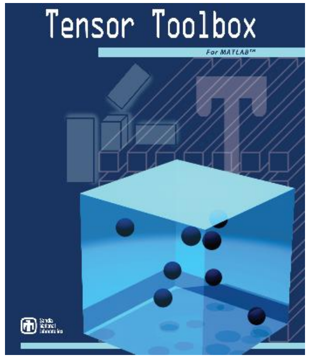


Sandia National Laboratories

Creativity + Randomization = Improved Data Analysis

- Applied naively, randomization fails
  - Computationally expensive to sketch/sample
  - High error and/or slow convergence
- Sketching creates a smaller problem
  - Mixing is expensive – make it cheaper or avoid it?
  - Theoretical bounds much worse than practice - why?
  - For subproblems – do things improve or get worse ?
  - How can we handle missing data in sketches?
- Stochastic gradient descent uses cheap estimate
  - Relationship to sketching largely unexplored
  - Variance reduction – too little versus too much?
  - More work needed on controlling step length

Questions/Comments: tgkolda@sandia.gov



Tensor Toolbox for MATLAB
www.tensortoolbox.org

Papers & Slides
www.kolda.net

References Published/Posted

- A. H. Williams et al. **Unsupervised Discovery of Demixed, Low-dimensional Neural Dynamics across Multiple Timescales through Tensor Components Analysis**. *Neuron*, 2018
- C. Battaglino, G. Ballard, T. G. Kolda. **A Practical Randomized CP Tensor Decomposition**. *SIAM Journal on Matrix Analysis and Applications*, 2018
- D. Hong, T. G. Kolda, J. A. Duersch. **Generalized Canonical Polyadic Tensor Decomposition**. *SIAM Review*, in press, 2019
- T. G. Kolda, D. Hong. **Stochastic Gradients for Large-Scale Tensor Decomposition**. arXiv:1906.01687, 2019

References Coming Soon

- R. Jin, T. G. Kolda, R. Ward. **Faster Johnson-Lindenstrauss Transforms via Kronecker Product.**
- T. G. Kolda, B. Larsen. **Leverage Score Sampling for Randomized CP Tensor Decomposition.**
- S. Sherman, T. G. Kolda, **Estimating Higher-Order Moments Using Symmetric Tensor Decomposition**.