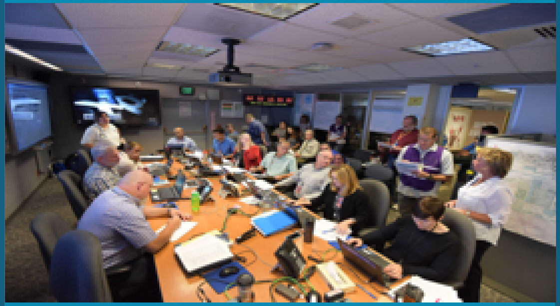
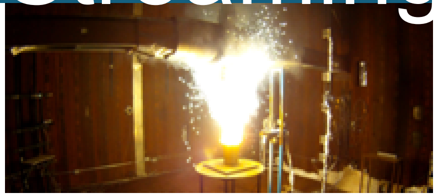
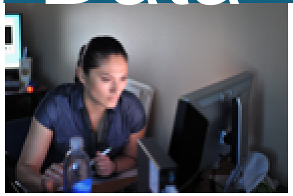
Sandia National Laboratories

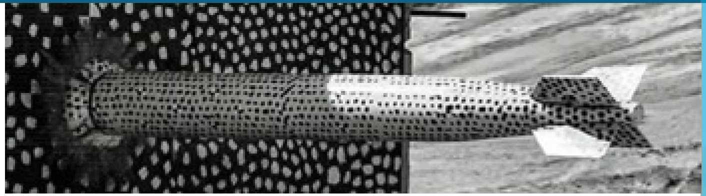
SAND2021-2087C

Generalized Canonical Polyadic Tensor Decompositions for Streaming Data









SIAM CSE 21, March 1-5, 2021

Eric Phipps, Nick Johnson, Tammy Kolda

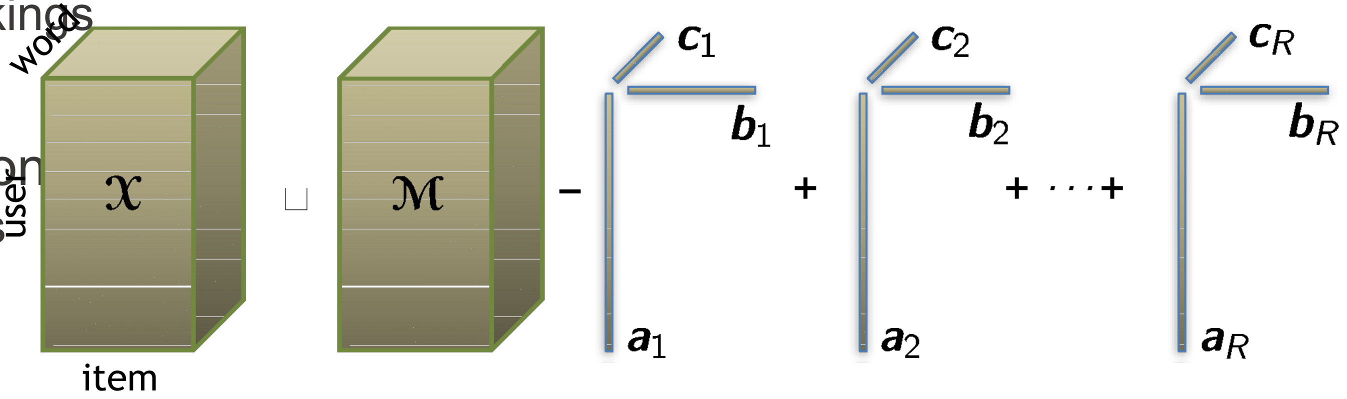
Sandia National Laboratories

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & 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-NA0003525.

Tensors and the CP Decomposition

N-way array used to represent multi-relationship data

- E.g., word frequencies in Amazon product rankings



Canonical Polyadic (CP) tensor decomposition

- Approximate tensor as a sum of rank-1 tensors
- Discovers dominant relationships in data

CP minimization problem

$$\min_{\mathcal{M}} \|\mathcal{X} - \mathcal{M}\|_F^2 \text{ s.t. } \mathcal{M} = \mathbf{a}_1 \circ \mathbf{b}_1 \circ \mathbf{c}_1 + \cdots + \mathbf{a}_R \circ \mathbf{b}_R \circ \mathbf{c}_R = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}]\!]$$

$$\mathbf{A} = [\mathbf{a}_1 \ \ldots \ \mathbf{a}_R]$$
$$\mathbf{B} = [\mathbf{b}_1 \ \ldots \ \mathbf{b}_R]$$
$$\mathbf{C} = [\mathbf{c}_1 \ \ldots \ \mathbf{c}_R]$$

Often solved via alternating linear least squares

- Fix all but one term, solve linear least squares problem, iterate

Using…

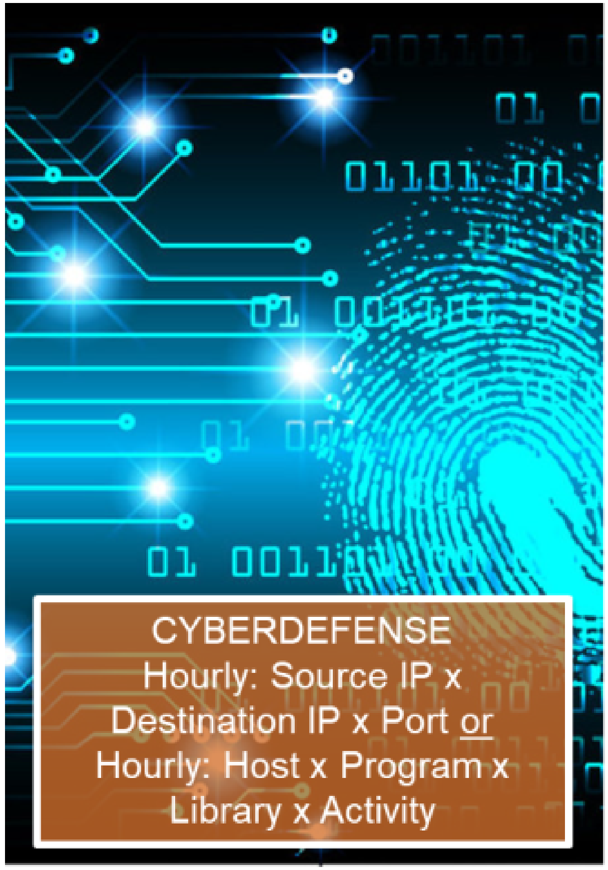
$$\mathcal{M}_{(1)} = \mathbf{A}(\mathbf{C} \odot \mathbf{B})^T$$
$$\mathcal{M}_{(2)} = \mathbf{B}(\mathbf{C} \odot \mathbf{A})^T$$
$$\mathcal{M}_{(3)} = \mathbf{C}(\mathbf{B} \odot \mathbf{A})^T$$

Repeat until convergence…

$$\min_{\mathbf{A}} \|\mathcal{X}_{(1)} - \mathbf{A}(\mathbf{C} \odot \mathbf{B})^T\|_F^2$$
$$\min_{\mathbf{B}} \|\mathcal{X}_{(2)} - \mathbf{B}(\mathbf{C} \odot \mathbf{A})^T\|_F^2$$
$$\min_{\mathbf{C}} \|\mathcal{X}_{(3)} - \mathbf{C}(\mathbf{B} \odot \mathbf{A})^T\|_F^2$$

Repeat until convergence…

$$\mathbf{A} = \mathcal{X}_{(1)}(\mathbf{C} \odot \mathbf{B})(\mathbf{C}^T\mathbf{C} * \mathbf{B}^T\mathbf{B})^\dagger$$
$$\mathbf{B} = \mathcal{X}_{(2)}(\mathbf{C} \odot \mathbf{A})(\mathbf{C}^T\mathbf{C} * \mathbf{A}^T\mathbf{A})^\dagger$$
$$\mathbf{C} = \underbrace{\mathcal{X}_{(3)}(\mathbf{B} \odot \mathbf{A})}_{\text{MTTKRP}}(\mathbf{B}^T\mathbf{B} * \mathbf{A}^T\mathbf{A})^\dagger$$

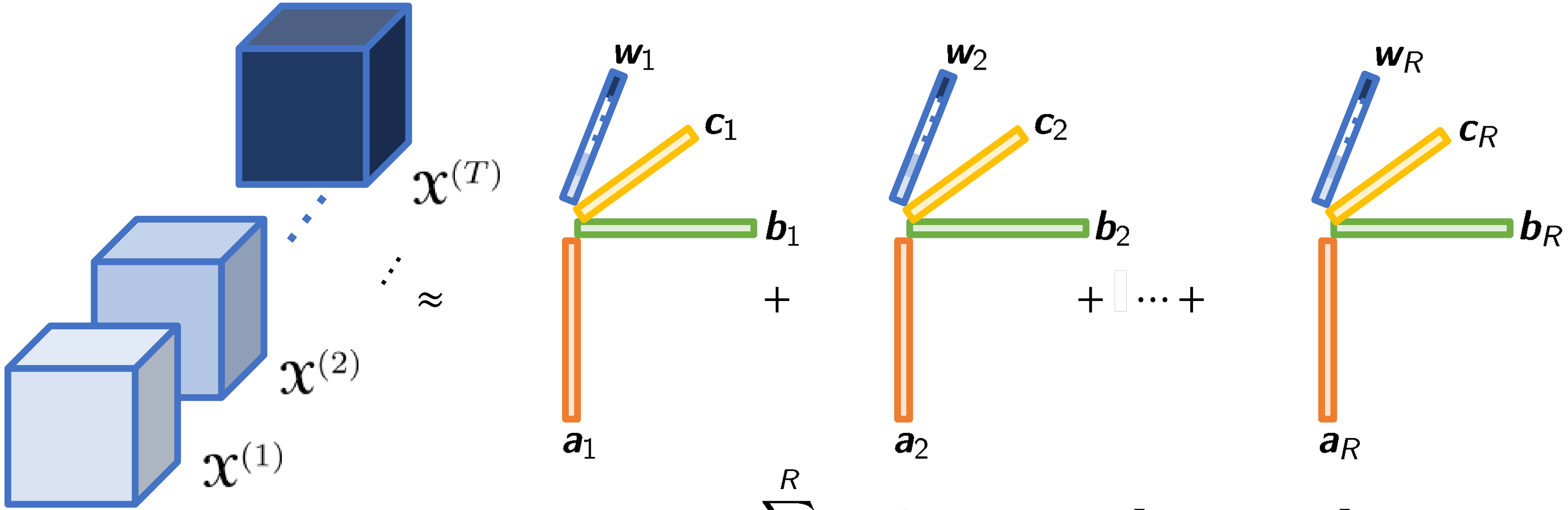


Examples of Streaming Data

Streaming Tensors – Two Points-of-View

At each time step t, new 3-way hyperslice added

If we assume factor matrices $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ *fixed through time*, then the ideal factorization looks like…



$$\mathcal{X} \approx \sum_{j=1}^{R} \mathbf{a}_j \circ \mathbf{b}_j \circ \mathbf{c}_j \circ \mathbf{w}_j = [\![\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{W}]\!]$$

$$\mathcal{X} \in \mathbb{R}^{N_1 \times N_2 \times N_3 \times T}$$

$$\mathcal{X}^{(t)} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$$

$$\mathbf{A} = [\mathbf{a}_1 \; \mathbf{a}_2 \; \ldots \; \mathbf{a}_R] \in \mathbb{R}^{N_1 \times R}$$

$$\mathbf{B} = [\mathbf{b}_1 \; \mathbf{b}_2 \; \ldots \; \mathbf{b}_R] \in \mathbb{R}^{N_2 \times R}$$

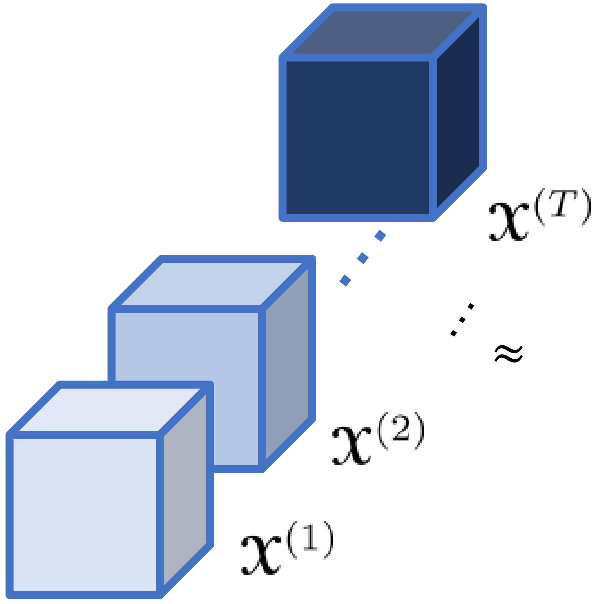
$$\mathbf{C} = [\mathbf{c}_1 \; \mathbf{c}_2 \; \ldots \; \mathbf{c}_R] \in \mathbb{R}^{N_3 \times R}$$

$$\mathbf{W} = [\mathbf{w}_1 \; \mathbf{w}_2 \; \ldots \; \mathbf{w}_R] \in \mathbb{R}^{T \times R}$$

Generic d-way setup is similar.

Streaming Tensors – Two Points-of-View

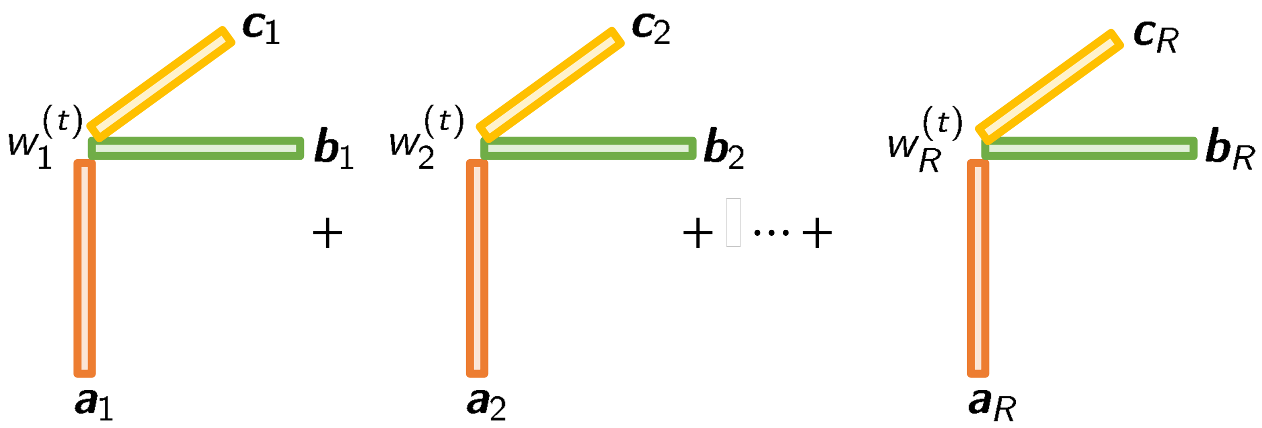
At each time step t, new 3-way hyperslice added



$$\mathcal{X} \in \mathbb{R}^{N_1 \times N_2 \times N_3 \times T}$$

$$\mathcal{X}^{(t)} \in \mathbb{R}^{N_1 \times N_2 \times N_3}$$

If we assume factor matrices $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$ *fixed through time*, then the ideal factorization looks like…



$$\mathcal{X}^{(t)} \approx \sum_{j=1}^{R} w_j^{(t)} \boldsymbol{a}_j \circ \boldsymbol{b}_j \circ \boldsymbol{c}_j = [\![\boldsymbol{w}^{(t)}; \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}]\!]$$

$$\boldsymbol{A} = [\boldsymbol{a}_1 \ \boldsymbol{a}_2 \ \ldots \ \boldsymbol{a}_R] \in \mathbb{R}^{N_1 \times R}$$

$$\boldsymbol{B} = [\boldsymbol{b}_1 \ \boldsymbol{b}_2 \ \ldots \ \boldsymbol{b}_R] \in \mathbb{R}^{N_2 \times R}$$

$$\boldsymbol{C} = [\boldsymbol{c}_1 \ \boldsymbol{c}_2 \ \ldots \ \boldsymbol{c}_R] \in \mathbb{R}^{N_3 \times R}$$

Generic d-way setup is similar.

Generic Streaming Formulation

$$\min \left\| \mathcal{X}^{(t)} - [\![\mathbf{w}^{(t)}; \mathbf{A}^{(t)}, \mathbf{B}^{(t)}, \mathbf{C}^{(t)}]\!] \right\|^2 + \sum_{\tau=1}^{t-1} \theta^{t-\tau} \left\| \mathcal{X}^{(\tau)} - [\![\mathbf{w}^{(\tau)}; \mathbf{A}^{(t)}, \mathbf{B}^{(t)}, \mathbf{C}^{(t)}]\!] \right\|^2 + \mathcal{R}(\mathbf{w}^{(t)}) + \mathcal{R}(\mathbf{A}^{(t)}) + \mathcal{R}(\mathbf{B}^{(t)}) + \mathcal{R}(\mathbf{C}^{(t)})$$

Matching to the latest temporal observation

Factor matrices vary slowly or not at all

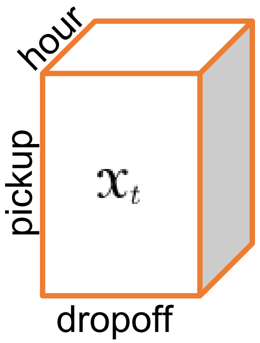
Fit historical data with updated factors (which should change slowly)

Down-weight old information

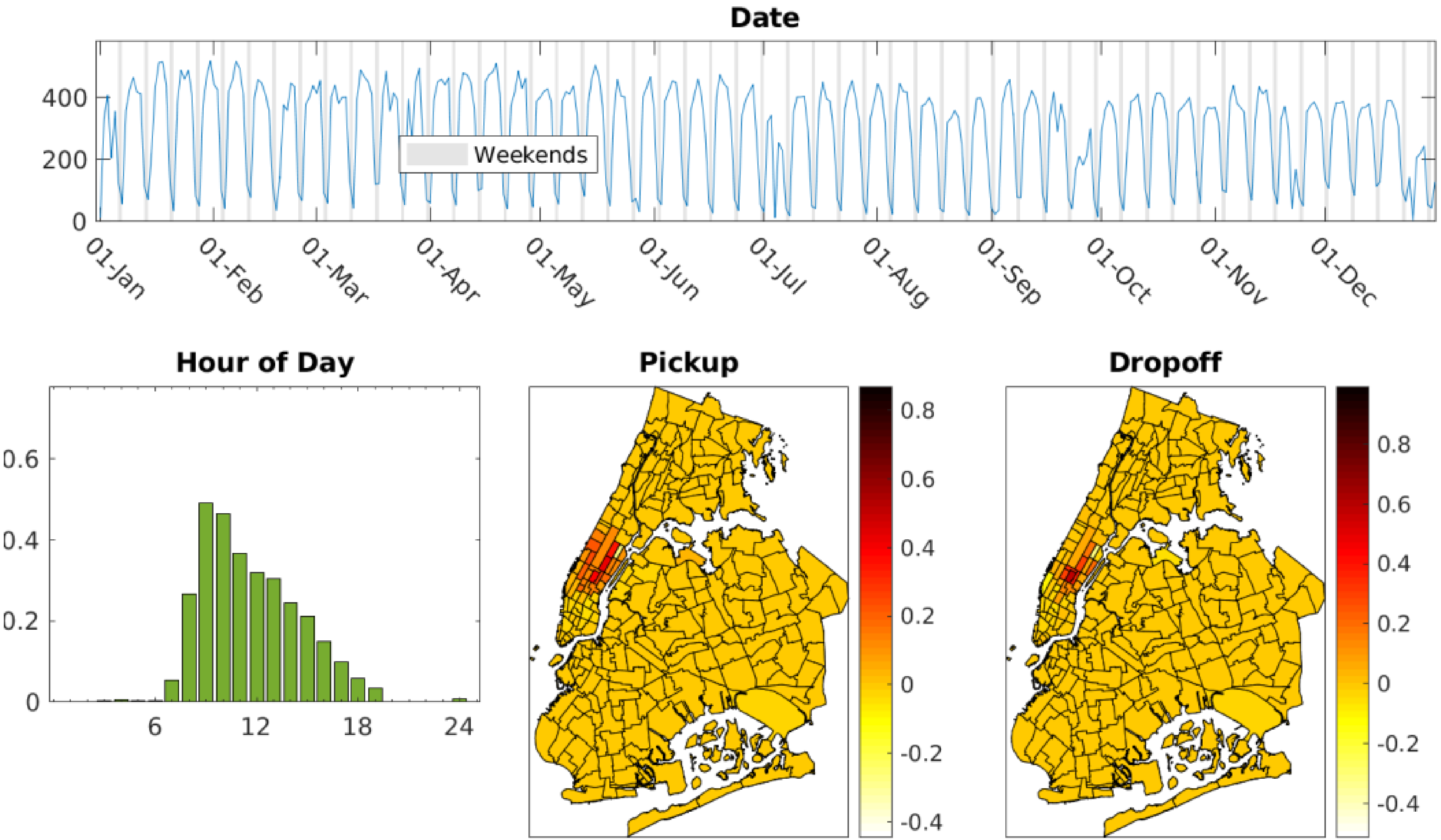
Optionally regularize or impose other constraints like nonnegativity

NYC Taxi Dataset – 4-way Tensor

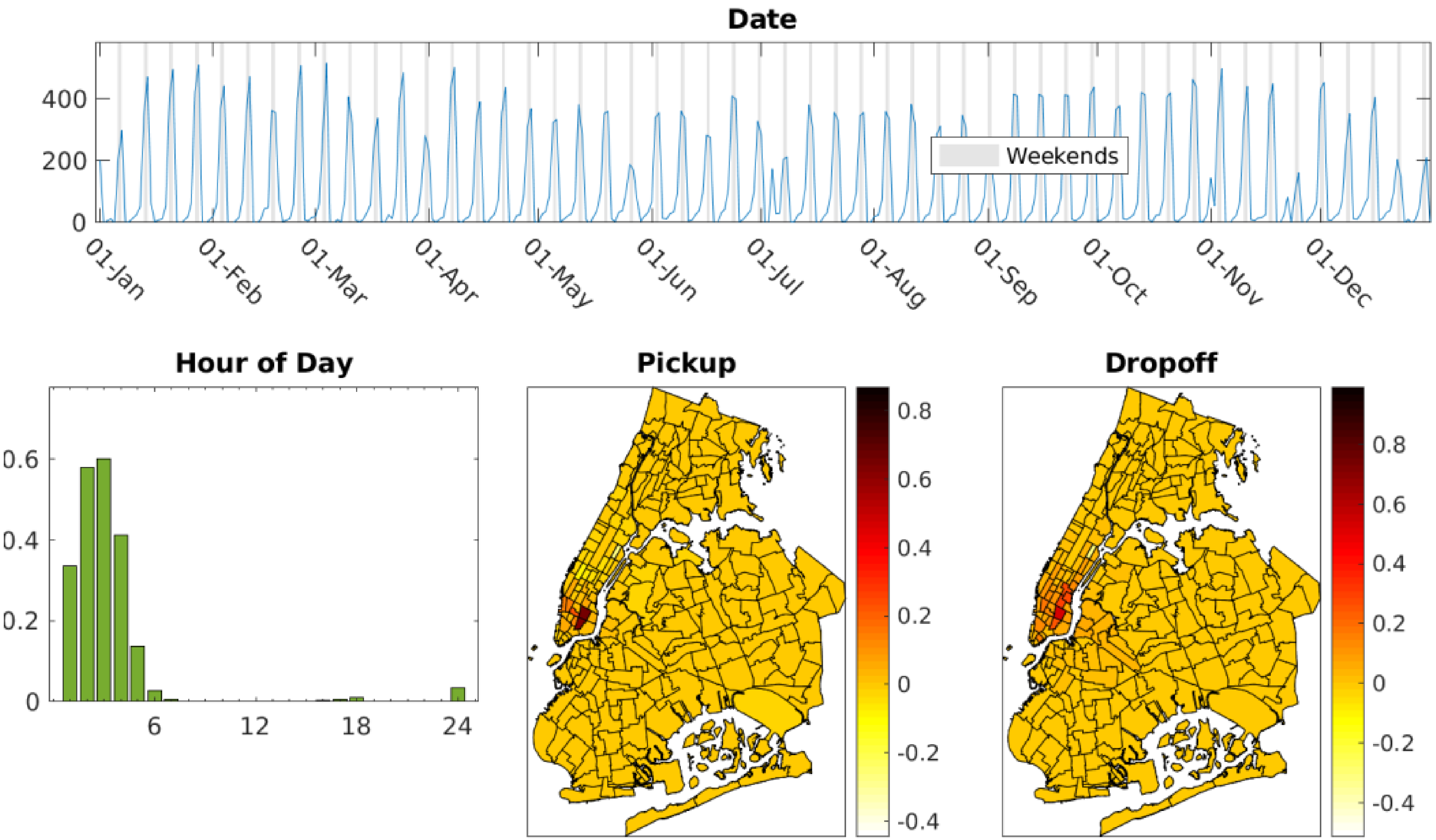
- **Data from NYC public records**
  https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page
- **10+ Years of Data**
- **4-way Tensor, Updated Daily**
  - Pickup Zone
  - Dropoff Zone
  - Pickup Hour
  - *New 3-way tensor each day*
- **265 Taxi Zones**
  https://catalog.data.gov/dataset/nyc-taxi-zones



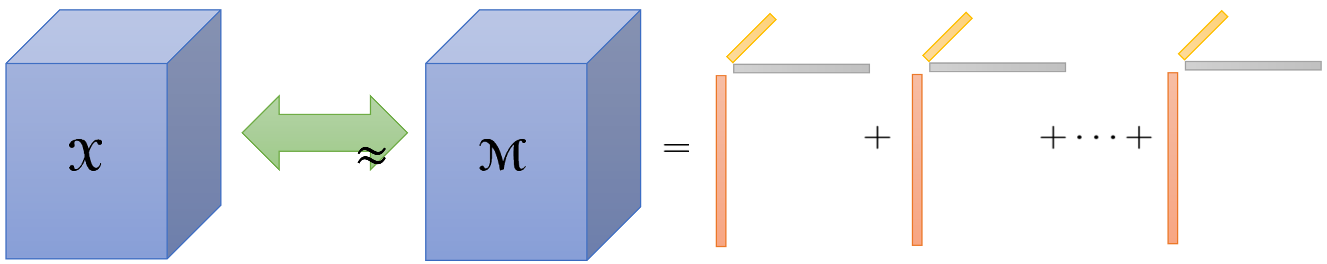
Component #1 (of 50) – Standard Mid-morning Weekday Traffic



Component #21 (of 50): Weekend Nightlife



Generalized CP (GCP) Tensor Decomposition Allows Flexible Loss Function



Generalized CP (GCP)

$$\min_{\mathbf{A}_1,\dots,\mathbf{A}_d} \quad F(\mathcal{X},\mathcal{M}) = \sum_i f(x_i, m_i)$$

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

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

Hong, Kolda, Duersh, *SIREV* 2020 (arXiv:1808.07452)

Example Loss Functions

Normal ($x, m \in \mathbb{R}$)

$$f(x,m) = (x-m)^2$$

Poisson ($x \in \mathbb{N}, m > 0$)

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

Bernoulli ($x \in \{0,1\}, m > 0$)

$$f(x,m) = \log(m+1) - x \log m$$

β-divergence ($x > 0, m > 0, \beta = \frac{1}{2}$)

$$f(x,m) = x/\sqrt{m} + \sqrt{m}$$

Fitting GCP Model*

$$\min_{\mathcal{M}} \; F(\mathcal{X}, \mathcal{M}) = \sum_i f(x_i, m_i) \quad \text{s.t.} \quad \mathcal{M} = [\![\boldsymbol{A}_1, \ldots, \boldsymbol{A}_d]\!]$$

Lose the least-squares structure underlying ALS-type algorithms. Instead pursue gradient-based optimization approach.

Define tensor Y such that

$$y(i_1, \ldots, i_d) = y_i = \frac{\partial f}{\partial m}(x_i, m_i)$$

Then gradient of objective function given by

$$\boldsymbol{G}_k = \frac{\partial F}{\partial \boldsymbol{A}_k} = \mathcal{Y}_{(k)}(\boldsymbol{A}_d \odot \cdots \odot \boldsymbol{A}_{k+1} \odot \boldsymbol{A}_{k-1} \odot \cdots \odot \boldsymbol{A}_1)$$

← MTTKRP!

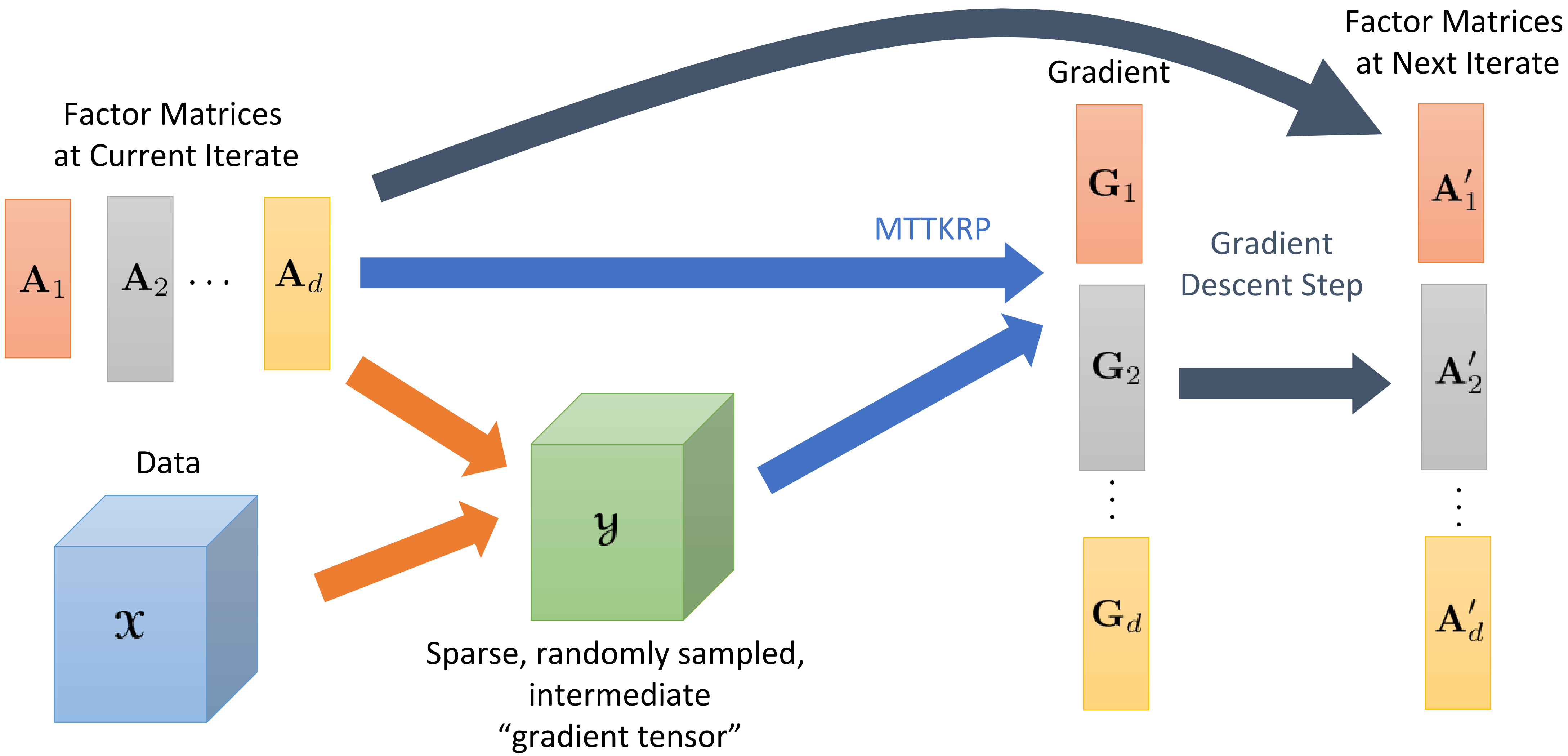
Unfortunately, Y is in general dense, even when X is sparse, making standard optimization infeasible.

Instead, employ Stochastic Gradient Descent (SGD) where Y is only randomly sampled

- Stratified: sample zeros and nonzeros separately (requires tensor search)
- Semi-stratified: skip tensor search and adjust for "zeros" that are really nonzeros

*Kolda, Hong, Duersch. Stochastic Gradients for Large-Scale Tensor Decomposition. arXiv 1906.01687, 2019.

High-level View of Optimization & Dependencies



Towards Streaming "Online GCP"

Starting from OnlineSGD (Mardani et al., TSP 2015) :

$$\min \left[\left\| \mathcal{X}^{(t)} - [\![\boldsymbol{w}^{(t)}; \boldsymbol{A}^{(t)}, \boldsymbol{B}^{(t)}, \boldsymbol{C}^{(t)}]\!] \right\|^2 + \lambda \left(\|\boldsymbol{w}^{(t)}\|^2 + \|\boldsymbol{A}^{(t)}\|^2 + \|\boldsymbol{B}^{(t)}\|^2 + \|\boldsymbol{C}^{(t)}\|^2 \right) \right]$$

For each time step *t:*

- Least-squares solve for $\boldsymbol{w}^{(t)}$ holding $\boldsymbol{A}^{(t)}=\boldsymbol{A}^{(t-1)}$, $\boldsymbol{B}^{(t)}=\boldsymbol{B}^{(t-1)}$, $\boldsymbol{C}^{(t)}=\boldsymbol{C}^{(t-1)}$ fixed
- Gradient descent updates to $\boldsymbol{A}^{(t)}$,$\boldsymbol{B}^{(t)}$, $\boldsymbol{C}^{(t)}$ holding $\boldsymbol{w}^{(t)}$ fixed

Implemented in combination of Matlab Tensor Toolbox and C++ GenTen* GCP library

- High-level algorithm in Matlab
- Fast GenTen math kernels using MEX interface

*Phipps and Kolda, *Software for Sparse Tensor Decomposition on Emerging Computing Architectures*, SIAM SISC, 2019.
https://gitlab.com/tensors/genten

Towards Streaming "Online GCP"

Starting from OnlineSGD (Mardani et al., TSP 2015) :

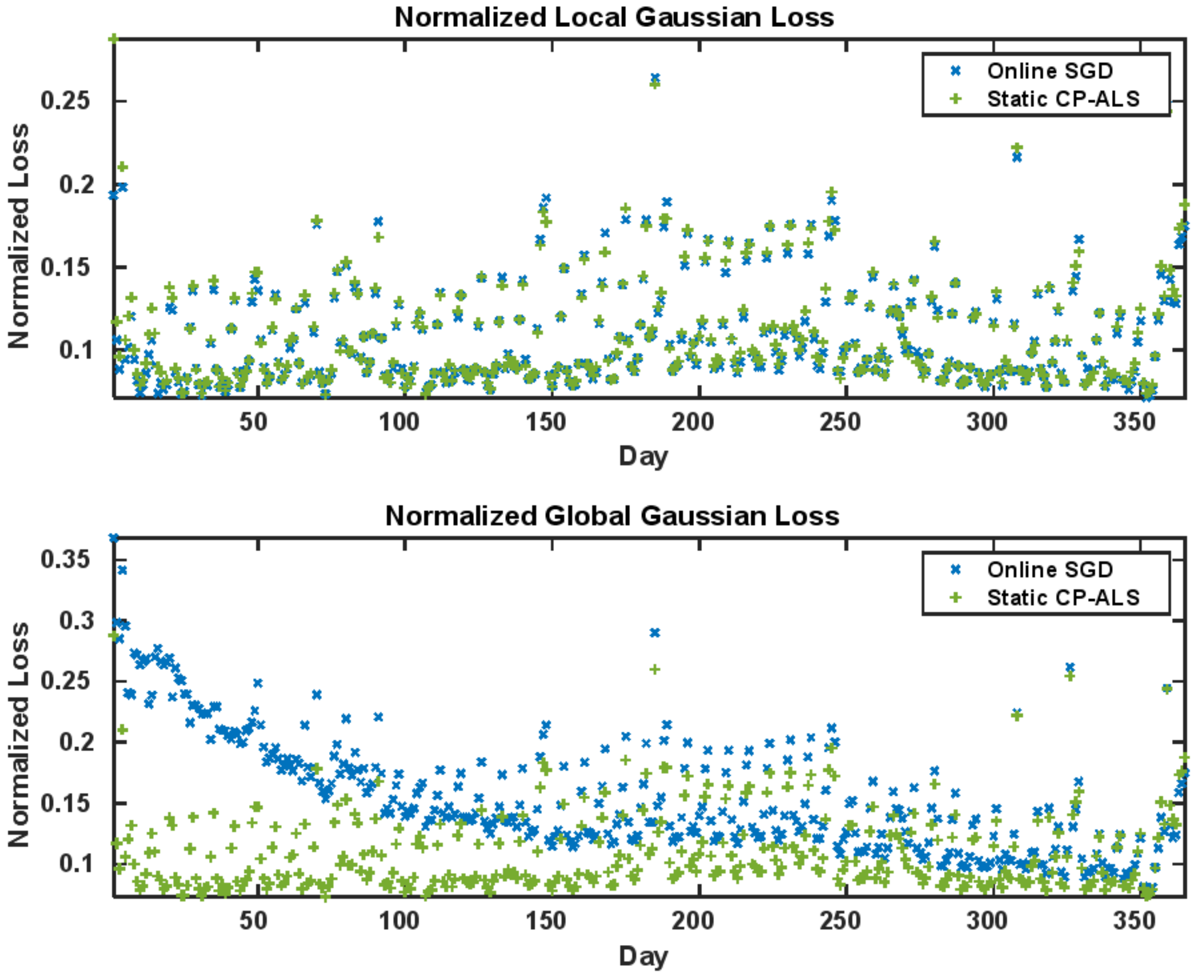
$$\min \left[\left\| \mathcal{X}^{(t)} - [\![\boldsymbol{w}^{(t)}; \boldsymbol{A}^{(t)}, \boldsymbol{B}^{(t)}, \boldsymbol{C}^{(t)}]\!] \right\|^2 + \lambda \left(\|\boldsymbol{w}^{(t)}\|^2 + \|\boldsymbol{A}^{(t)}\|^2 + \|\boldsymbol{B}^{(t)}\|^2 + \|\boldsymbol{C}^{(t)}\|^2 \right) \right]$$

For each time step *t:*

- Least-squares solve for $\boldsymbol{w}^{(t)}$ holding $\boldsymbol{A}^{(t)}=\boldsymbol{A}^{(t-1)}$, $\boldsymbol{B}^{(t)}=\boldsymbol{B}^{(t-1)}$, $\boldsymbol{C}^{(t)}=\boldsymbol{C}^{(t-1)}$ fixed
- Gradient descent updates to $\boldsymbol{A}^{(t)}$,$\boldsymbol{B}^{(t)}$, $\boldsymbol{C}^{(t)}$ holding $\boldsymbol{w}^{(t)}$ fixed

Implemented in combination of Matlab Tensor Toolbox and C++ GenTen* GCP library

- High-level algorithm in Matlab
- Fast GenTen math kernels using MEX interface



Comparison of Online SGD local and global Gaussian loss applied to 1 year of NYC taxicab data with non-streaming CP-ALS (from GenTen).

$$\text{Normalized Local Gaussian Loss} = \|\mathcal{X}^{(t)} - [\![\boldsymbol{w}^{(t)}; \boldsymbol{A}^{(t)}, \boldsymbol{B}^{(t)}, \boldsymbol{C}^{(t)}]\!]\|^2 / \|\mathcal{X}^{(t)}\|^2$$

$$\text{Normalized Global Gaussian Loss} = \|\mathcal{X}^{(t)} - [\![\boldsymbol{w}^{(t)}; \boldsymbol{A}^{(T)}, \boldsymbol{B}^{(T)}, \boldsymbol{C}^{(T)}]\!]\|^2 / \|\mathcal{X}^{(t)}\|^2$$

*Phipps and Kolda, *Software for Sparse Tensor Decomposition on Emerging Computing Architectures*, SIAM SISC, 2019.
https://gitlab.com/tensors/genten

Towards Streaming "Online GCP"

Starting from OnlineSGD (Mardani et al., TSP 2015) :

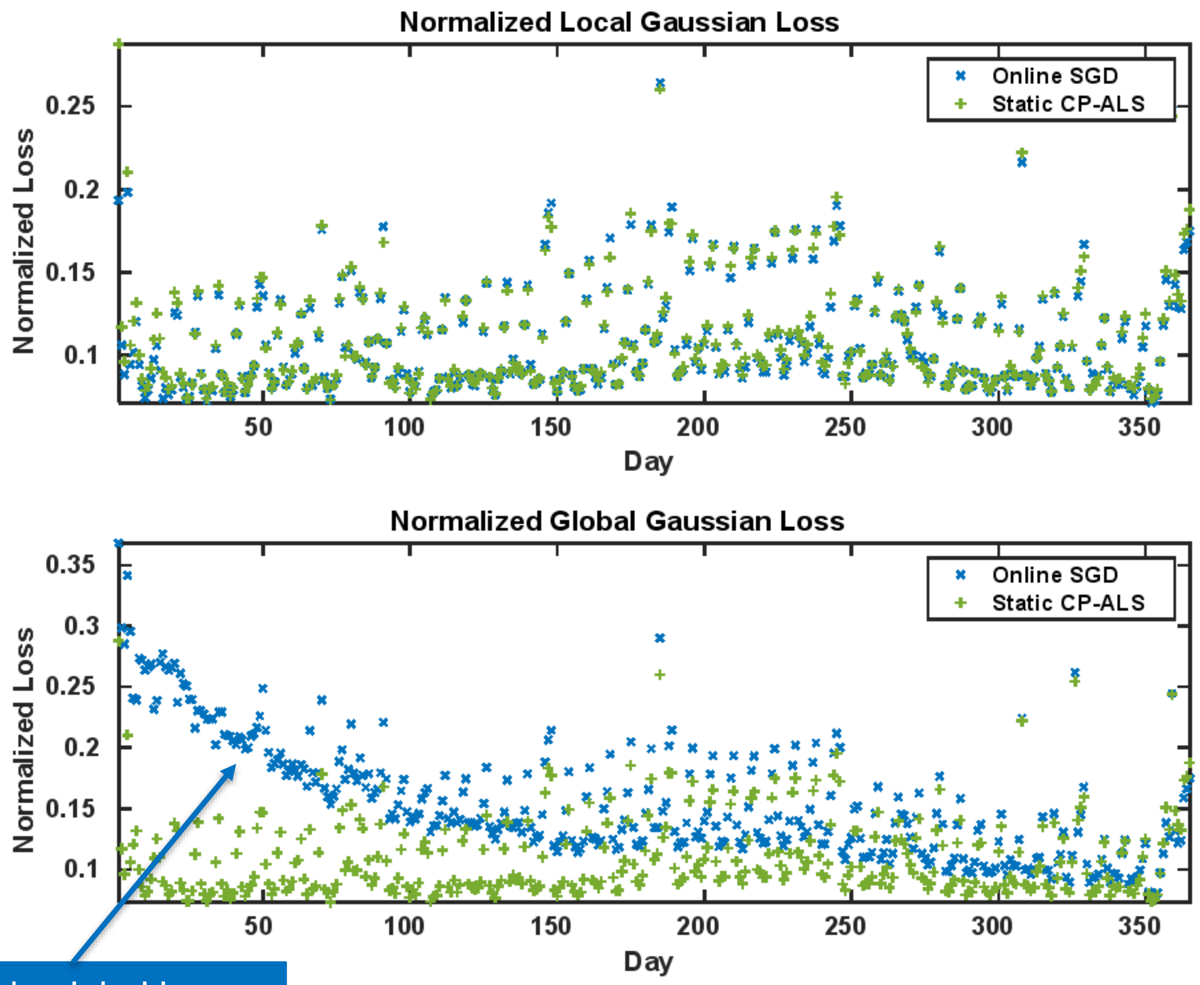
$$\min \left[\left\| \mathcal{X}^{(t)} - [\![\boldsymbol{w}^{(t)}; \boldsymbol{A}^{(t)}, \boldsymbol{B}^{(t)}, \boldsymbol{C}^{(t)}]\!] \right\|^2 + \lambda \left(\|\boldsymbol{w}^{(t)}\|^2 + \|\boldsymbol{A}^{(t)}\|^2 + \|\boldsymbol{B}^{(t)}\|^2 + \|\boldsymbol{C}^{(t)}\|^2 \right) \right]$$

For each time step *t*:

- Least-squares solve for $\boldsymbol{w}^{(t)}$ holding $\boldsymbol{A}^{(t)}=\boldsymbol{A}^{(t-1)}$, $\boldsymbol{B}^{(t)}=\boldsymbol{B}^{(t-1)}$, $\boldsymbol{C}^{(t)}=\boldsymbol{C}^{(t-1)}$ fixed
- Gradient descent updates to $\boldsymbol{A}^{(t)}$, $\boldsymbol{B}^{(t)}$, $\boldsymbol{C}^{(t)}$ holding $\boldsymbol{w}^{(t)}$ fixed

Implemented in combination of Matlab Tensor Toolbox and C++ GenTen* GCP library

- High-level algorithm in Matlab
- Fast GenTen math kernels using MEX interface



Growth in global loss due to missing history

Comparison of Online SGD local and global Gaussian loss applied to 1 year of NYC taxicab data with non-streaming CP-ALS (from GenTen).

$$\text{Normalized Local Gaussian Loss} = \|\mathcal{X}^{(t)} - [\![\boldsymbol{w}^{(t)}; \boldsymbol{A}^{(t)}, \boldsymbol{B}^{(t)}, \boldsymbol{C}^{(t)}]\!]\|^2 / \|\mathcal{X}^{(t)}\|^2$$

$$\text{Normalized Global Gaussian Loss} = \|\mathcal{X}^{(t)} - [\![\boldsymbol{w}^{(t)}; \boldsymbol{A}^{(T)}, \boldsymbol{B}^{(T)}, \boldsymbol{C}^{(T)}]\!]\|^2 / \|\mathcal{X}^{(t)}\|^2$$

*Phipps and Kolda, *Software for Sparse Tensor Decomposition on Emerging Computing Architectures*, SIAM SISC, 2019.
https://gitlab.com/tensors/genten

Adding History Term To Improve Global Loss

Add history regularization term to prevent over-solving for new slices

- Essentially penalizing change in factor matrices

$$\min_{\mathcal{M}^{(t)}} \Bigg[\left\| \mathcal{X}^{(t)} - \mathcal{M}^{(t)} \right\|^2 + \sum_{\tau \in T_h} \theta^{t-\tau} \left\| \mathcal{X}^{(\tau)} - \left[\!\left[\boldsymbol{w}^{(\tau)}; \boldsymbol{A}^{(t)}, \boldsymbol{B}^{(t)}, \boldsymbol{C}^{(t)} \right]\!\right] \right\|^2 + \lambda \left(\|\boldsymbol{w}^{(t)}\|^2 + \|\boldsymbol{A}^{(t)}\|^2 + \|\boldsymbol{B}^{(t)}\|^2 + \|\boldsymbol{C}^{(t)}\|^2 \right) \Bigg]$$

Fixed-size history window

- Each new slice randomly evicts a previous entry
- Many ways history could be captured depending on the problem of interest

Adding History Term To Improve Global Loss

Add history regularization term to prevent over-solving for new slices

- Approximate old slices with CP-model from previous time step and old time weights (e.g., CP-Stream, Smith et al, SDM, 2018)
- Only requires storing old time weights, not slices
- Essentially penalizing change in factor matrices

$$\min_{\mathcal{M}^{(t)}} \Bigg[\left\| \mathcal{X}^{(t)} - \mathcal{M}^{(t)} \right\|^2 + \sum_{\tau \in T_h} \theta^{t-\tau} \left\| [\![\boldsymbol{w}^{(\tau)}; \boldsymbol{A}^{(t-1)}, \boldsymbol{B}^{(t-1)}, \boldsymbol{C}^{(t-1)}]\!] - [\![\boldsymbol{w}^{(\tau)}; \boldsymbol{A}^{(t)}, \boldsymbol{B}^{(t)}, \boldsymbol{C}^{(t)}]\!] \right\|^2 + \lambda \left(\|\boldsymbol{w}^{(t)}\|^2 + \|\boldsymbol{A}^{(t)}\|^2 + \|\boldsymbol{B}^{(t)}\|^2 + \|\boldsymbol{C}^{(t)}\|^2 \right) \Bigg]$$

Fixed-size history window

- Each new slice randomly evicts a previous entry
- Many ways history could be captured depending on the problem of interest

Adding History Term To Improve Global Loss

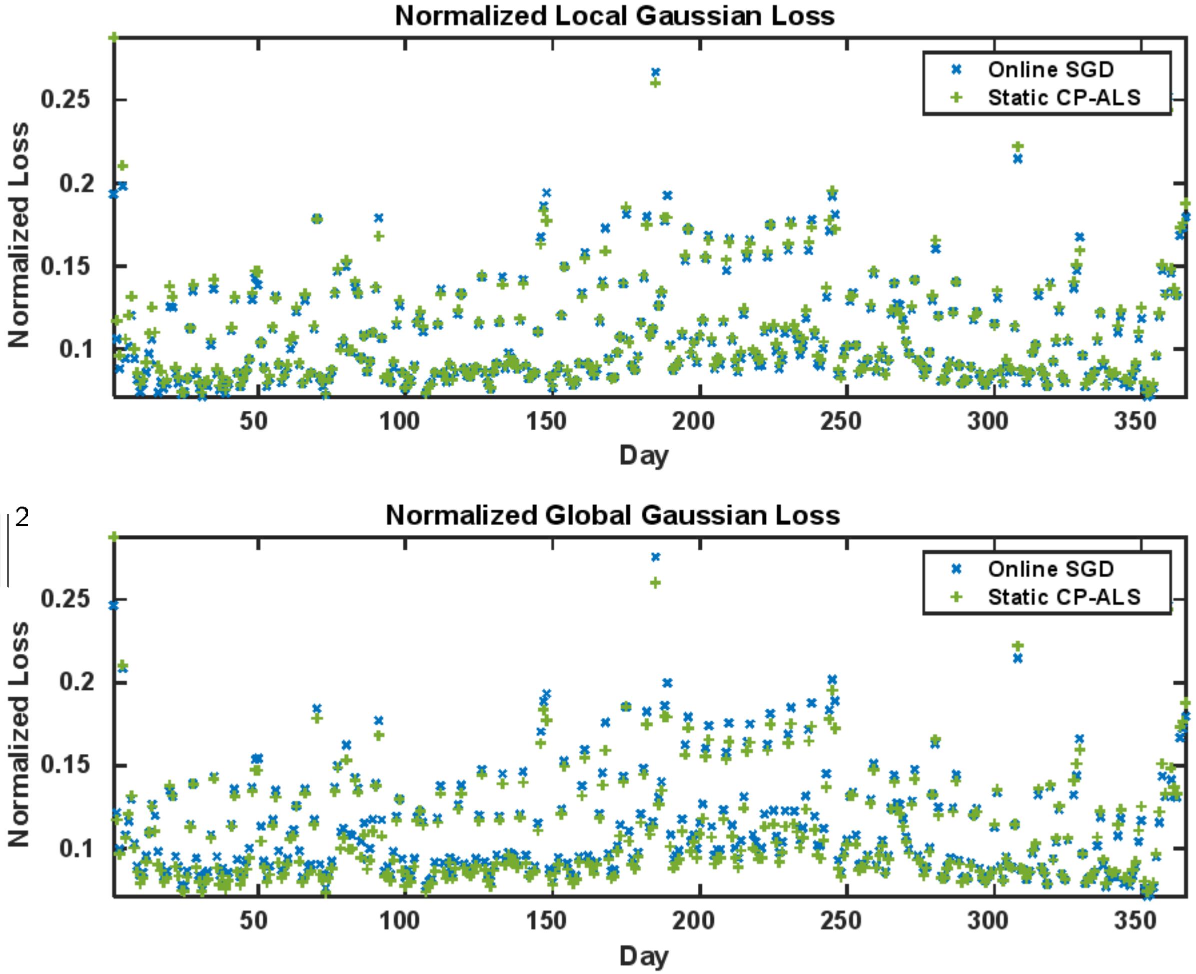
Add history regularization term to prevent over-solving for new slices

- Approximate old slices with CP-model from previous time step and old time weights (e.g., CP-Stream, Smith et al, SDM, 2018)
- Only requires storing old time weights, not slices
- Essentially penalizing change in factor matrices

$$\min_{\mathcal{M}^{(t)}} \Bigg[\left\|\mathcal{X}^{(t)} - \mathcal{M}^{(t)}\right\|^2 + \sum_{\tau \in T_h} \theta^{t-\tau} \left\| [\![\mathbf{w}^{(\tau)}; \mathbf{A}^{(t-1)}, \mathbf{B}^{(t-1)}, \mathbf{C}^{(t-1)}]\!] - [\![\mathbf{w}^{(\tau)}; \mathbf{A}^{(t)}, \mathbf{B}^{(t)}, \mathbf{C}^{(t)}]\!] \right\|^2 + \lambda \left(\|\mathbf{w}^{(t)}\|^2 + \|\mathbf{A}^{(t)}\|^2 + \|\mathbf{B}^{(t)}\|^2 + \|\mathbf{C}^{(t)}\|^2 \right) \Bigg]$$

Fixed-size history window

- Each new slice randomly evicts a previous entry
- Many ways history could be captured depending on the problem of interest



Comparison of local and global Gaussian loss applied to 1 year of NYC taxicab data with 30 randomly selected history window.

Streaming "Online GCP"

Replace sum-of-squares Frobenius norm with general GCP loss function (i.e., negative log likelihood):

- Replace least-squares solve with GCP-SGD solve for $\boldsymbol{w}^{(t)}$
- Replace gradient descent updates with GCP-SGD updates to $\boldsymbol{A}^{(t)}, \boldsymbol{B}^{(t)}, \boldsymbol{C}^{(t)}$ holding $\boldsymbol{w}^{(t)}$ fixed

$$\min_{\mathcal{M}^{(t)}} \Bigg[\sum_{i \in \Omega^{(t)}} f(x_i^{(t)}, m_i^{(t)}) + \sum_{\tau \in T_h} \theta^{t-\tau} \left\| [\![\boldsymbol{w}^{(\tau)}; \boldsymbol{A}^{(t-1)}, \boldsymbol{B}^{(t-1)}, \boldsymbol{C}^{(t-1)}]\!] - [\![\boldsymbol{w}^{(\tau)}; \boldsymbol{A}^{(t)}, \boldsymbol{B}^{(t)}, \boldsymbol{C}^{(t)}]\!] \right\|^2 + \lambda \left(\|\boldsymbol{w}^{(t)}\|^2 + \|\boldsymbol{A}^{(t)}\|^2 + \|\boldsymbol{B}^{(t)}\|^2 + \|\boldsymbol{C}^{(t)}\|^2 \right) \Bigg]$$

Streaming "Online GCP"

Replace sum-of-squares Frobenius norm with general GCP loss function (i.e., negative log likelihood):

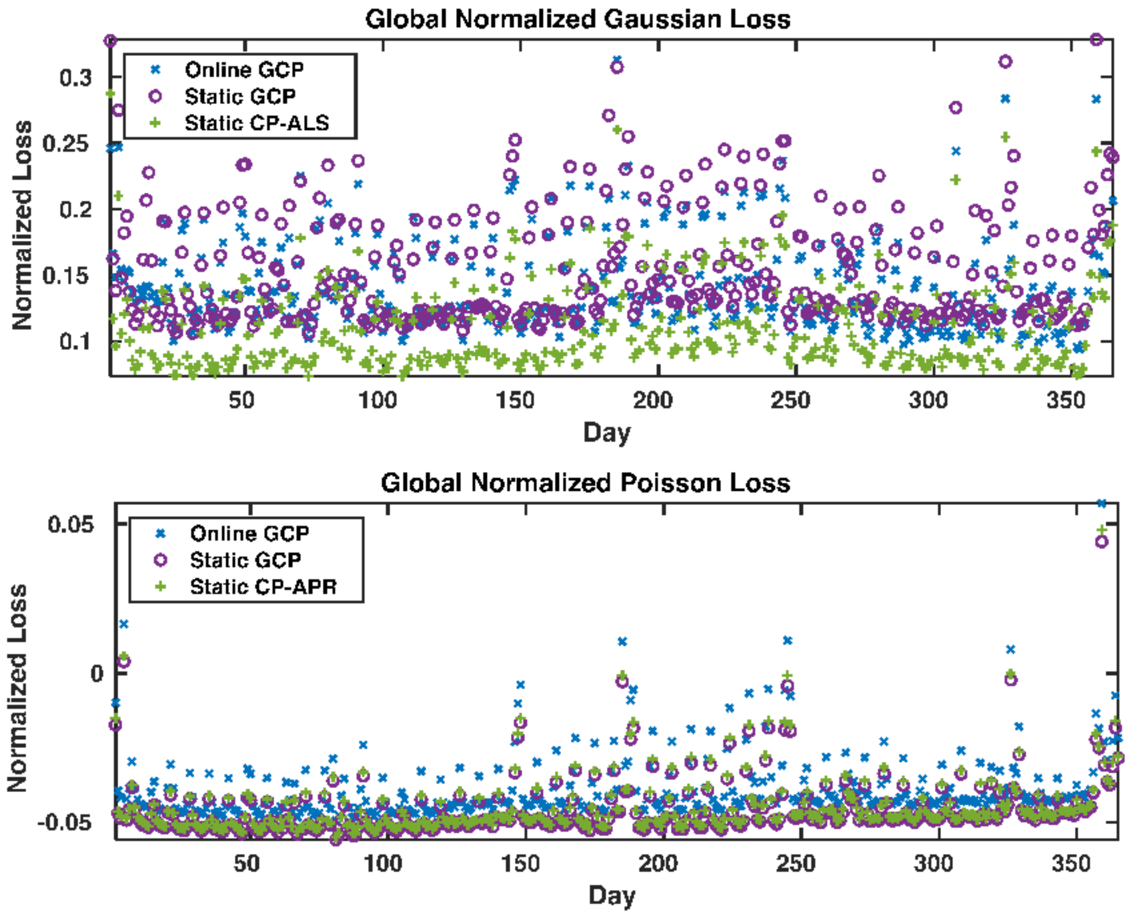
- Replace least-squares solve with GCP-SGD solve for $\boldsymbol{w}^{(t)}$
- Replace gradient descent updates with GCP-SGD updates to $\boldsymbol{A}^{(t)}$,$\boldsymbol{B}^{(t)}$, $\boldsymbol{C}^{(t)}$ holding $\boldsymbol{w}^{(t)}$ fixed

$$\min_{\mathcal{M}^{(t)}} \Bigg[\sum_{i \in \Omega^{(t)}} f(x_i^{(t)}, m_i^{(t)}) + \sum_{\tau \in T_h} \theta^{t-\tau} \left\| [\![\boldsymbol{w}^{(\tau)}; \boldsymbol{A}^{(t-1)}, \boldsymbol{B}^{(t-1)}, \boldsymbol{C}^{(t-1)}]\!] - [\![\boldsymbol{w}^{(\tau)}; \boldsymbol{A}^{(t)}, \boldsymbol{B}^{(t)}, \boldsymbol{C}^{(t)}]\!] \right\|^2 + \lambda \left(\|\boldsymbol{w}^{(t)}\|^2 + \|\boldsymbol{A}^{(t)}\|^2 + \|\boldsymbol{B}^{(t)}\|^2 + \|\boldsymbol{C}^{(t)}\|^2 \right) \Bigg]$$

$$\text{Global Normalized Gaussian Loss} = \|\mathcal{X}^{(t)} - \tilde{\mathcal{M}}^{(t,T)}\|^2 / \|\mathcal{X}^{(t)}\|^2,$$

$$\text{Global Normalized Poisson Loss} = \left(\sum_i \left(\tilde{m}_i^{(t,T)} - x_i^{(t)} \log(\tilde{m}_i^{(t,T)} + \epsilon) \right) \right) / \|\mathcal{X}^{(t)}\|^2,$$

$$\tilde{\mathcal{M}}^{(t,T)} = [\![\boldsymbol{w}^{(t)}; \boldsymbol{A}^{(T)}, \boldsymbol{B}^{(T)}, \boldsymbol{C}^{(T)}]\!]$$



Demonstration of streaming OnlineGCP with two likelihood functions, including non-streaming GCP, CP-ALS (Gaussian) and CP-APR (Poisson) on 1 year of NYC taxicab data.

Summary and Conclusions

Summary

- Streaming tensor decomposition method for general statistical data types (continuous, count, binary, …)
- Incorporation of history sampling to eliminate growth in global loss
- Software implementation combining rapid prototyping of Tensor Toolbox and manycore performance of GenTen

Challenges

- Online (and static) GCP require tuning of many hyper-parameters (number of samples, learning rates, …) particularly when balancing cost versus accuracy
- SGD solver may converge slowly increasing computational cost

Moving forward

- Proper formulation of history window regularization term
  - Likely should replace Frobenius norm with GCP loss.
  - How to sample that term (same samples as tensor or different, stratified or uniform, …)?
- In depth evaluation of performance, accuracy
- Address online and static GCP solver challenges

Backup Slides

Multiway "Tensor" Data is Ubiquitous

Neuron activity:
Neuron x Time x Trial

Crime data: Crime x Location x Hour x Date

Social interaction data:
Person A x Person B
x Venue x Time

Twitter co-occurrence:
Term A x Term B x Time

Travel data: Start Location
x Finish Location
x Departure Hour
x Departure Date

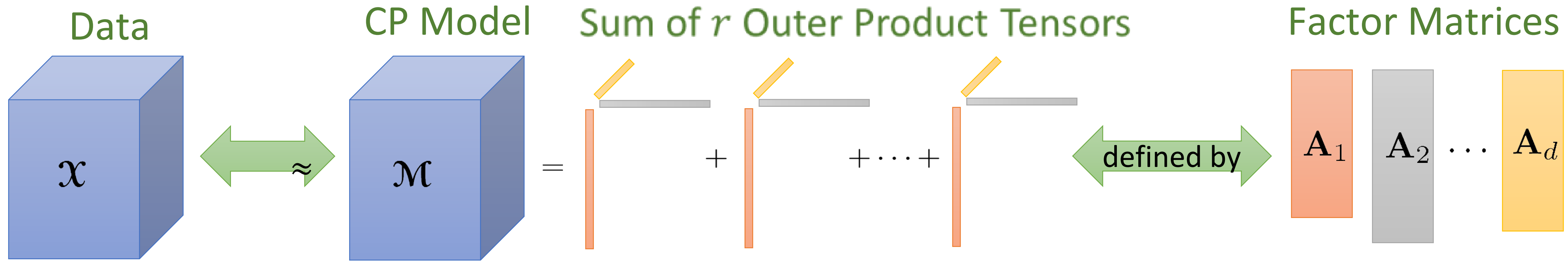
Signal processing:
Sensor x Frequency x Time

Cyber data: Src IP x Dst IP
x Dst Port x Time

Host
x Acti

Tensor Decomposition Finds Patterns in Massive Data (Unsupervised Learning)

Tensor Decomposition Identifies Factors



$\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$

$\mathcal{M} = [\![\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_d]\!] \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$

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

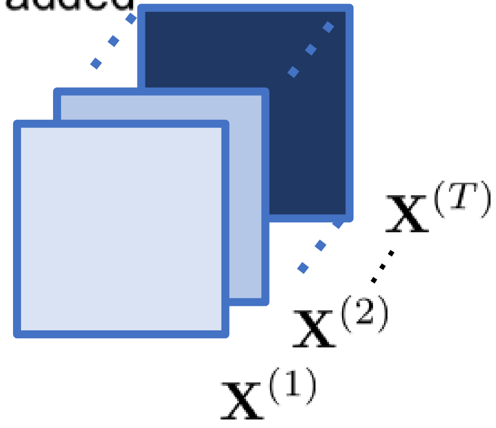
$x_i = x(i_1, i_2, \ldots, i_d)$

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

Model Rank

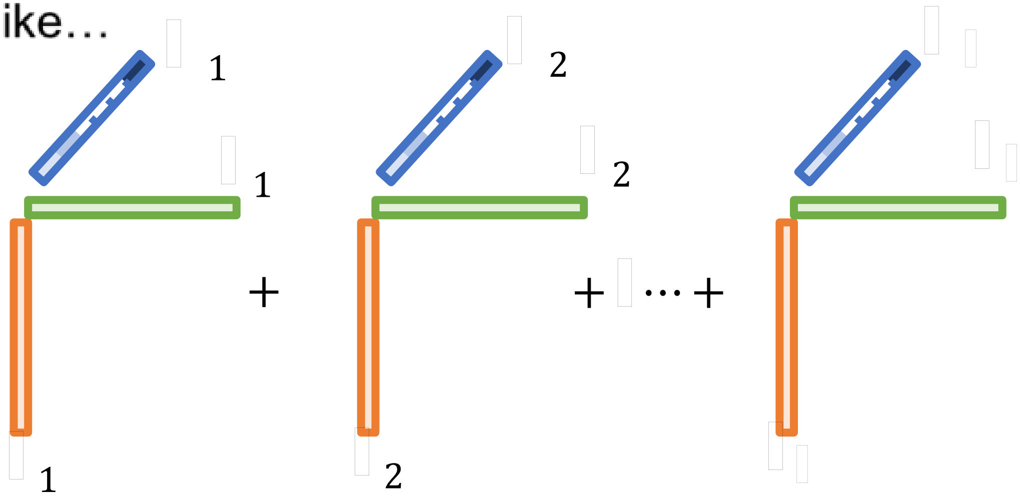
Streaming Tensors – 3-way Case

At each time step t, a new 2-way slice is added



$\approx$

If we assume factor matrices $\mathbf{A}$ and $\mathbf{B}$ *fixed through time*, then the ideal factorization looks like…



Add matrix each time step:

$$\mathbf{X}^{(t)} \in \mathbb{R}^{N_1 \times N_2}$$

Entirety is 3-way tensor:

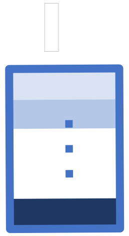
$$\mathfrak{X} \in \mathbb{R}^{N_1 \times N_2 \times T}$$

$$\mathfrak{X} = \sum_{j=1}^{R} \mathbf{a}_j \circ \mathbf{b}_j \circ \mathbf{w}_j = [\![\mathbf{A}, \mathbf{B}, \mathbf{W}]\!]$$

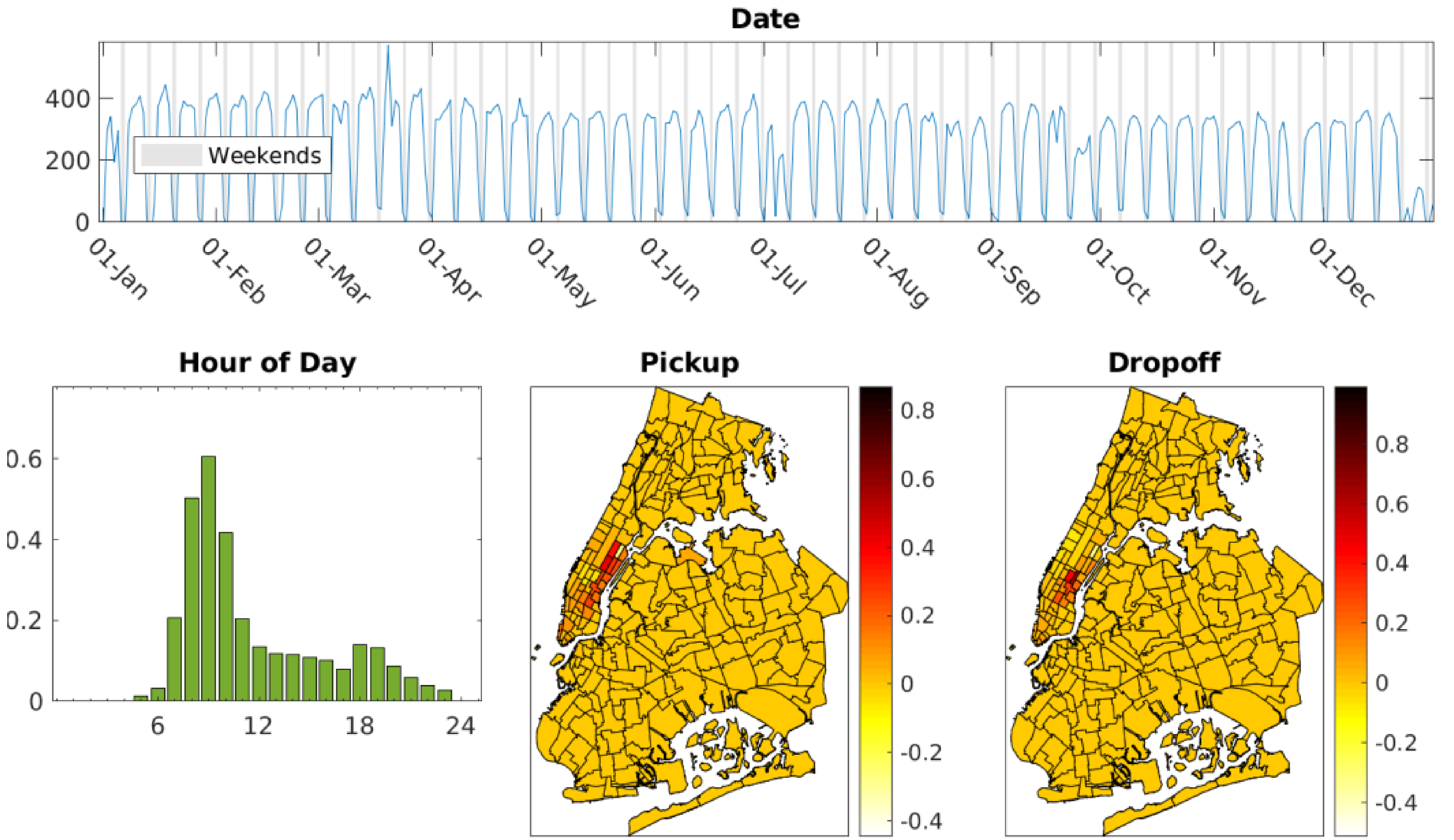
$$\mathbf{A} = \left[\mathbf{a}_1 \ \mathbf{a}_2 \ \cdots \ \mathbf{a}_R \right] \in \mathbb{R}^{N_1 \times R}$$

$$\mathbf{B} = \left[\mathbf{b}_1 \ \mathbf{b}_2 \ \cdots \ \mathbf{b}_R \right] \in \mathbb{R}^{N_2 \times R}$$

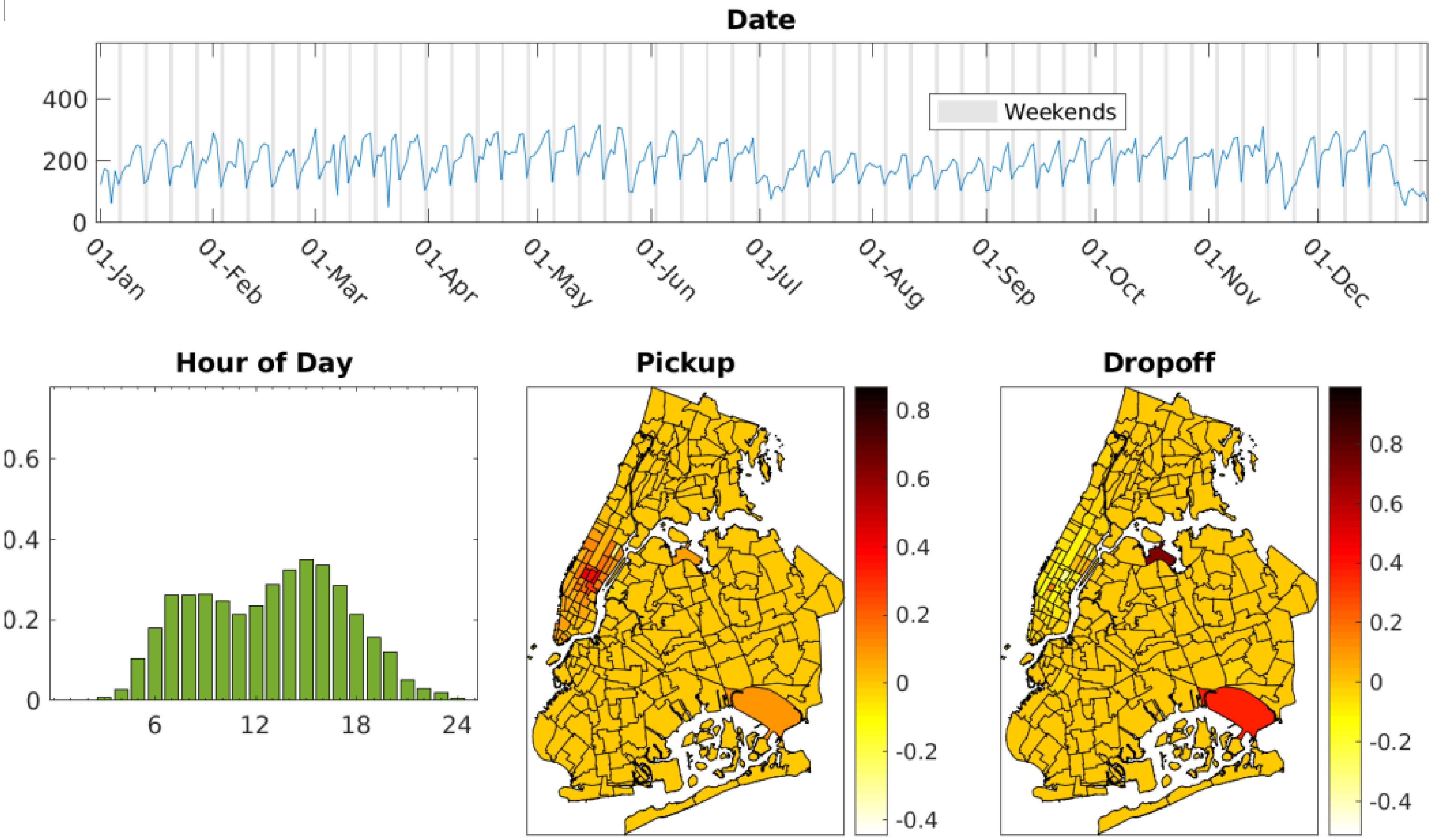
$$\mathbf{W} = \left[\mathbf{w}_1 \ \mathbf{w}_2 \ \cdots \ \mathbf{w}_R \right] \in \mathbb{R}^{T \times R}$$



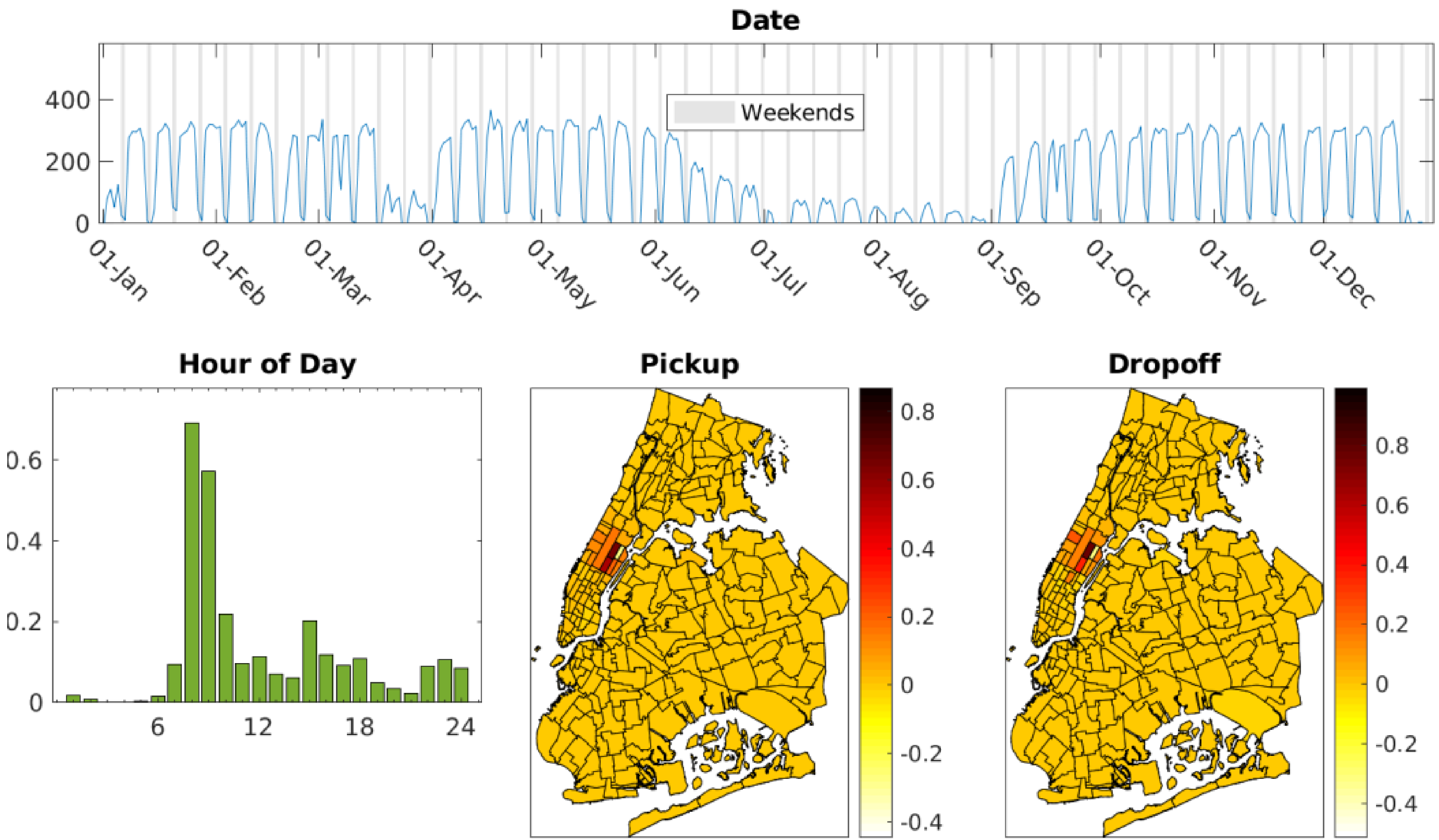
Component #4 (of 50): Morning Commute to Rockefeller Center



Component #17 (of 50): Travel to JFK and La Guardia Airports

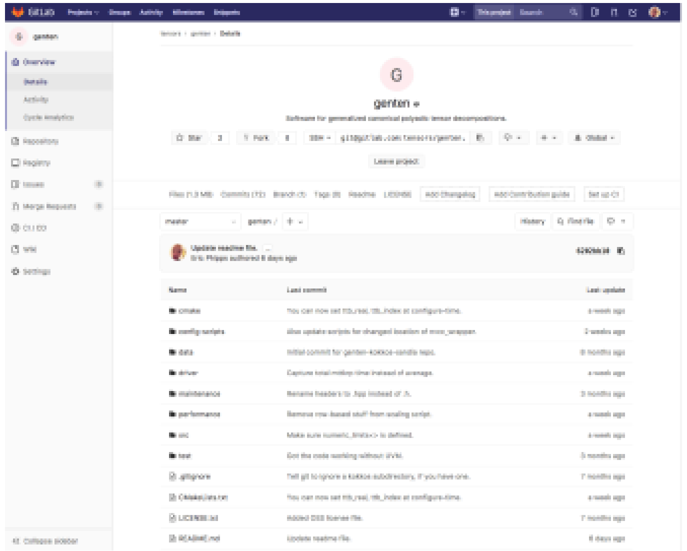


Component #20 (of 50): School Morning Dropoff



GenTen : Software for Generalized Canonical Polyadic Tensor Decompositions

New software package GenTen developed at SNL
- E. Phipps, T. Kolda, D. Dunlavy, G. Ballard, T. Plantenga
- Based on C++ port of Matlab Tensor Toolbox
- Publicly available at https://gitlab.com/tensors/genten
- Implements full CP-ALS algorithm for sparse (and dense) tensors, as well as GCP algorithm for sparse tensors

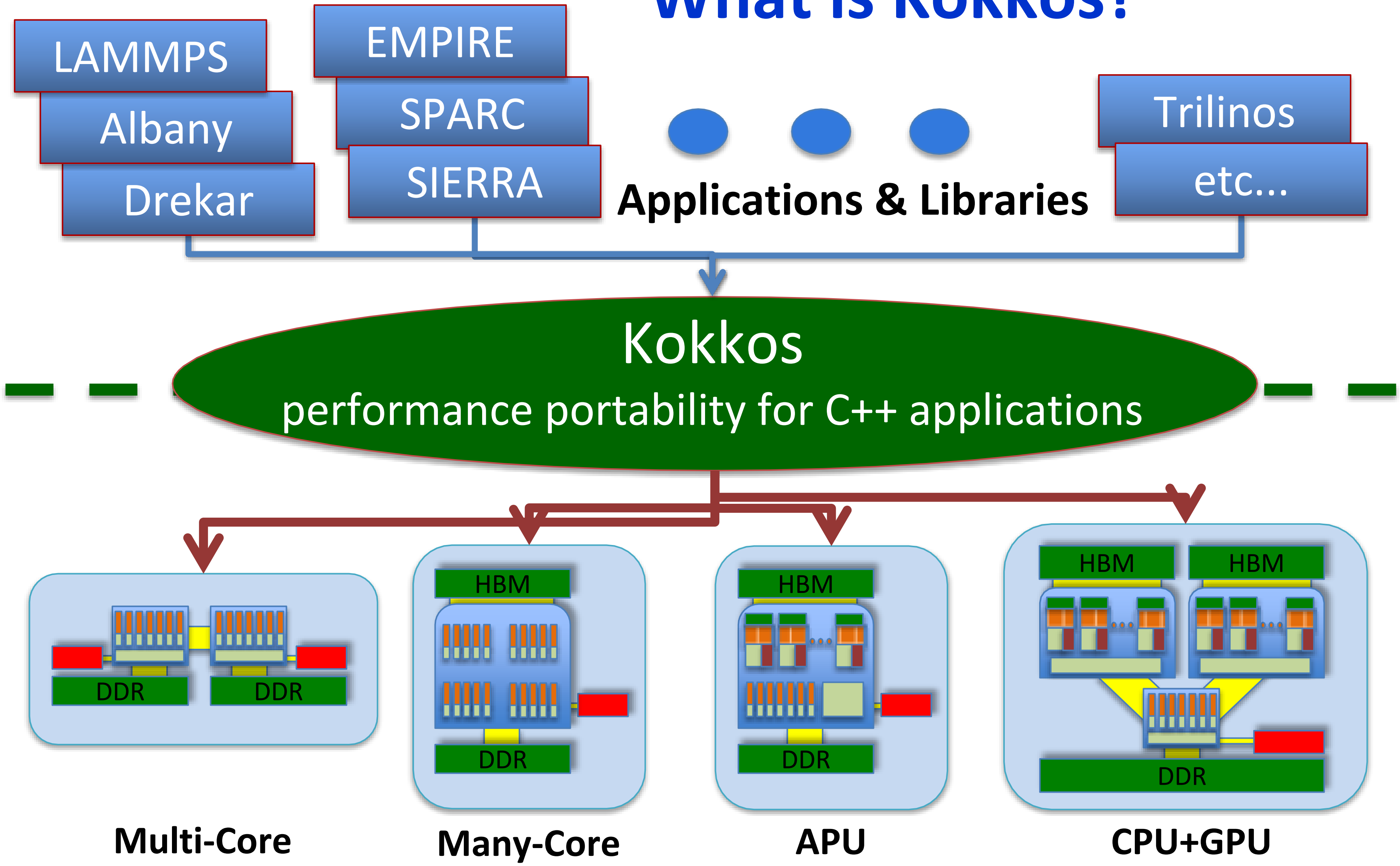


Incorporates shared memory parallelism for emerging manycore hardware using **Kokkos**
- Multicore CPUs via OpenMP, pThreads
- GPUs via Nvidia Cuda (Intel and AMD coming soon)
- Intel Xeon Phi (a.k.a. KNC/KNL) via OpenMP

Implements parallelism for all performance-critical operations
- **MTTKRP**, tensor inner product, norms, ...
- Can use optimized third-party libraries (MKL, cuBLAS, ...)
- Natively handles data transfers between CPU, GPU memory

Callable from Matlab Tensor Toolbox!

What is Kokkos?



C. Trott, *et al.*, https://github.com/kokkos/kokkos