POLYTECHNIQUE MONTRÉAL

UNIVERSITÉ D'INGÉNIERIE

IVADO

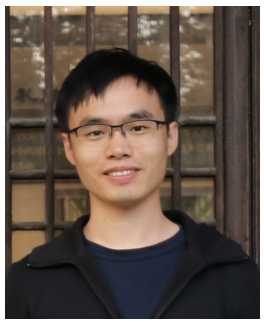
Matrix and Tensor Models for Spatiotemporal Traffic Data Imputation and Forecasting

Ph.D. Defense

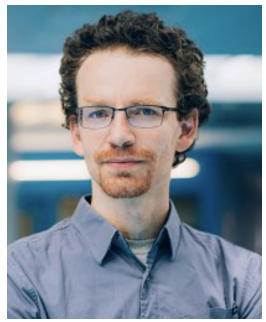
Xinyu Chen

Polytechnique Montreal, Canada

December 11, 2023



Ph.D. Candidate
Xinyu Chen



Supervisor
Prof. Nicolas Saunier

Co-supervisor
Prof. Lijun Sun (McGill)

Outline

- **Background**
- **Literature Review**
  - Tensor Factorization
  - Tensor Factorization (TF)
- **Nonstationary Temporal Matrix Factorization**
- **Low-Rank Autoregressive Tensor Completion**
- **Laplacian Convolutional Representation**
  - Motivation
  - Reformulate Laplacian Regularization
  - Traffic Time Series Imputation
- **Hankel Tensor Factorization**
  - Motivation
  - Hankel Structure
  - Hankel Structure
- **Experiments**
- **Conclusion**

Our studies: (Pattern discovery, traffic data imputation/forecasting)

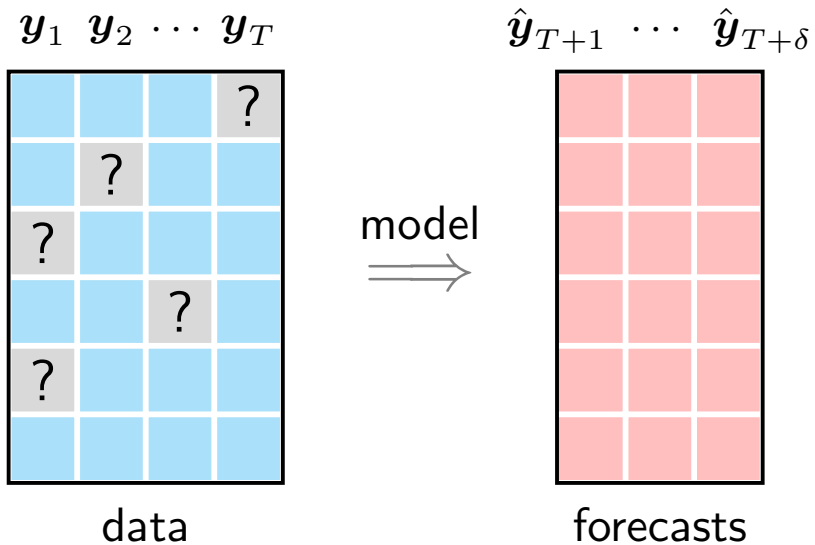
1. X. Chen, C. Zhang, X. Chen, N. Saunier, L. Sun (2023). Discovering dynamic patterns from spatiotemporal data with time-varying low-rank autoregression. IEEE Transactions on Knowledge and Data Engineering. Early access.
2. X. Chen, Z. Cheng, N. Saunier, L. Sun (2022). Laplacian convolutional representation for traffic time series imputation. arXiv preprint arXiv:2212.01529.
3. X. Chen, L. Sun (2022). Bayesian temporal factorization for multidimensional time series prediction. IEEE Transactions on Pattern Analysis and Machine Intelligence. 44 (9): 4659-4673.

GitHub repository:

- **transdim**: Machine learning for spatiotemporal traffic data imputation and forecasting. (1,000 stars & 270 forks on GitHub)
  https://github.com/xinychen/transdim

Problem Formulation

- Given a partially observed traffic data matrix $\boldsymbol{Y} \in \mathbb{R}^{N \times T}$ consisting of time series $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_T \in \mathbb{R}^N$, how to perform imputation and forecasting on these data?



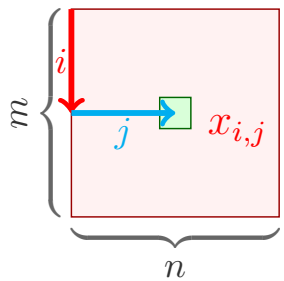
[Q]

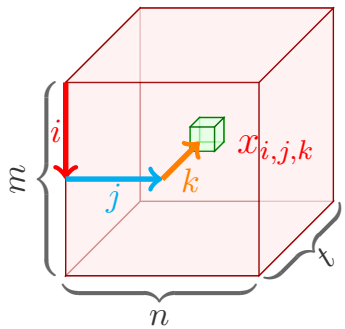
- How to impute missing values in $\boldsymbol{Y}$?
- How to forecast the data points $\boldsymbol{y}_{T+\delta}, \delta \in \mathbb{N}^+$?

Whole Picture

Tensors

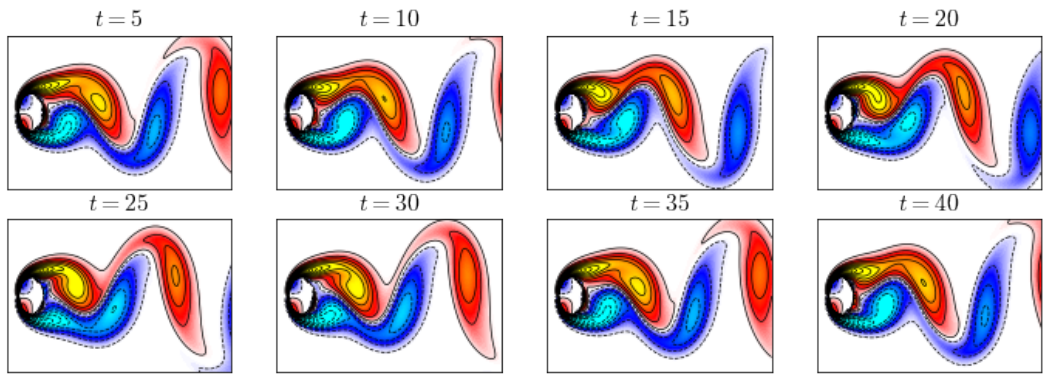
- What is tensor? $\boldsymbol{X} \in \mathbb{R}^{m \times n}$ vs. $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{m \times n \times t}$





- Tensors are everywhere!

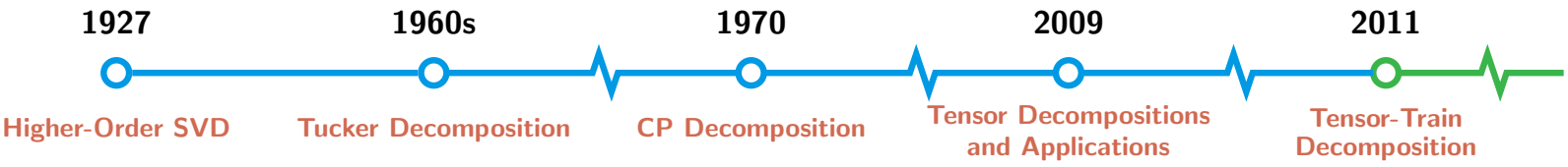
Color image with RGB channels



Dynamical system (fluid flow)

Tensor Factorization

- Revisit tensor factorization (TF)



Frank Lauren Hitchcock

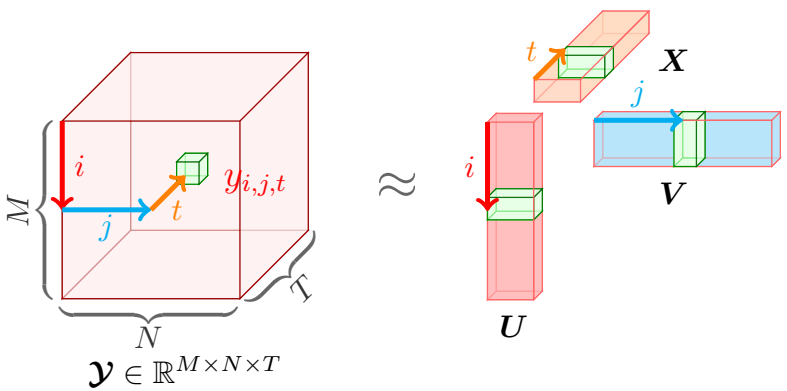
Ledyard R. Tucker

J. Douglas Carroll
Jih-Jie Chang
Richard A. Harshman

Tamara G. Kolda

Ivan Oseledets

- **CP tensor factorization**: Factorize $\mathcal{Y}$ into the combination of three rank-R factor matrices (i.e., low-dimensional latent factors).



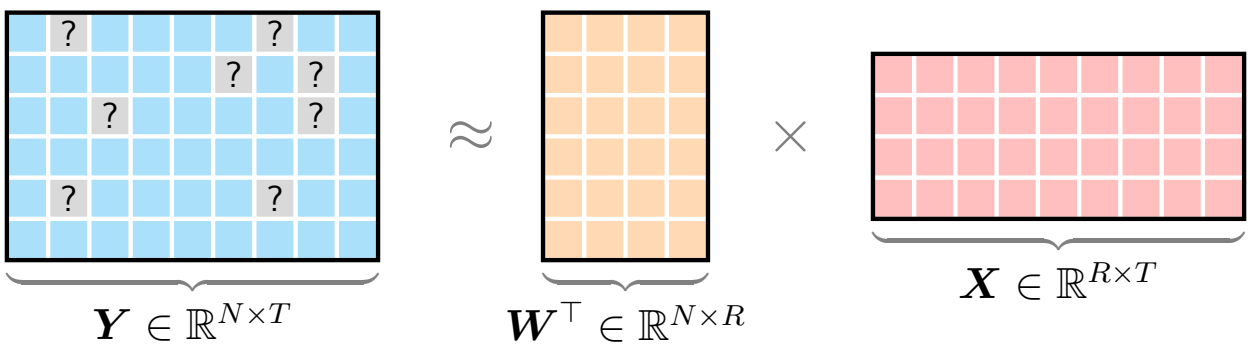
$$\begin{cases} y_{i,j,t} \approx \sum_{r=1}^{R} u_{i,r} v_{j,r} x_{t,r} \\ \mathcal{Y} \approx \sum_{r=1}^{R} \boldsymbol{u}_r \otimes \boldsymbol{v}_r \otimes \boldsymbol{x}_r \end{cases}$$

Matrix Factorization

Low-Rank Matrix Factorization (Koren et al.'09)

Given any partially observed data matrix $\boldsymbol{Y} \in \mathbb{R}^{N \times T}$ with observed index set Ω, then the optimization problem of matrix factorization with rank R can formulated as

$$\min_{\boldsymbol{W}, \boldsymbol{X}} \frac{1}{2} \|\mathcal{P}_{\Omega}(\boldsymbol{Y} - \boldsymbol{W}^{\top} \boldsymbol{X})\|_F^2 + \frac{\rho}{2} (\|\boldsymbol{W}\|_F^2 + \|\boldsymbol{X}\|_F^2)$$



✗ Cannot capture temporal correlations.

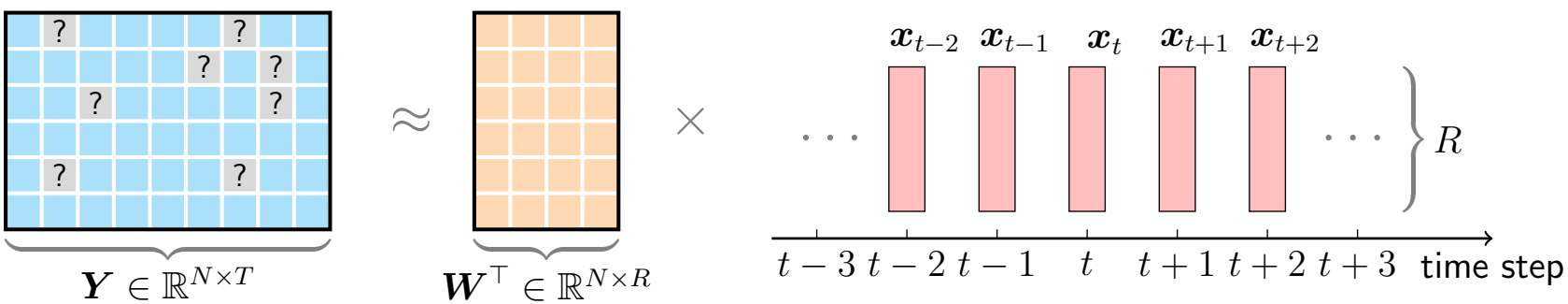
✗ Cannot perform time series forecasting.

Temporal Matrix Factorization

Temporal Matrix Factorization (Yu et al.'16; Chen & Sun'21)

Given any partially observed time series data $\boldsymbol{Y} \in \mathbb{R}^{N \times T}$ with observed index set Ω, then temporal matrix factorization assumes a dth-order vector autoregressive (VAR) process on the temporal factor matrix:

$$\min_{\boldsymbol{W}, \boldsymbol{X}, \{\boldsymbol{A}_k\}_{k=1}^{d}} \frac{1}{2}\|\mathcal{P}_{\Omega}(\boldsymbol{Y} - \boldsymbol{W}^{\top}\boldsymbol{X})\|_F^2 + \frac{\rho}{2}(\|\boldsymbol{W}\|_F^2 + \|\boldsymbol{X}\|_F^2) + \frac{\lambda}{2}\sum_{t=d+1}^{T}\left\|\boldsymbol{x}_t - \sum_{k=1}^{d}\boldsymbol{A}_k\boldsymbol{x}_{t-k}\right\|_2^2$$



✗ VAR is usually built on stationary time series (temporal factors).

Methodology

Nonstationary temporal matrix factorization (NoTMF)

Given any partially observed time series data $\boldsymbol{Y} \in \mathbb{R}^{N \times T}$ with observed index set Ω, then we assume a season-m differencing on the latent temporal factors:

$$\begin{aligned}\min_{\boldsymbol{W},\boldsymbol{X},\{\boldsymbol{A}_k\}_{k=1}^{d}} \quad & \frac{1}{2}\|\mathcal{P}_{\Omega}(\boldsymbol{Y}-\boldsymbol{W}^{\top}\boldsymbol{X})\|_F^2+\frac{\rho}{2}(\|\boldsymbol{W}\|_F^2+\|\boldsymbol{X}\|_F^2) \\ & +\frac{\lambda}{2}\sum_{t=d+m+1}^{T}\left\|(\boldsymbol{x}_t-\boldsymbol{x}_{t-m})-\sum_{k=1}^{d}\boldsymbol{A}_k(\boldsymbol{x}_{t-k}-\boldsymbol{x}_{t-m-k})\right\|_2^2\end{aligned}$$

- First-order differencing $\boldsymbol{x}_t' = \boldsymbol{x}_t - \boldsymbol{x}_{t-1}$.
- Second-order differencing $\boldsymbol{x}_t'' = (\boldsymbol{x}_t - \boldsymbol{x}_{t-1}) - (\boldsymbol{x}_{t-1} - \boldsymbol{x}_{t-2})$.
- Twice-differenced series $\boldsymbol{x}_t'' = (\boldsymbol{x}_t - \boldsymbol{x}_{t-m}) - (\boldsymbol{x}_{t-1} - \boldsymbol{x}_{t-m-1})$.

✓ Stationarizing a time series with differencing can improve the prediction.[1]

[1] Stationarity and differencing: `https://otexts.com/fpp2/stationarity.html`

Nonstationary Temporal Matrix Factorization

Rewrite NoTMF:

$$\min_{\boldsymbol{W},\boldsymbol{X},\boldsymbol{A}} \frac{1}{2}\|\mathcal{P}_{\Omega}(\boldsymbol{Y}-\boldsymbol{W}^{\top}\boldsymbol{X})\|_F^2+\frac{\rho}{2}(\|\boldsymbol{W}\|_F^2+\|\boldsymbol{X}\|_F^2)$$
$$+\frac{\lambda}{2}\|\boldsymbol{X}\boldsymbol{\Psi}_0^{\top}-\boldsymbol{A}(\boldsymbol{I}_d\otimes\boldsymbol{X})\boldsymbol{\Psi}^{\top}\|_F^2$$

Alternating minimization method:

- w.r.t. $\boldsymbol{W}$:

$$\frac{\partial f}{\partial \boldsymbol{W}}=-\boldsymbol{X}\mathcal{P}_{\Omega}^{\top}(\boldsymbol{Y}-\boldsymbol{W}^{\top}\boldsymbol{X})+\rho\boldsymbol{W}=\boldsymbol{0}$$

- w.r.t. $\boldsymbol{X}$:

$$\frac{\partial f}{\partial \boldsymbol{X}}=-\boldsymbol{W}\mathcal{P}_{\Omega}(\boldsymbol{Y}-\boldsymbol{W}^{\top}\boldsymbol{X})+\rho\boldsymbol{X}+\lambda\sum_{k=0}^{d}\boldsymbol{A}_k^{\top}\left(\sum_{h=0}^{d}\boldsymbol{A}_h\boldsymbol{X}\boldsymbol{\Psi}_h^{\top}\right)\boldsymbol{\Psi}_k=\boldsymbol{0}$$

- w.r.t. $\boldsymbol{A}$:

$$\boldsymbol{A}=\boldsymbol{X}\boldsymbol{\Psi}_0^{\top}[(\boldsymbol{I}_d\otimes\boldsymbol{X})\boldsymbol{\Psi}^{\top}]^{\dagger}$$

Nonstationary Temporal Matrix Factorization

Rewrite VAR in the form of matrix

Temporal operators

For any multivariate time series $\boldsymbol{X} \in \mathbb{R}^{R\times T}$ with $m, d \in \mathbb{N}^+$, if we define temporal operators as

$$\begin{aligned}\boldsymbol{\Psi}_k \triangleq & \begin{bmatrix}\boldsymbol{0}_{(T-d-m)\times(d-k)} & -\boldsymbol{I}_{T-d-m} & \boldsymbol{0}_{(T-d-m)\times(k+m)}\end{bmatrix} \\ & + \begin{bmatrix}\boldsymbol{0}_{(T-d-m)\times(d+m-k)} & \boldsymbol{I}_{T-d-m} & \boldsymbol{0}_{(T-d-m)\times k}\end{bmatrix} \\ & \in \mathbb{R}^{(T-d-m)\times T},\ k = 0, 1, \ldots, d\end{aligned}$$

then

$$\sum_{t=d+m+1}^{T} \left\| (\boldsymbol{x}_t - \boldsymbol{x}_{t-m}) - \sum_{k=1}^{d} \boldsymbol{A}_k (\boldsymbol{x}_{t-k} - \boldsymbol{x}_{t-m-k}) \right\|_2^2$$

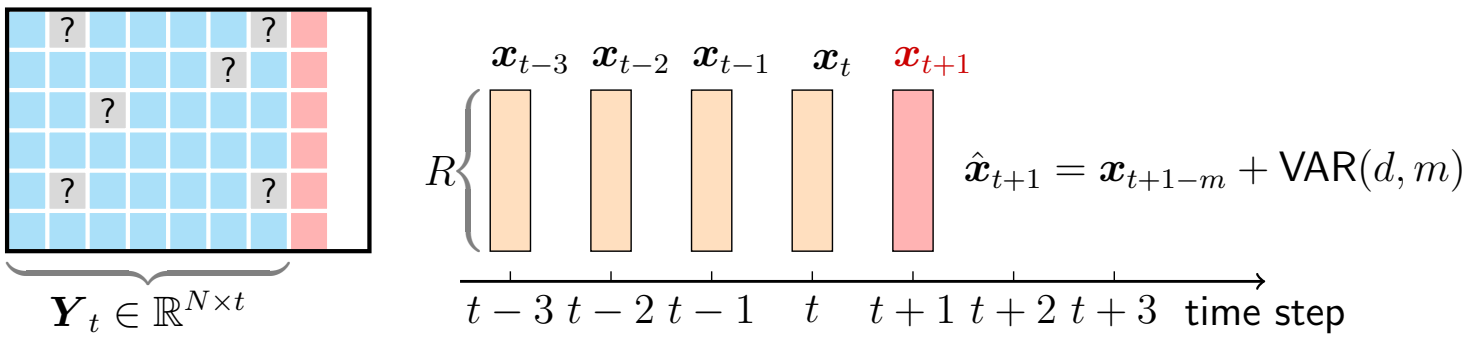
$$\equiv \| \boldsymbol{X}\boldsymbol{\Psi}_0^\top - \sum_{k=1}^{d} \boldsymbol{A}_k \boldsymbol{X}\boldsymbol{\Psi}_k^\top \|_F^2 \triangleq \| \boldsymbol{X}\boldsymbol{\Psi}_0^\top - \boldsymbol{A}(\boldsymbol{I}_d \otimes \boldsymbol{X})\boldsymbol{\Psi}^\top \|_F^2$$

where $\boldsymbol{A} \triangleq \begin{bmatrix}\boldsymbol{A}_1 & \cdots & \boldsymbol{A}_d\end{bmatrix}$ and $\boldsymbol{\Psi} \triangleq \begin{bmatrix}\boldsymbol{\Psi}_1 & \cdots & \boldsymbol{\Psi}_d\end{bmatrix}$.

Nonstationary Temporal Matrix Factorization

NoTMF forecasting on streaming data?

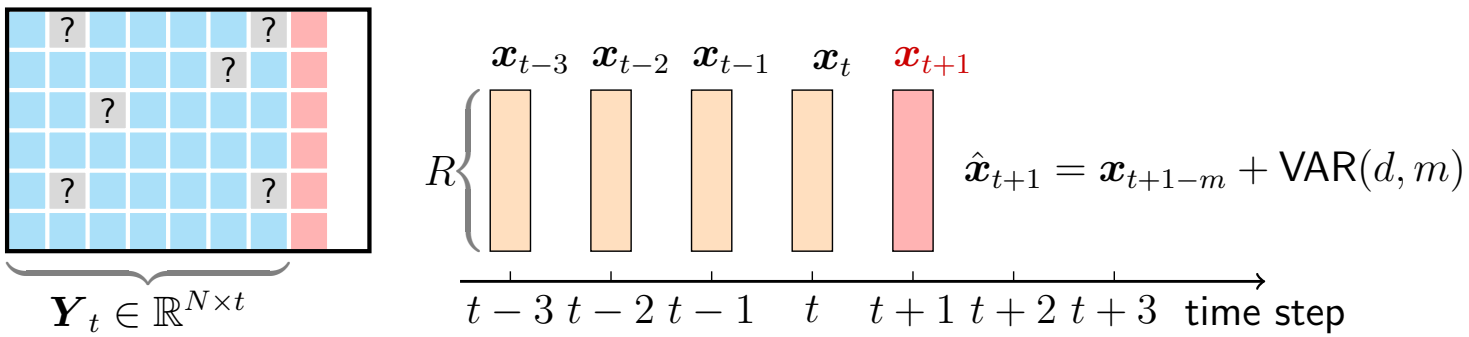
- NoTMF: Use $\boldsymbol{Y}_t$ to estimate $\{\boldsymbol{W}, \boldsymbol{X}, \boldsymbol{A}\}$.



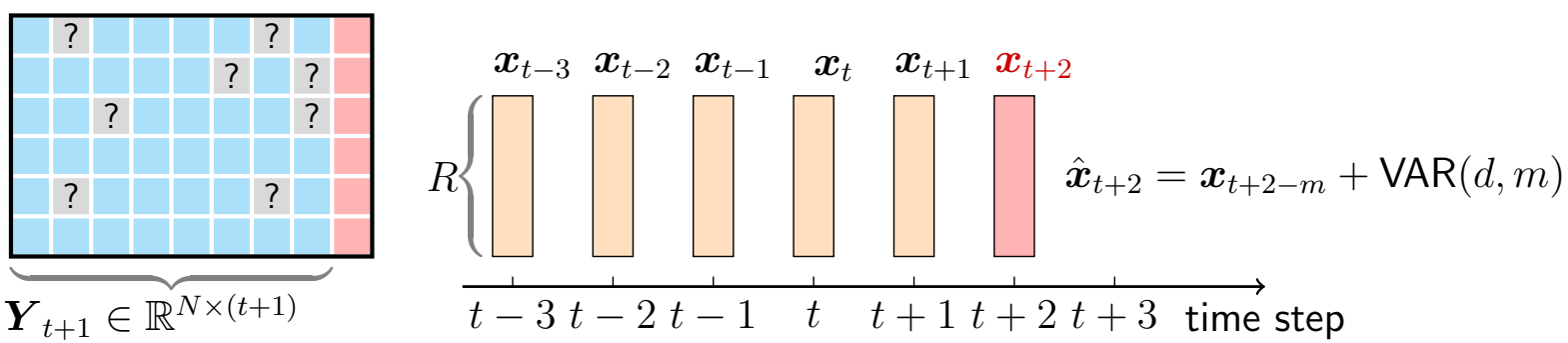
Nonstationary Temporal Matrix Factorization

NoTMF forecasting on streaming data?

- NoTMF: Use $\boldsymbol{Y}_t$ to estimate $\{\boldsymbol{W}, \boldsymbol{X}, \boldsymbol{A}\}$.



- Online forecasting (Gultekin & Paisley'18): Fix $\boldsymbol{W}$ and use $\boldsymbol{Y}_{t+1}$ to update $\{\boldsymbol{X}, \boldsymbol{A}\}$.

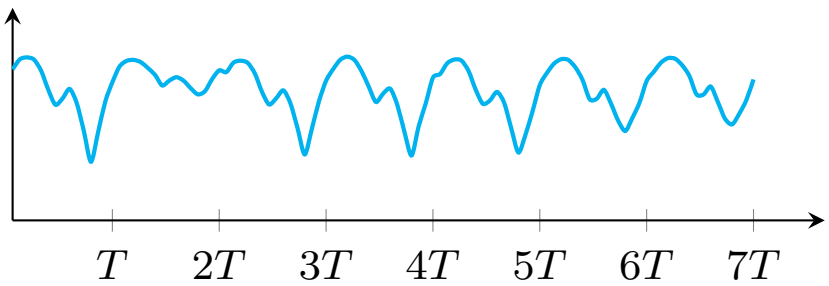


LATC

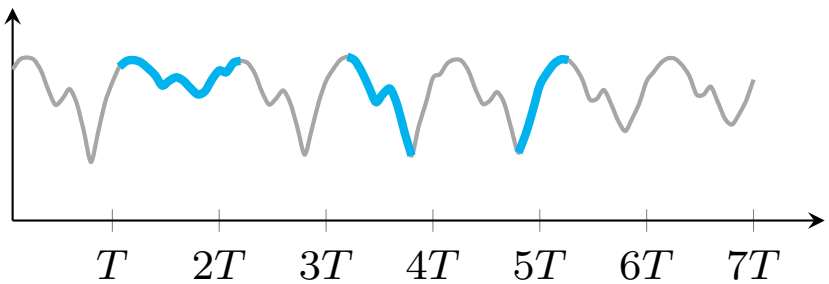
Laplacian Convolutional Representation

Motivation: Time series imputation

- Global trends (e.g., long-term quasi-seasonality & daily/weekly rhythm):



- Local trends (e.g., short-term time series trends):

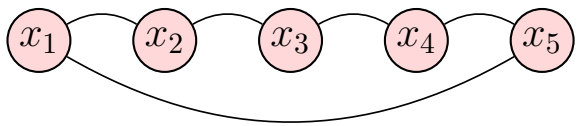


- **[Question]** How to characterize both global and local trends in sparse time series data?

Laplacian Convolutional Representation

[Local trend modeling] Reformulate temporal regularization with circular convolution.

- Intuition of (circulant) Laplacian matrix.



Undirected and circulant graph

Modeling $\longrightarrow$

$$\boldsymbol{L} = \begin{bmatrix} 2 & -1 & 0 & 0 & -1 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ 0 & 0 & -1 & 2 & -1 \\ -1 & 0 & 0 & -1 & 2 \end{bmatrix}$$

(Circulant) Laplacian matrix

- Define Laplacian kernel:

$$\boldsymbol{\ell} \triangleq (2, -1, 0, 0, -1)^\top$$

$$\Downarrow$$

$$\boldsymbol{\ell} \triangleq (\underbrace{2\tau}_{\text{degree}}, \underbrace{-1, \cdots, -1}_{\tau}, 0, \cdots, 0, \underbrace{-1, \cdots, -1}_{\tau})^\top \in \mathbb{R}^T$$

for any time series $\boldsymbol{x} = (x_1, \ldots, x_T)^\top \in \mathbb{R}^T$.

- (Laplacian) Temporal regularization:

$$\mathcal{R}_\tau(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{L}\boldsymbol{x}\|_2^2 = \frac{1}{2}\|\boldsymbol{\ell} \star \boldsymbol{x}\|_2^2$$

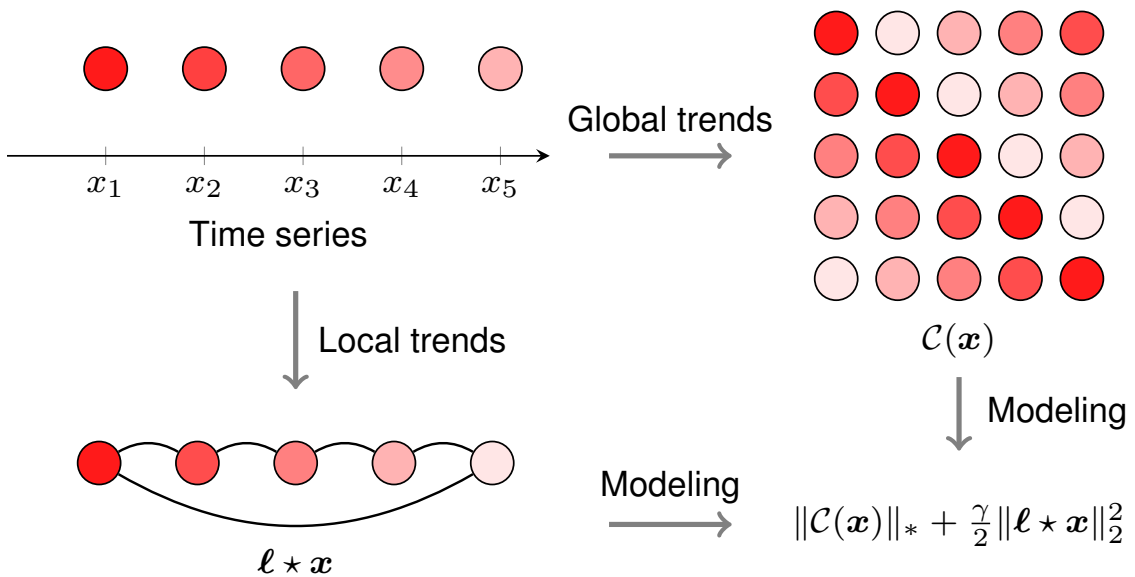
Laplacian Convolutional Representation

Laplacian Convolutional Representation (LCR)

For any partially observed time series $\boldsymbol{y} \in \mathbb{R}^T$ with observed index set Ω, LCR utilizes circulant matrix and Laplacian kernel to characterize **global and local trends** in time series, respectively, i.e.,

$$\begin{aligned} \min_{\boldsymbol{x}} \ & \|\mathcal{C}(\boldsymbol{x})\|_* + \frac{\gamma}{2}\|\boldsymbol{\ell} \star \boldsymbol{x}\|_2^2 \\ \text{s.t.} \ & \|\mathcal{P}_\Omega(\boldsymbol{x} - \boldsymbol{y})\|_2 \leq \epsilon \end{aligned}$$

where $\mathcal{C} : \mathbb{R}^T \to \mathbb{R}^{T \times T}$ denotes the circulant operator. $\|\cdot\|_*$ denotes the nuclear norm of matrix, namely, the sum of singular values.



Laplacian Convolutional Representation

- Augmented Lagrangian function:

$$\mathcal{L}(\boldsymbol{x},\boldsymbol{z},\boldsymbol{w}) = \|\mathcal{C}(\boldsymbol{x})\|_* + \frac{\gamma}{2}\|\boldsymbol{\ell}\star\boldsymbol{x}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{x}-\boldsymbol{z}\|_2^2 + \langle\boldsymbol{w},\boldsymbol{x}-\boldsymbol{z}\rangle + \frac{\eta}{2}\|\mathcal{P}_\Omega(\boldsymbol{z}-\boldsymbol{y})\|_2^2$$

  where $\boldsymbol{w}\in\mathbb{R}^T$ is the Lagrange multiplier, and $\langle\cdot,\cdot\rangle$ denotes the inner product.

- The ADMM scheme:

$$\begin{cases} \boldsymbol{x} := \arg\min\limits_{\boldsymbol{x}} \ \mathcal{L}(\boldsymbol{x},\boldsymbol{z},\boldsymbol{w}) & \text{(Nuclear norm minimization)} \\ \boldsymbol{z} := \arg\min\limits_{\boldsymbol{z}} \ \mathcal{L}(\boldsymbol{x},\boldsymbol{z},\boldsymbol{w}) & \text{(Closed-form solution)} \\ \quad = \dfrac{1}{\lambda+\eta}\mathcal{P}_\Omega(\lambda\boldsymbol{x}+\boldsymbol{w}+\eta\boldsymbol{y}) + \dfrac{1}{\lambda}\mathcal{P}_\Omega^\perp(\lambda\boldsymbol{x}+\boldsymbol{w}) & \\ \boldsymbol{w} := \boldsymbol{w}+\lambda(\boldsymbol{x}-\boldsymbol{z}) & \text{(Standard update)} \end{cases}$$

- Optimize $\boldsymbol{x}$?

$$\|\mathcal{C}(\boldsymbol{x})\|_* = \|\mathcal{F}(\boldsymbol{x})\|_1 \qquad \& \qquad \frac{1}{2}\|\boldsymbol{\ell}\star\boldsymbol{x}\|_2^2 = \frac{1}{2T}\|\mathcal{F}(\boldsymbol{\ell})\circ\mathcal{F}(\boldsymbol{x})\|_2^2$$

  Nuclear norm minimization $\Rightarrow$ ℓ_1-**norm minimization with FFT** (in $\mathcal{O}(T\log T)$ time).

Laplacian Convolutional Representation

- Optimize $\boldsymbol{x}$ via FFT (in $\mathcal{O}(T \log T)$ time):

$$\boldsymbol{x} := \arg\min_{\boldsymbol{x}} \ \|\mathcal{C}(\boldsymbol{x})\|_* + \frac{\gamma}{2}\|\boldsymbol{\ell} \star \boldsymbol{x}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{x} - \boldsymbol{z} + \boldsymbol{w}/\lambda\|_2^2$$

$$\Longrightarrow \hat{\boldsymbol{x}} := \arg\min_{\hat{\boldsymbol{x}}} \ \|\hat{\boldsymbol{x}}\|_1 + \frac{\gamma}{2T}\|\hat{\boldsymbol{\ell}} \circ \hat{\boldsymbol{x}}\|_2^2 + \frac{\lambda}{2T}\|\hat{\boldsymbol{x}} - \hat{\boldsymbol{z}} + \hat{\boldsymbol{w}}/\lambda\|_2^2$$

where we introduce $\{\hat{\boldsymbol{\ell}}, \hat{\boldsymbol{x}}, \hat{\boldsymbol{z}}, \hat{\boldsymbol{w}}\} \triangleq \mathcal{F}\{\boldsymbol{\ell}, \boldsymbol{x}, \boldsymbol{z}, \boldsymbol{w}\}$ (i.e., FFT).

ℓ_1-norm Minimization in Complex Space (Liu & Zhang'23)

For any optimization problem in the form of ℓ_1-norm minimization in complex space:

$$\min_{\hat{\boldsymbol{x}}} \ \|\hat{\boldsymbol{x}}\|_1 + \frac{\omega}{2}\|\hat{\boldsymbol{x}} - \hat{\boldsymbol{h}}\|_2^2$$

with complex-valued vectors $\hat{\boldsymbol{x}}, \hat{\boldsymbol{h}} \in \mathbb{C}^T$ and weight parameter ω, element-wise, the solution is given by

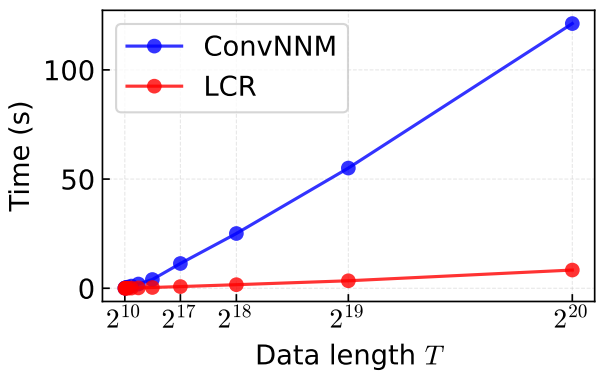
$$\hat{x}_t := \frac{\hat{h}_t}{|\hat{h}_t|} \cdot \max\{0, |\hat{h}_t| - 1/\omega\}, t = 1, \ldots, T.$$

Laplacian Convolutional Representation

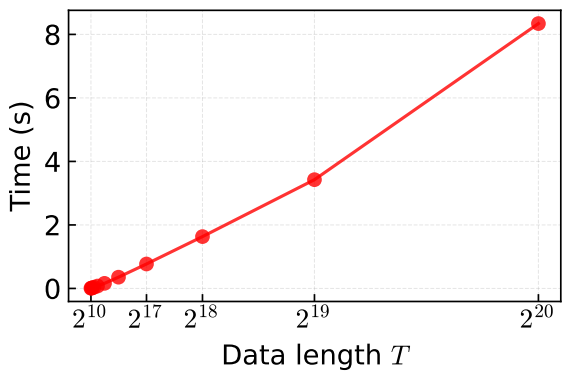
Empirical time complexity

On the synthetic data $\boldsymbol{y} \in \mathbb{R}^T$ with $T \in \{2^{10}, 2^{11}, \ldots, 2^{20}\}$

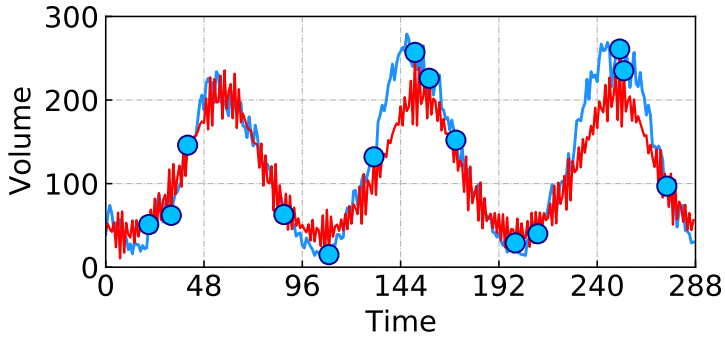
- Ours: **LCR**
  - An FFT implementation in $\mathcal{O}(T \log T)$
  - The logarithmic factor $\log T$ makes the FFT highly efficient
- Baseline: **ConvNNM** (Convolution nuclear norm minimization, Liu & Zhang'23)
  - Convolution matrix $\mathcal{C}_{\tilde{\tau}}(\boldsymbol{y}) \in \mathbb{R}^{T \times \tilde{\tau}}$ with kernel size $\tilde{\tau} \in \mathbb{N}^+$
  - Singular value thresholding in $\mathcal{O}(\tilde{\tau}^2 T)$



ConvNNM vs. LCR



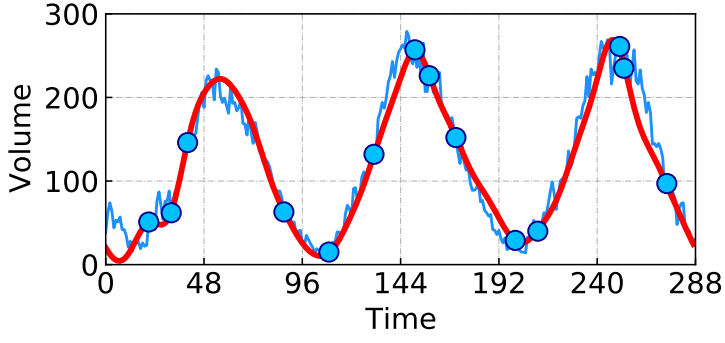
LCR



CircNNM:

$$\min_{\boldsymbol{x}} \ \|\mathcal{C}(\boldsymbol{x})\|_* \quad \text{s.t.} \ \|\mathcal{P}_\Omega(\boldsymbol{x}-\boldsymbol{y})\|_2 \le \epsilon$$

Plus temporal regularization (TR)

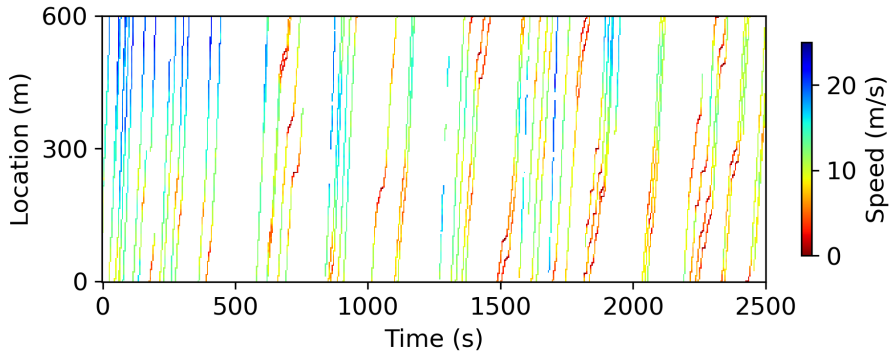


LCR:

$$\min_{\boldsymbol{x}} \ \|\mathcal{C}(\boldsymbol{x})\|_* + \frac{\gamma}{2}\|\boldsymbol{\ell} \star \boldsymbol{x}\|_2^2 \quad \text{s.t.} \ \|\mathcal{P}_\Omega(\boldsymbol{x}-\boldsymbol{y})\|_2 \le \epsilon$$

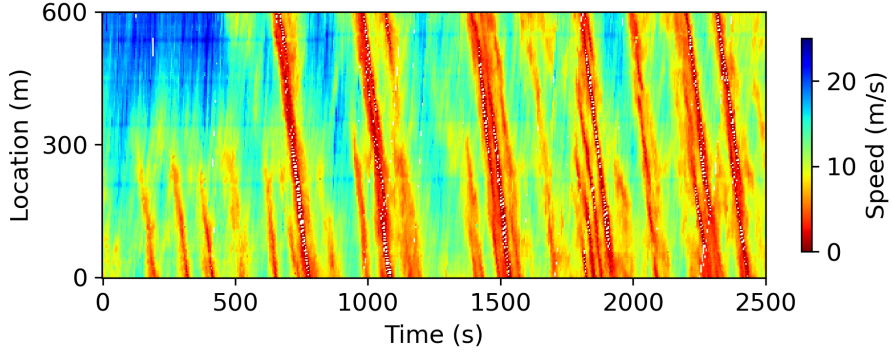
Motivation: Spatiotemporal data reconstruction

- Speed field reconstruction problem in vehicular traffic flow.



200-by-500 matrix (NGSIM)

Reconstruct speed field from 5% sparse trajectories?



- How to learn from sparse spatiotemporal data?
- How to characterize spatial/temporal dependencies?

Hankel Tensor Factorization

- Hankel matrix
  - Given $\boldsymbol{x} = (1, 2, 3, 4, 5)^\top$ and window length $\tau = 2$, we have

$$\mathcal{H}_\tau(\boldsymbol{x}) = \begin{bmatrix} 1 & 2 \\ 2 & 3 \\ 3 & 4 \\ 4 & 5 \end{bmatrix} \in \mathbb{R}^{4\times 2}$$

Hankel matrix (Source: Twitter)

Hankel Tensor Factorization

- Hankel matrix
  - On time series $\boldsymbol{y} = (y_1, y_2, \ldots, y_5)^\top$ with $\tau = 2$:

$$\mathcal{H}_\tau(\boldsymbol{y}) = \begin{bmatrix} y_1 & y_2 \\ y_2 & y_3 \\ y_3 & y_4 \\ y_4 & y_5 \end{bmatrix} \approx \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix} \otimes \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\Longrightarrow \quad \hat{\boldsymbol{y}} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \\ \hat{y}_4 \\ \hat{y}_5 \end{bmatrix} = \mathcal{H}_\tau^{-1}\left(\begin{bmatrix} v_1 x_1 & v_1 x_2 \\ v_2 x_1 & v_2 x_2 \\ v_3 x_1 & v_3 x_2 \\ v_4 x_1 & v_4 x_2 \end{bmatrix} \right) = \begin{bmatrix} v_1 x_1 \\ (v_1 x_2 + v_2 x_1)/2 \\ (v_2 x_2 + v_3 x_1)/2 \\ v_4 x_2 \end{bmatrix}$$

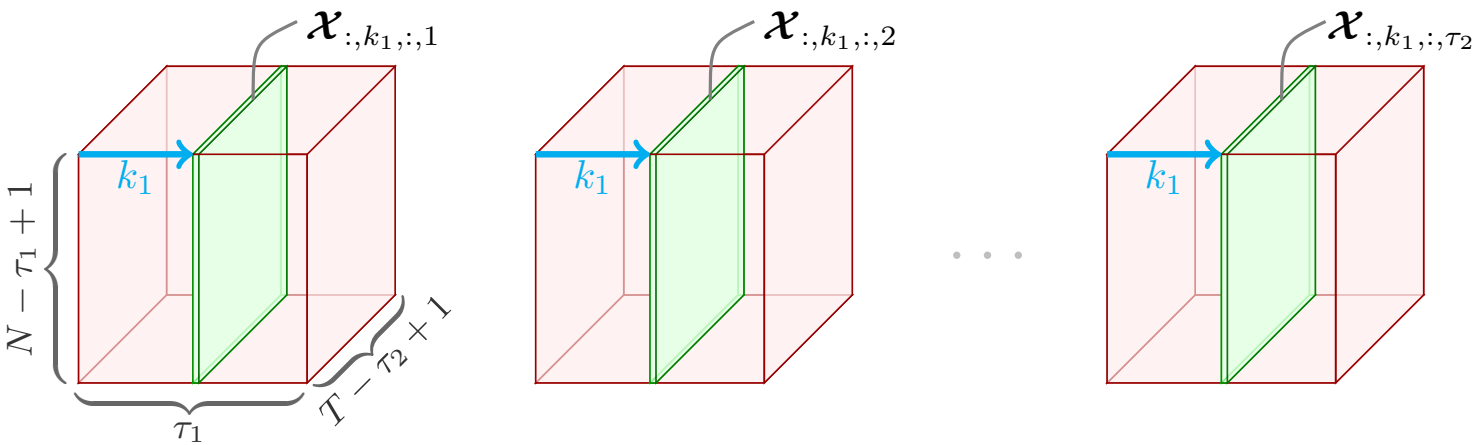
  - Automatic temporal modeling.

Hankel Tensor Factorization

- Hankel tensor: Given any matrix $\boldsymbol{X} \in \mathbb{R}^{N \times T}$, we have

$$\boldsymbol{\mathcal{X}} \triangleq \mathcal{H}_{\tau_1, \tau_2}(\boldsymbol{X})$$

  - Window lengths: $\tau_1, \tau_2 \in \mathbb{N}^+$;
  - Tensor size: $(N - \tau_1 + 1) \times \tau_1 \times (T - \tau_2 + 1) \times \tau_2$;



(Figure) 4th order Hankel tensor: A sequence of third-order tensors.

  - Slice: $\boldsymbol{\mathcal{X}}_{:,k_1,:,k_2}$, $\forall k_1, k_2$;
  - Slice size: $(N - \tau_1 + 1) \times (T - \tau_2 + 1)$.

Hankel Tensor Factorization

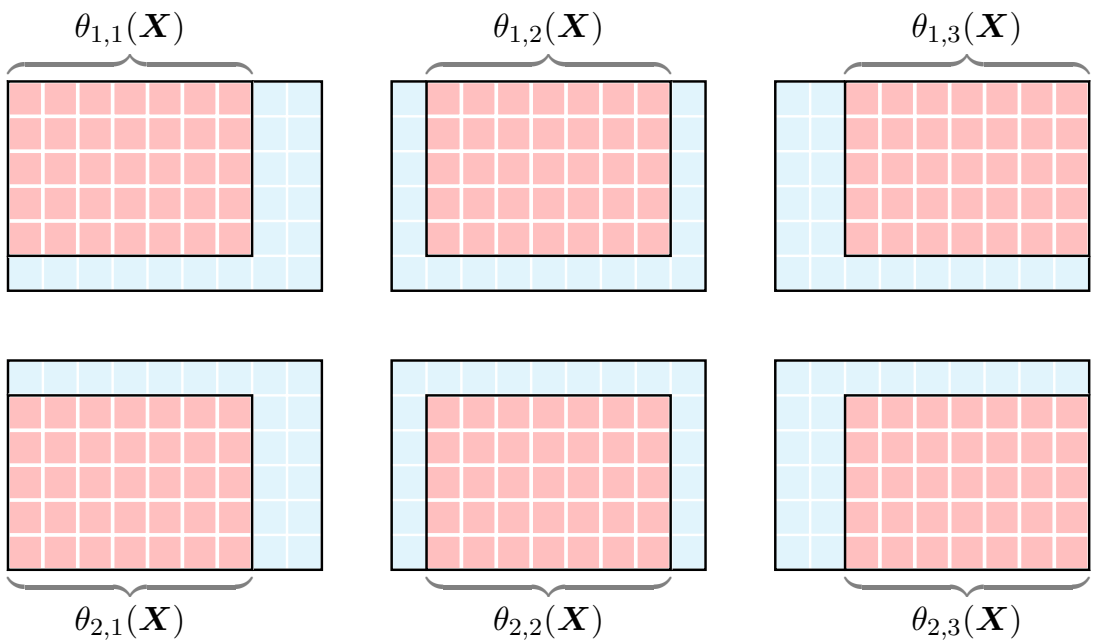
Hankel indexing:

- Sampling function for the Hankelization:

$$\theta_{k_1,k_2}(\boldsymbol{X}) \triangleq [\mathcal{H}_{\tau_1,\tau_2}(\boldsymbol{X})]_{:,k_1,:,k_2},$$

  referring to the tensor slice with $k_1 \in \{1,\ldots,\tau_1\}$, $k_2 \in \{1,\ldots,\tau_2\}$.

- **[Importance]** Developing memory-efficient algorithms.



- Tensor slices $\theta_{k_1,k_2}(\boldsymbol{X})$ vs. data matrix $\boldsymbol{X}$

Hankel Tensor Factorization

Our model:

- Convolutional tensor decomposition (circular convolution $\star_{\text{row}}$):

$$\theta_{k_1,k_2}(\boldsymbol{Y}) \approx (\boldsymbol{Q} \star_{\text{row}} \boldsymbol{s}_{k_1}^\top)(\boldsymbol{U} \star_{\text{row}} \boldsymbol{v}_{k_2}^\top)^\top$$

Baselines:

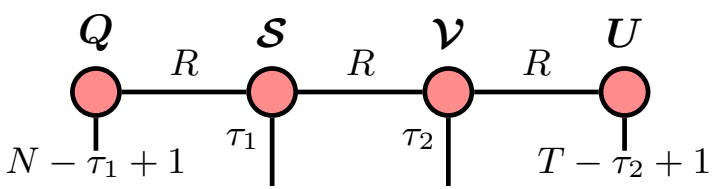
- CP tensor decomposition (Khatri-Rao product $\odot$):

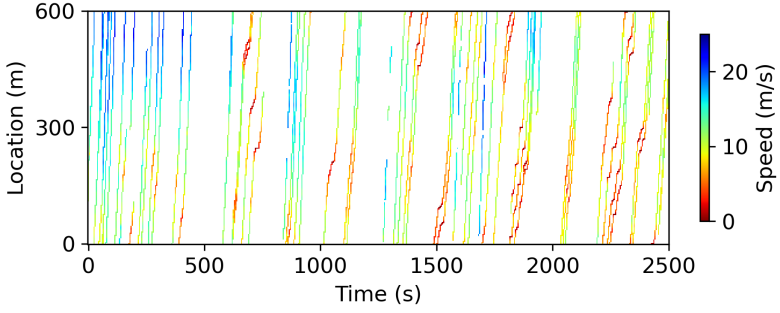
$$\theta_{k_1,k_2}(\boldsymbol{Y}) \approx (\boldsymbol{Q} \odot \boldsymbol{s}_{k_1}^\top)(\boldsymbol{U} \odot \boldsymbol{v}_{k_2}^\top)^\top$$

- Tensor-train decomposition:

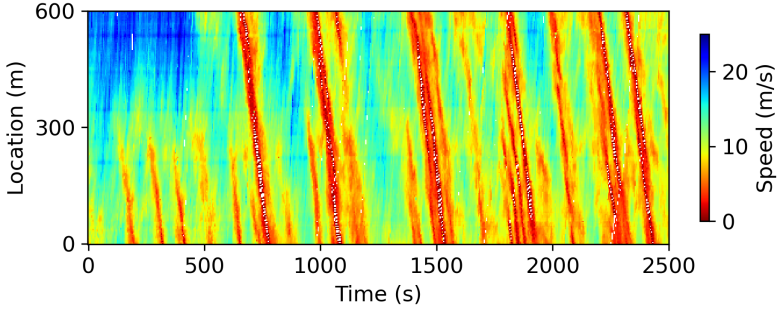
$$\theta_{k_1,k_2}(\boldsymbol{Y}) \approx (\boldsymbol{Q}\boldsymbol{S}_{k_1})(\boldsymbol{U}\boldsymbol{V}_{k_2})^\top$$

  - $\{\boldsymbol{S}_{k_1}, \boldsymbol{V}_{k_2}\}$ are **circulant matrices** $\Rightarrow$ convolutional decomposition
  - $\{\boldsymbol{S}_{k_1}, \boldsymbol{V}_{k_2}\}$ are **diagonal matrices** $\Rightarrow$ CP decomposition



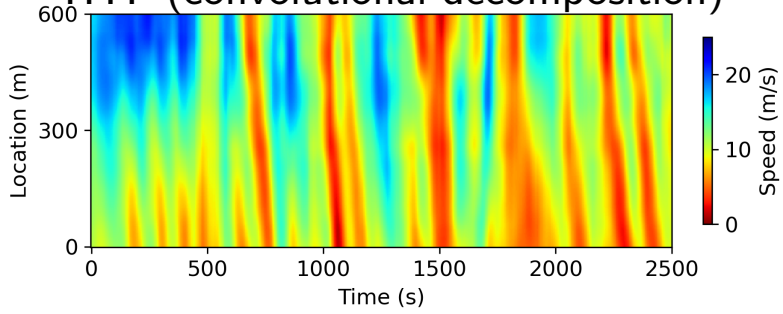


Sparse speed field



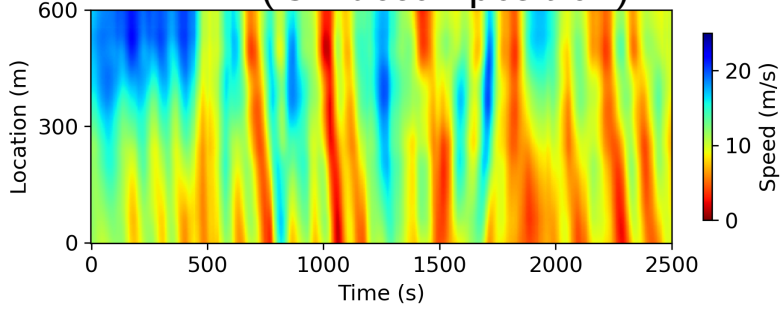
Ground truth speed field

HTF (convolutional decomposition)



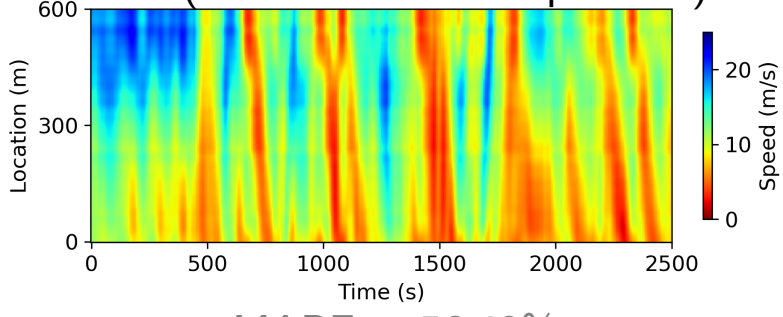
MAPE = **51.92%**

HTF (CP decomposition)



MAPE = 53.93%

HTF (tensor-train decomposition)



MAPE = 56.48%

Conclusion

Time-Varying Autoregression:

- (Highlight) Interpretable model with tensor factorization.
  - ✓ Parameter compression ✓ Pattern discovery

Laplacian Convolutional Representation:

- (Solution) Time series trend modeling in the low-rank framework?
  - Global time series trend modeling (low-rank model):

$$\begin{aligned} \min_{\boldsymbol{x}} \ & \|\mathcal{C}(\boldsymbol{x})\|_* \\ \text{s. t.} \ & \|\mathcal{P}_\Omega(\boldsymbol{x}-\boldsymbol{y})\|_2 \leq \epsilon \end{aligned}$$

  - Local time series trend modeling (temporal regularization):

$$\mathcal{R}_\tau(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{\ell} \star \boldsymbol{x}\|_2^2$$

- (Highlight) A unified framework with the FFT implementation.

Hankel Tensor Factorization:

- (Highlight) Memory-efficient Hankel indexing & convolutional parameterization.

POLYTECHNIQUE MONTRÉAL

UNIVERSITÉ D'INGÉNIERIE

IVADO

Thank you for your attention!

Any Questions?

About me:

- Homepage: https://xinychen.github.io
- GitHub: https://github.com/xinychen
- How to reach me: chenxy346@gmail.com

Appendix

HTF (convolutional decomposition)

- Optimization problem:

$$\min_{\boldsymbol{Q},\boldsymbol{S},\boldsymbol{U},\boldsymbol{V}} \ \frac{1}{2}\sum_{k_1,k_2}\left\|\mathcal{P}_{\Omega_{k_1,k_2}}\left(\theta_{k_1,k_2}(\boldsymbol{Y})-(\boldsymbol{Q}\star_{\text{row}}\boldsymbol{s}_{k_1})(\boldsymbol{U}\star_{\text{row}}\boldsymbol{v}_{k_2})^\top\right)\right\|_F^2 + \frac{\rho}{2}\left(\|\boldsymbol{Q}\|_F^2+\|\boldsymbol{S}\|_F^2+\|\boldsymbol{U}\|_F^2+\|\boldsymbol{V}\|_F^2\right)$$

- Alternating minimization:

$$\begin{cases} \boldsymbol{Q} \coloneqq \{\boldsymbol{Q} \mid \dfrac{\partial f}{\partial \boldsymbol{Q}} = \boldsymbol{0}\} \\ \boldsymbol{s}_{k_1} \coloneqq \{\boldsymbol{s}_{k_1} \mid \dfrac{\partial f}{\partial \boldsymbol{s}_{k_1}} = \boldsymbol{0}\},\ k_1 \in \{1,2,\ldots,\tau_1\} \\ \boldsymbol{U} \coloneqq \{\boldsymbol{U} \mid \dfrac{\partial f}{\partial \boldsymbol{U}} = \boldsymbol{0}\} \\ \boldsymbol{v}_{k_2} \coloneqq \{\boldsymbol{v}_{k_2} \mid \dfrac{\partial f}{\partial \boldsymbol{v}_{k_2}} = \boldsymbol{0}\},\ k_2 \in \{1,2,\ldots,\tau_2\} \end{cases}$$