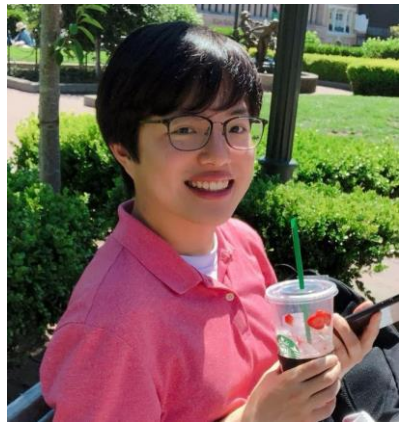
Robust Factorization of Real-world Tensor Streams with Patterns, Missing Values, and Outliers

Dongjin Lee



Kijung Shin

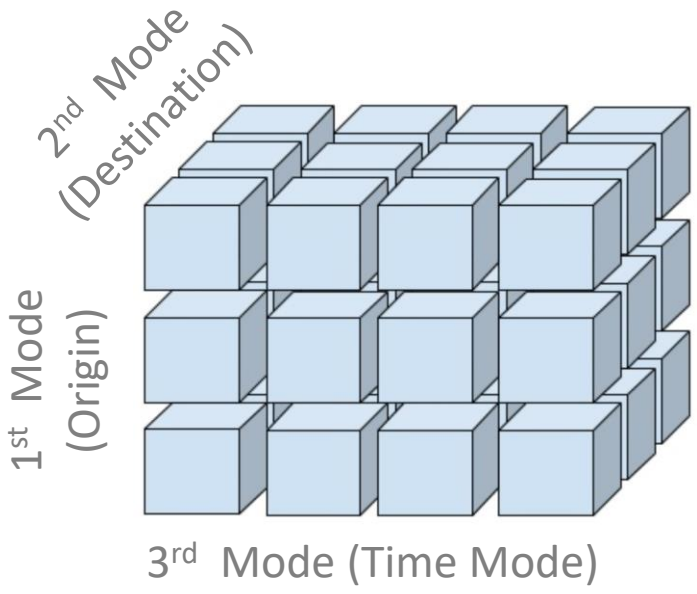
Road Map

- **Introduction** ⬅
- Problem Definition
- Proposed Method: **SOFIA**
  - Tensor Factorization Model
  - Optimization Algorithm
- Experiment Results
- Conclusions

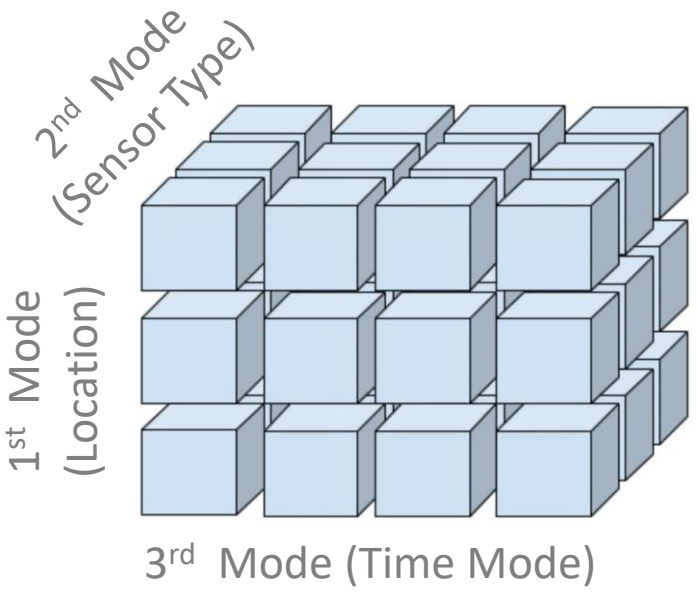
Tensors are Everywhere

Tensors

- Multi-dimensional arrays
- Tensors are effective tools to represent multi-aspect data.



Ex) Taxi origin-destination data

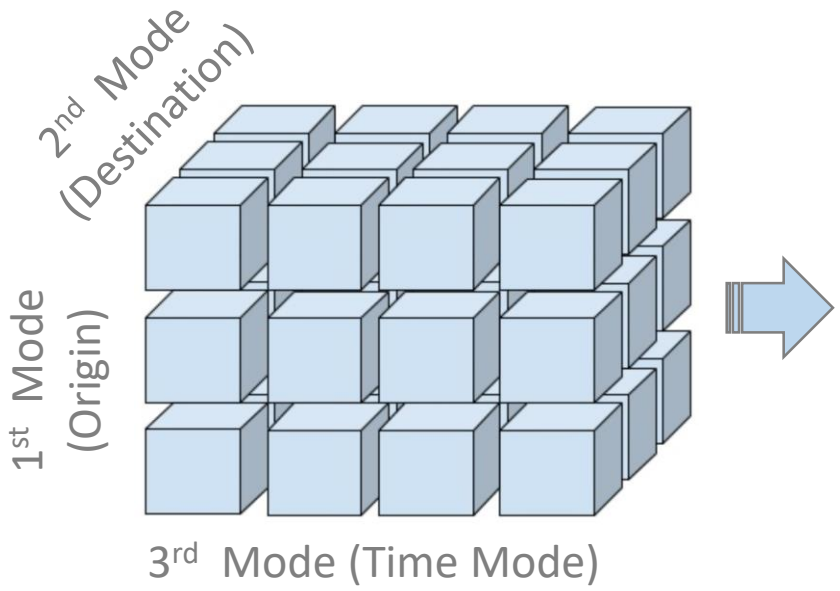


Ex) Indoor sensor data

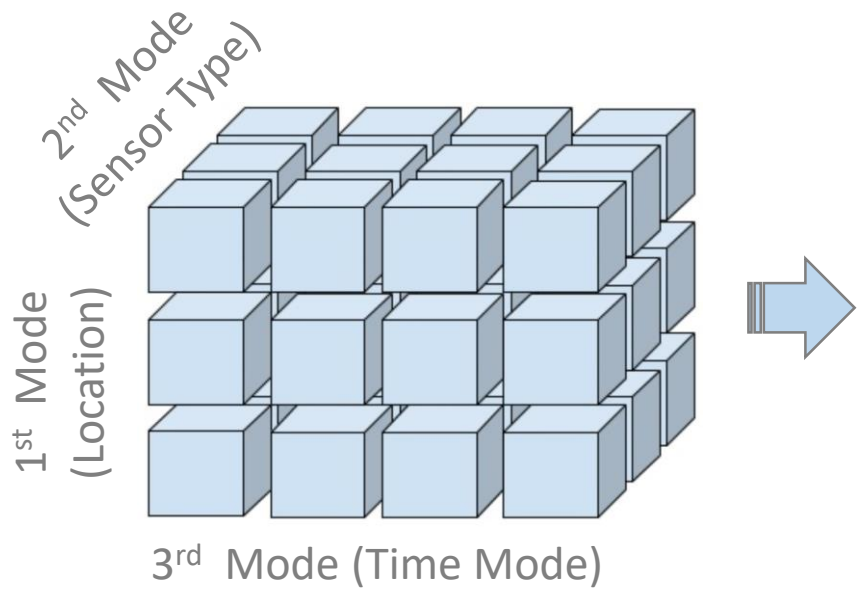
Tensor Streams

Tensor Streams

- In many applications, tensor data are usually collected incrementally over time in the form of a tensor stream.



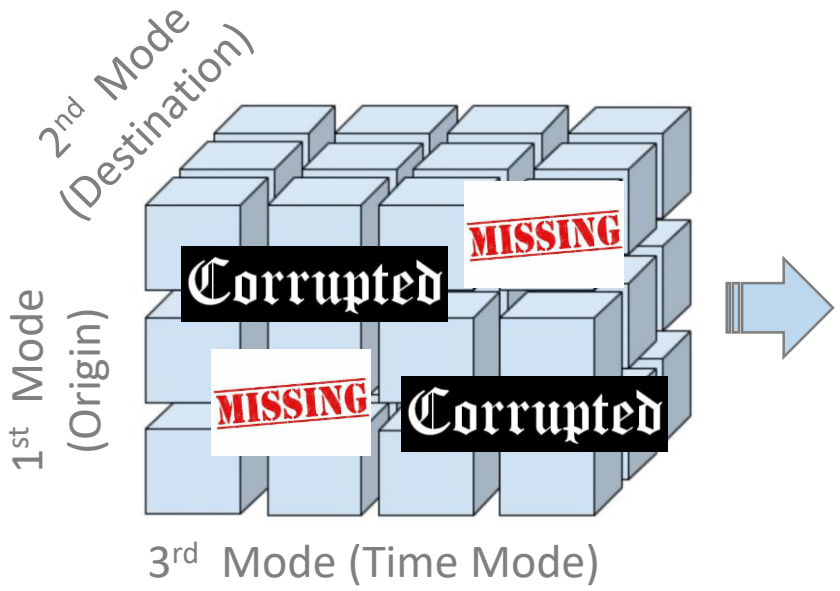
Ex) Taxi origin-destination data



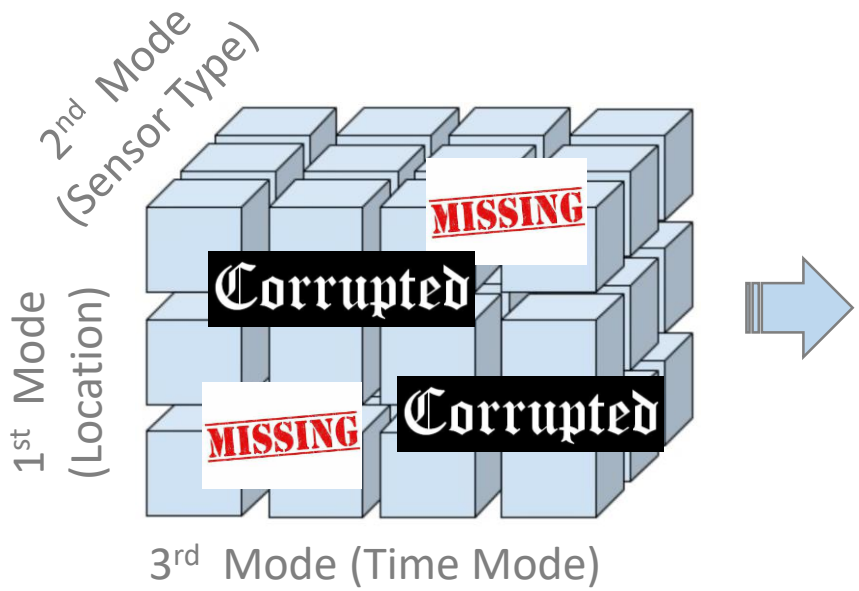
Ex) Indoor sensor data

Missing Values and Outliers in Real-world Tensor Data

- Real-world tensor streams often include missing entries (e.g., due to network disconnection) and unexpected outliers (e.g., due to system errors).



Ex) Taxi ride origin-destination tensor



Ex) Indoor sensor record tensor

Questions

Given that real-world tensor streams with missing values and outliers

How can we **estimate the missing entries**?

Can we also **predict future entries**?

Can both imputation and prediction **be performed accurately in an online manner**?

Traditional Methods and Their Limitations

- **CANDECOMP/PARAFAC (CP) Factorization** of Incomplete Tensors

Consider: $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, $\mathbf{\Omega} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, rank R

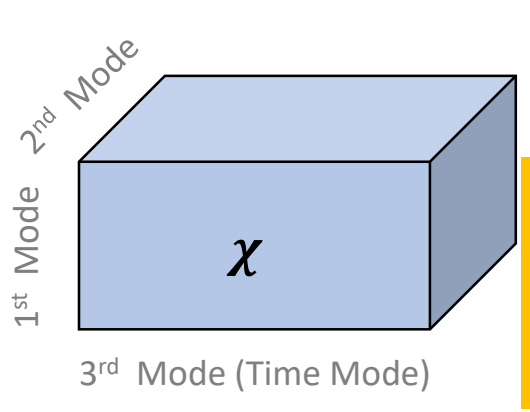
※ $\omega_{i_1 \cdots i_N} = \begin{cases} 1 & \text{if } x_{i_1 \cdots i_N} \text{ is known,} \\ 0 & \text{if } x_{i_1 \cdots i_N} \text{ is missing.} \end{cases}$

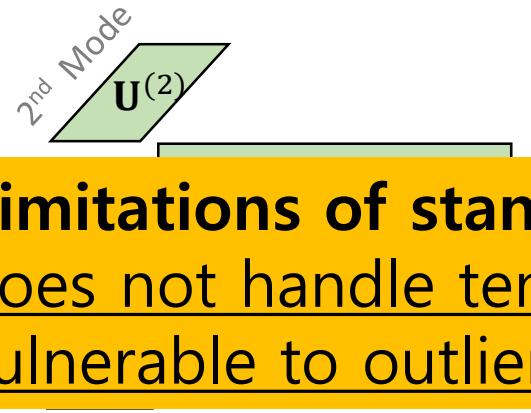
To Find: factor matrices $\mathbf{U}^{(1)}, \cdots, \mathbf{U}^{(N)}$

Minimize: $\frac{1}{2} \left\| \mathbf{\Omega} \circledast (\mathcal{X} - [\![\mathbf{U}^{(1)}, \cdots, \mathbf{U}^{(N)}]\!]) \right\|_F^2,$

where $[\![\mathbf{U}^{(1)}, \cdots, \mathbf{U}^{(N)}]\!] = \sum_{r=1}^{R} \mathbf{U}^{(1)}(:, r) \circ \cdots \circ \mathbf{U}^{(N)}(:, r).$

Ex) 3-order tensor case





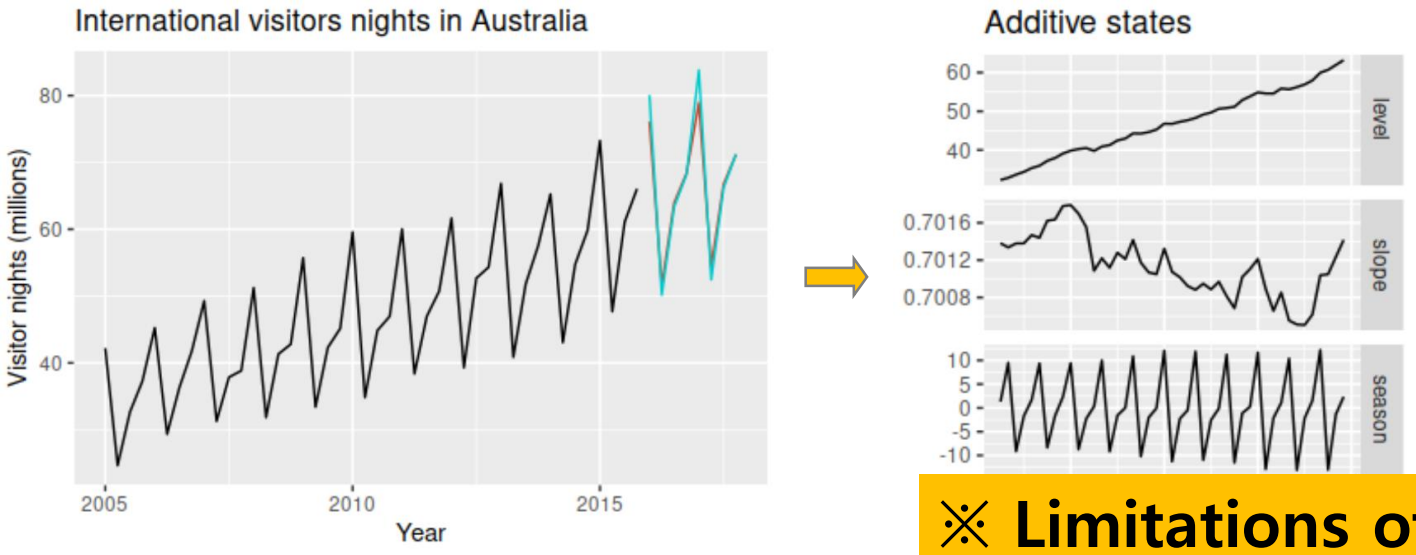
※ **Limitations of standard CP factorization**

- Does not handle tensor streams
- Vulnerable to outliers

Traditional Methods and Their Limitations (cont.)

Holt-Winters Forecasting Algorithm (Exponential Smoothing)

- Decomposes time series into **level**, **trend (slope)**, **seasonality** components
- Using three components, **it can forecast future evolution** of the time series.



From Hyndman, Rob J., and George Athanasopoulos. *Forecasting:*

Three smoothing equations:

$$l_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(l_{t-1} + b_{t-1}),$$
$$b_t = \beta(l_t - l_{t-1}) + (1-\beta)b_{t-1},$$
$$s_t = \gamma(y_t - l_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m}.$$

Forecasting equation:

$$\hat{y}_{t+h|t} = l_t + hb_t + s_{t+h-m(\lfloor \frac{h-1}{m} \rfloor + 1)},$$

※ Limitations of standard Holt-Winters Method

- Does not handle missing values
- Vulnerable to outliers

In this work

We propose **SOFIA**, a CP factorization-based streaming algorithm for real-world tensors with missing entries and outliers.

Questions


How can we **estimate the missing entries**?


Can we also **predict future entries**?

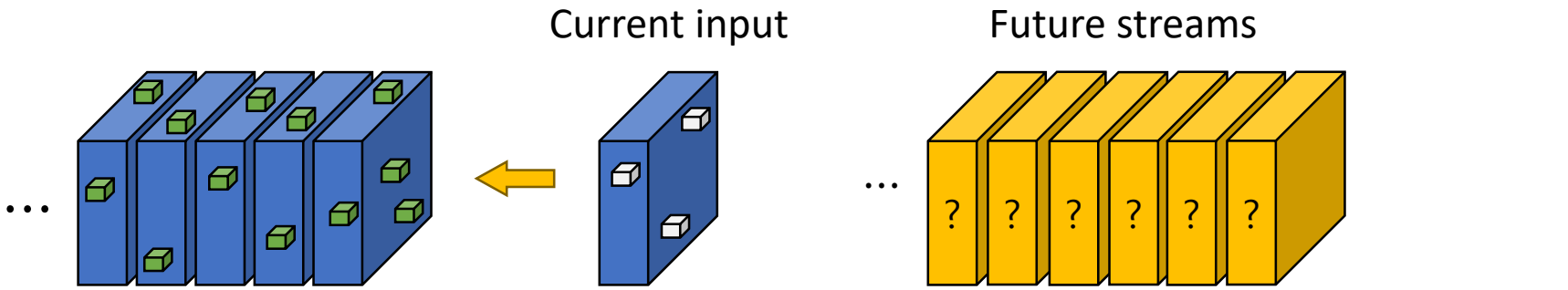

Can both imputation and prediction **be performed accurately in an online manner**?

Road Map

- Introduction
- **Problem Definition** ⬅
- Proposed Method: **SOFIA**
  - Tensor Factorization Model
  - Optimization Algorithm
- Experiment Results
- Conclusions

Problem Definition: Imputation & Forecasting

- **Given**:
  - A sequence of incomplete and noisy subtensors
- **Estimate**:
  - Missing entries in the subtensors (**imputation**)
  - Future values of the tensor stream (**forecasting**)
- **To minimize**:
  - Imputation and forecasting error



Estimate the value of missing entries

Forecasting future tensor streams

Road Map

- Introduction
- Problem Definition
- **Proposed Method: SOFIA**
  - Tensor Factorization Model
  - Optimization Algorithm
- Experiment Results
- Conclusions

Proposed Tensor Factorization Model

For static tensors

Temporal properties in real-world tensor streams.
1. **Temporal smoothness**
2. **Seasonal smoothness**

- **Input**
  - Corrupted tensor: $\mathcal{Y} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ $(\mathcal{Y} = \mathcal{X} + \mathcal{O})$

- **Optimization Problem**

Encourage temporal smoothness | Encourage seasonal smoothness | Encourage sparsity of outliers

$$min_{\{\mathbf{U}^{(n)}\}_{n=1}^{N},\mathcal{O}} \|\Omega \circledast (\mathcal{Y} - \mathcal{O} - \mathcal{X})\|_F^2 + \lambda_1 \left\|\mathbf{L}_1 \mathbf{U}^{(N)}\right\|_F^2 + \lambda_2 \left\|\mathbf{L}_m \mathbf{U}^{(N)}\right\|_F^2 + \lambda_3 \|\mathcal{O}\|_1$$

- **Output**
  - Non-temporal factor matrices: $\{\mathbf{U}^{(n)}\}_{n=1}^{N-1}$
  - Temporal factor matrix: $\mathbf{U}^{(N)}$ (Temporally and seasonally smooth)
  - Outlier tensor: $\mathcal{O}$

$$\boldsymbol{L}_1 = \begin{bmatrix} 1 & -1 & & \cdots & & \\ & 1 & -1 & & & \\ & \vdots & & \ddots & & \vdots \\ & & & & 1 & -1 \\ & & & \cdots & & 1 & -1 \end{bmatrix}$$

$$\boldsymbol{L}_m = \begin{bmatrix} 1 & \cdots & -1 & & \cdots & & \\ & 1 & \cdots & -1 & & & \\ & & \vdots & & \ddots & & \vdots \\ & & & & & 1 & \cdots & -1 \\ & & & & \cdots & & 1 & \cdots & -1 \end{bmatrix} \quad (m)$$

※ m: seasonal period

Proposed Tensor Factorization Model (cont.)

For dynamic tensors

- **Input**
  - Sequence of corrupted subtensor: $\boldsymbol{\mathcal{Y}}_t \in \mathbb{R}^{I_1 \times \cdots \times I_{N-1}}$ $(\boldsymbol{\mathcal{Y}}_t = \boldsymbol{\mathcal{X}}_t + \boldsymbol{\mathcal{O}}_t)$, where $t = 1, 2, \cdots$
- **Optimization Problem**
  - At each time t,

Encourage temporal smoothness | Encourage seasonal smoothness | Encourage sparsity of outliers

$$min_{\{\mathbf{U}^{(n)}\}_{n=1}^{N-1}, \{\mathbf{u}_\tau^{(N)}, \boldsymbol{\mathcal{O}}_\tau\}_{\tau=1}^{t}} \sum_{\tau=1}^{t} \left[\|\boldsymbol{\Omega}_\tau \circledast (\boldsymbol{\mathcal{Y}}_\tau - \boldsymbol{\mathcal{O}}_\tau - \boldsymbol{\mathcal{X}}_\tau)\|_F^2 + \lambda_1 \left\| \mathbf{u}_{\tau-1}^{(N)} - \mathbf{u}_\tau^{(N)} \right\|_F^2 + \lambda_2 \left\| \mathbf{u}_{\tau-m}^{(N)} - \mathbf{u}_\tau^{(N)} \right\|_F^2 + \lambda_3 \|\boldsymbol{\mathcal{O}}_\tau\|_1 \right]$$

Q. How to solve the optimization problem incrementally in an online manner?

- **Output**
  - Non-temporal factor matrices: $\{\mathbf{U}^{(n)}\}_{n=1}^{N-1}$
  - Temporal factor vectors: $\mathbf{u}_t^{(N)}$, where $t = 1, 2, \cdots$ (Temporally and seasonally smooth)
  - Outlier tensor: $\boldsymbol{\mathcal{O}}_t$, where $t = 1, 2, \cdots$

Proposed Optimization Algorithm

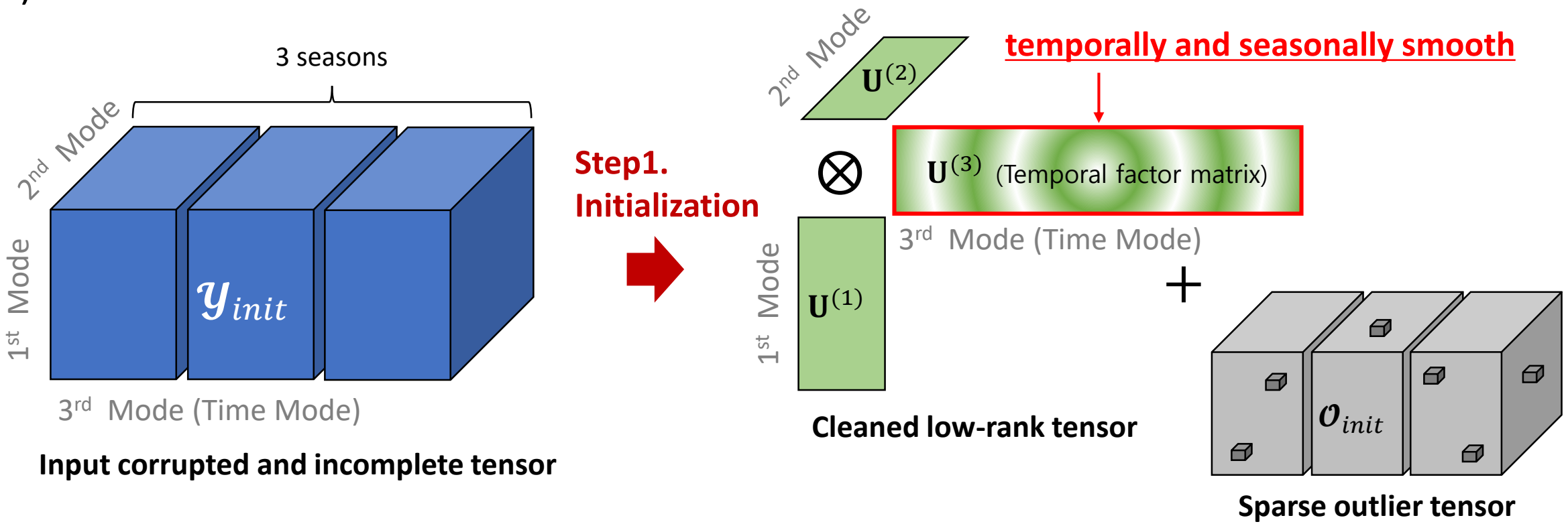
- SOFIA's optimization algorithm consists of three steps.

    1. **Initialization**
    2. **Fitting the Holt-Winters model**
    3. **Dynamic Update**

SOFIA – Step1. Initialization

- **Initialize all the factor matrices** $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ by solving the batch optimization problem using a subset of the corrupted tensor data over a short period of time (e.g., 3 seasons).

Ex) 3-order tensor case



SOFIA – Step1. Initialization (cont.)

Procedure:
SOFIA Initialization

Input: $\mathcal{Y}_{init}$

Initialization: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$

While $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ does not converge:

- Reject Outliers, $\mathcal{Y}_{init} - \mathcal{O}_{init}$
- Update $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ by **SOFIA**$_{\text{ALS}}$
- Update $\mathcal{O}_{init}$

Output: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$

※ **SOFIA**$_{\text{ALS}}$: the alternating least squares (ALS) method to minimize the objective function in the batch scenario.

Ex) 3-order tensor case

SOFIA – Step1. Initialization (cont.)

Procedure:
SOFIA Initialization

Input: $\mathcal{Y}_{init}$

Initialization: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$

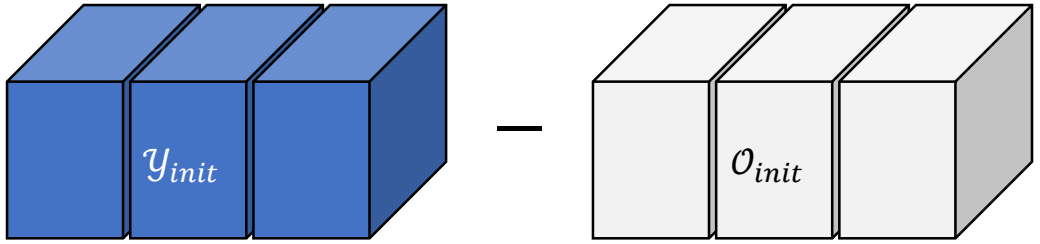
While $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ does not converge:

- Reject Outliers, $\mathcal{Y}_{init} - \mathcal{O}_{init}$ ⬅
- Update $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ by **SOFIA**$_{\text{ALS}}$
- Update $\mathcal{O}_{init}$

Output: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$

※ **SOFIA**$_{\text{ALS}}$: the alternating least squares (ALS) method to minimize the objective function in the batch scenario.

Ex) 3-order tensor case



SOFIA – Step1. Initialization (cont.)

Procedure:
SOFIA Initialization

Input: $\mathcal{Y}_{init}$

Initialization: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$

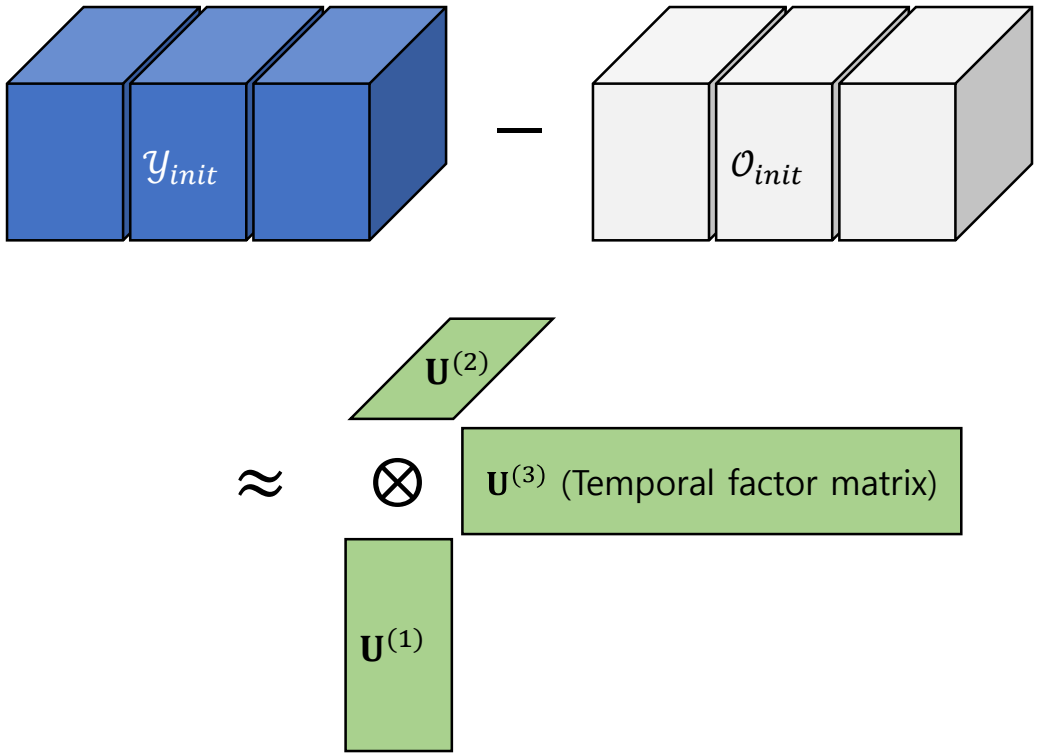
While $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ does not converge:

- Reject Outliers, $\mathcal{Y}_{init} - \mathcal{O}_{init}$
- Update $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ by **SOFIA**$_{\text{ALS}}$
- Update $\mathcal{O}_{init}$

Output: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$

※ **SOFIA**$_{\text{ALS}}$: the alternating least squares (ALS) method to minimize the objective function in the batch scenario.

Ex) 3-order tensor case



$$min_{\{\mathbf{U}^{(n)}\}_{n=1}^{N},\mathcal{O}} \|\Omega \circledast (\mathcal{Y} - \mathcal{O} - \mathcal{X})\|_F^2 + \lambda_1 \left\|\mathbf{L}_1 \mathbf{U}^{(N)}\right\|_F^2 + \lambda_2 \left\|\mathbf{L}_m \mathbf{U}^{(N)}\right\|_F^2 + \lambda_3 \|\mathcal{O}\|_1$$

SOFIA – Step1. Initialization (cont.)

Procedure:
SOFIA Initialization

Input: $\mathcal{Y}_{init}$

Initialization: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$

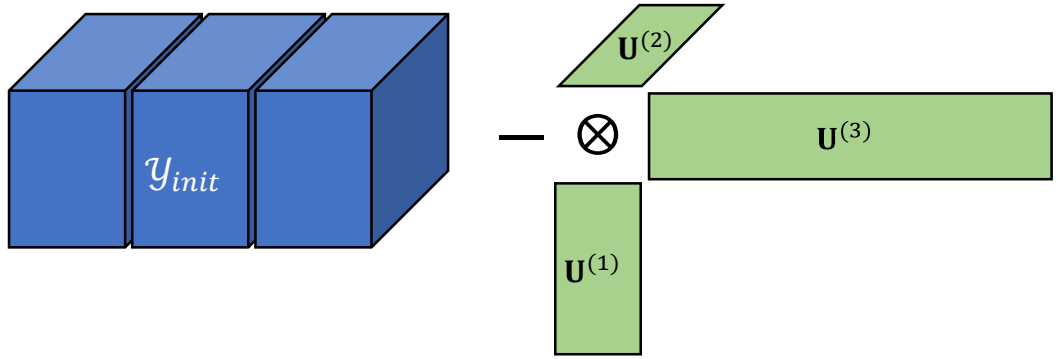
While $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ does not converge:

- Reject Outliers, $\mathcal{Y}_{init} - \mathcal{O}_{init}$
- Update $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ by **SOFIA**$_{\text{ALS}}$
- Update $\mathcal{O}_{init}$ ⬅

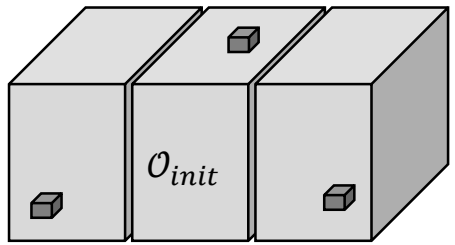
Output: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$

※ **SOFIA**$_{\text{ALS}}$: the alternating least squares (ALS) method to minimize the objective function in the batch scenario.

Ex) 3-order tensor case



$$= \mathcal{Y}_{init} - \left[\!\left[\{\mathbf{U}^{(n)}\}_{n=1}^{N}\right]\!\right] = \mathcal{R}$$



$$= \text{sign}(\mathcal{R}) \cdot \max(|\mathcal{R}| - \lambda_3, 0)$$

$$min_{\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}} \|\Omega \circledast (\mathcal{Y} - \mathcal{O} - \mathcal{X})\|_F^2 + \lambda_1 \left\|\mathbf{L}_1 \mathbf{U}^{(N)}\right\|_F^2 + \lambda_2 \left\|\mathbf{L}_m \mathbf{U}^{(N)}\right\|_F^2 + \lambda_3 \|\mathcal{O}\|_1$$

SOFIA – Step1. Initialization (cont.)

Procedure:
SOFIA Initialization

Input: $\mathcal{Y}_{init}$

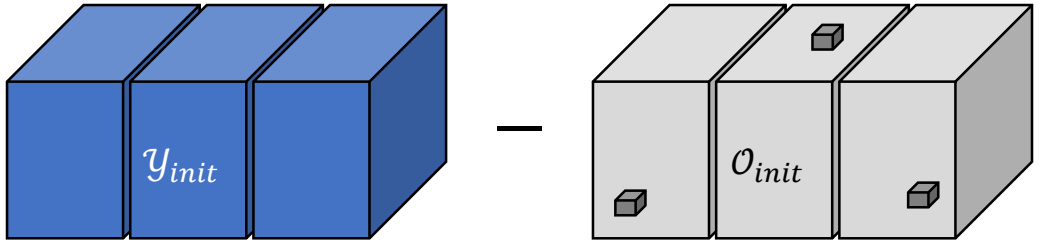
Initialization: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$

While $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ does not converge:

- Reject Outliers, $\mathcal{Y}_{init} - \mathcal{O}_{init}$ ⬅
- Update $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ by **SOFIA**$_{\text{ALS}}$
- Update $\mathcal{O}_{init}$

Output: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$

Ex) 3-order tensor case



※ **SOFIA**$_{\text{ALS}}$: the alternating least squares (ALS) method to minimize the objective function in the batch scenario.

$$min_{\{\mathbf{U}^{(n)}\}_{n=1}^{N},\mathcal{O}} \|\Omega \circledast (\mathcal{Y} - \mathcal{O} - \mathcal{X})\|_F^2 + \lambda_1 \left\|\mathbf{L}_1 \mathbf{U}^{(N)}\right\|_F^2 + \lambda_2 \left\|\mathbf{L}_m \mathbf{U}^{(N)}\right\|_F^2 + \lambda_3 \|\mathcal{O}\|_1$$

SOFIA – Step1. Initialization (cont.)

Procedure:
SOFIA Initialization

Input: $\mathcal{Y}_{init}$

Initialization: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$

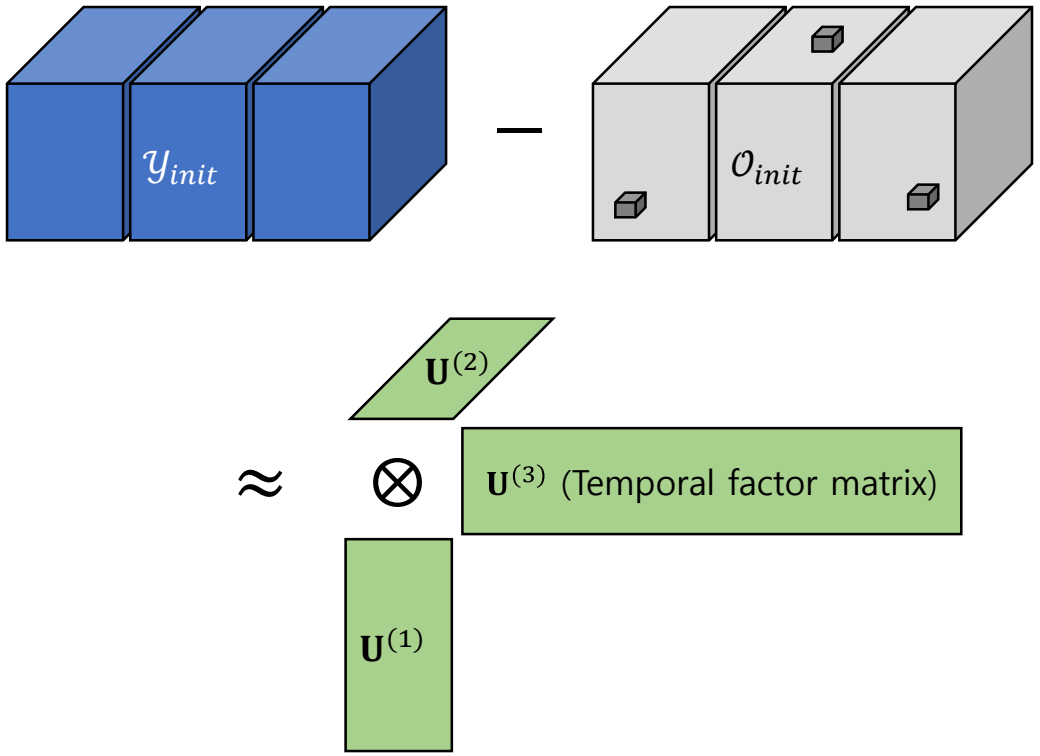
While $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ does not converge:

- Reject Outliers, $\mathcal{Y}_{init} - \mathcal{O}_{init}$
- Update $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ by **SOFIA**$_{\text{ALS}}$
- Update $\mathcal{O}_{init}$

Output: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$

※ **SOFIA**$_{\text{ALS}}$: the alternating least squares (ALS) method to minimize the objective function in the batch scenario.

Ex) 3-order tensor case



$$min_{\{\mathbf{U}^{(n)}\}_{n=1}^{N},\mathcal{O}} \|\Omega \circledast (\mathcal{Y} - \mathcal{O} - \mathcal{X})\|_F^2 + \lambda_1 \left\|\mathbf{L}_1 \mathbf{U}^{(N)}\right\|_F^2 + \lambda_2 \left\|\mathbf{L}_m \mathbf{U}^{(N)}\right\|_F^2 + \lambda_3 \|\mathcal{O}\|_1$$

SOFIA – Step1. Initialization (cont.)

Procedure:
SOFIA Initialization

Input: $\mathcal{Y}_{init}$

Initialization: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$

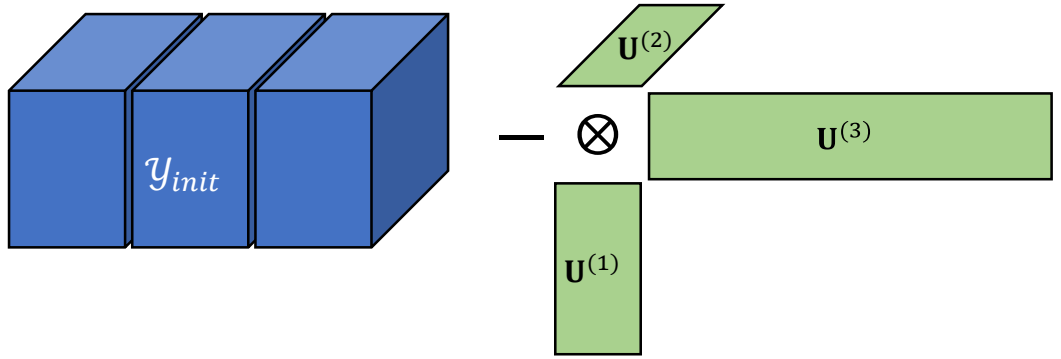
While $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ does not converge:

- Reject Outliers, $\mathcal{Y}_{init} - \mathcal{O}_{init}$
- Update $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ by **SOFIA**$_{\text{ALS}}$
- Update $\mathcal{O}_{init}$ ⬅

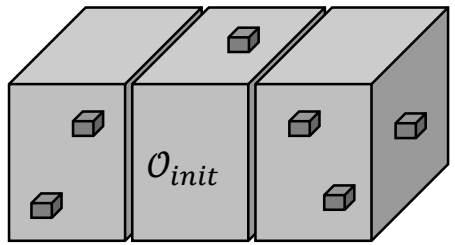
Output: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$

※ **SOFIA**$_{\text{ALS}}$: the alternating least squares (ALS) method to minimize the objective function in the batch scenario.

Ex) 3-order tensor case



$$= \mathcal{Y}_{init} - \left[\!\left[\{\mathbf{U}^{(n)}\}_{n=1}^{N}\right]\!\right] = \mathcal{R}$$



$$= \text{sign}(\mathcal{R}) \cdot \max(|\mathcal{R}| - \lambda_3, 0)$$

$$min_{\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}} \|\Omega \circledast (\mathcal{Y} - \mathcal{O} - \mathcal{X})\|_F^2 + \lambda_1 \left\|\mathbf{L}_1 \mathbf{U}^{(N)}\right\|_F^2 + \lambda_2 \left\|\mathbf{L}_m \mathbf{U}^{(N)}\right\|_F^2 + \lambda_3 \|\mathcal{O}\|_1$$

SOFIA – Step1. Initialization (cont.)

Procedure:
SOFIA Initialization

Input: $\mathcal{Y}_{init}$

Initialization: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$

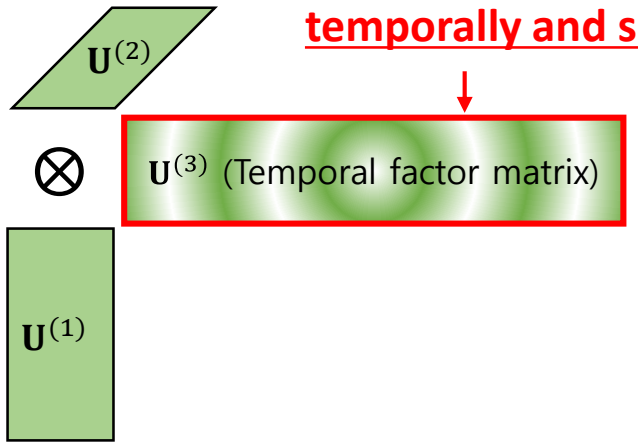
While $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ does not converge:

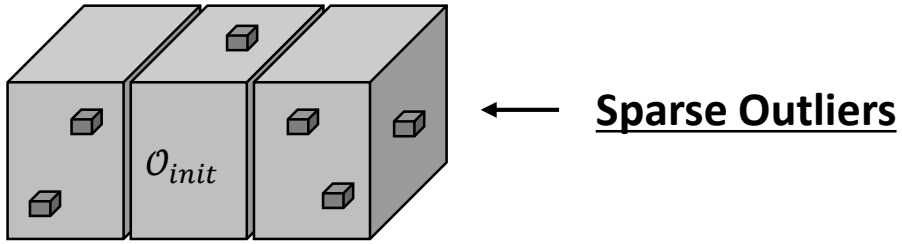
- Reject Outliers, $\mathcal{Y}_{init} - \mathcal{O}_{init}$
- Update $\{\mathbf{U}^{(n)}\}_{n=1}^{N}$ by **SOFIA**$_{\text{ALS}}$
- Update $\mathcal{O}_{init}$

Output: $\{\mathbf{U}^{(n)}\}_{n=1}^{N}, \mathcal{O}_{init}$ ⬅

※ **SOFIA**$_{\text{ALS}}$: the alternating least squares (ALS) method to minimize the objective function in the batch scenario.

Ex) 3-order tensor case





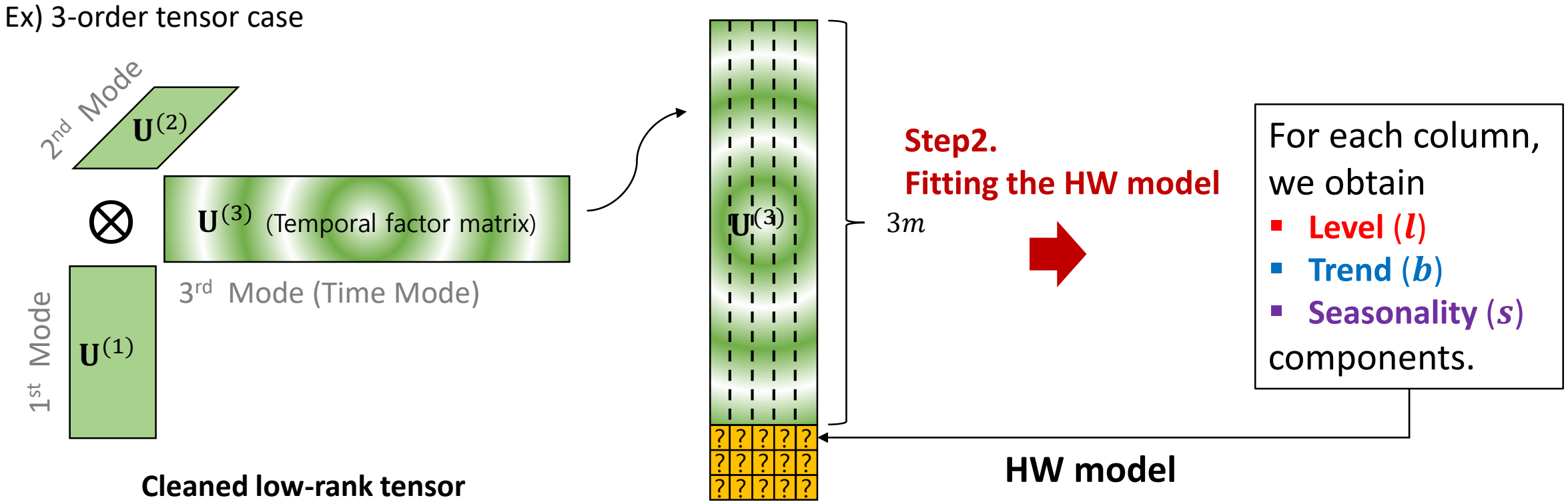
$$min_{\{\mathbf{U}^{(n)}\}_{n=1}^{N},\mathcal{O}} \|\boldsymbol{\Omega} \circledast (\mathcal{Y} - \mathcal{O} - \mathcal{X})\|_F^2 + \lambda_1 \left\|\mathbf{L}_1 \mathbf{U}^{(N)}\right\|_F^2 + \lambda_2 \left\|\mathbf{L}_m \mathbf{U}^{(N)}\right\|_F^2 + \lambda_3 \|\mathcal{O}\|_1$$

SOFIA – Step2. Fitting the Holt-Winters model

- Each column vectors of $\mathbf{U}^{(N)}$

  = Seasonal time series of length $3m$ with period m

- Using the Holt-Winters model, **we can forecast the next temporal factor vector**.



SOFIA – Step3. Dynamic Update

Procedure:
SOFIA Dynamic Update

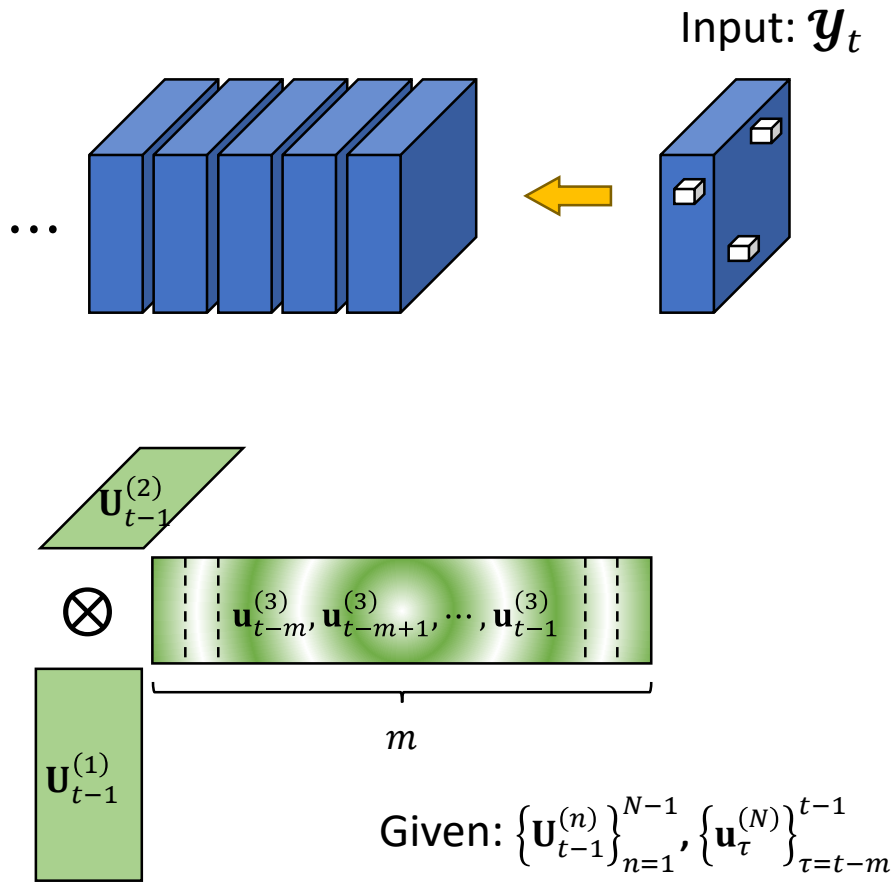
Input: $\mathcal{Y}_t$

Given: $\left\{\mathbf{U}_{t-1}^{(n)}\right\}_{n=1}^{N-1}, \left\{\mathbf{u}_{\tau}^{(N)}\right\}_{\tau=t-m}^{t-1}$

For $t = 3m+1, 3m+2, \cdots$ do

- Estimate $\mathcal{O}_t$
- Update $\left\{\mathbf{U}_t^{(n)}\right\}_{n=1}^{N-1}$
- Update $\mathbf{u}_t^{(N)}$
- Update Holt-Winters model
- Estimate missing values

Ex) 3-order tensor case



SOFIA – Step3. Dynamic Update (cont.)

Procedure:
SOFIA Dynamic Update

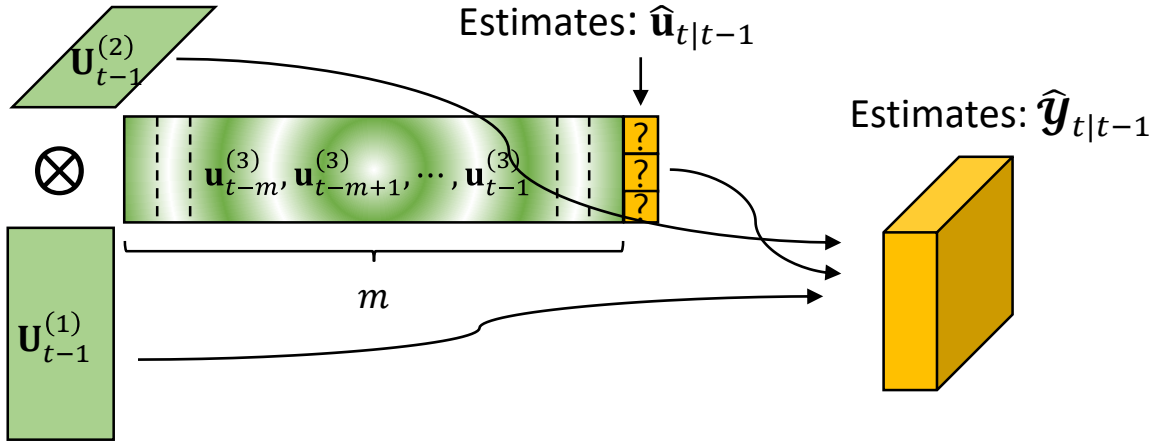
Input: $\mathcal{Y}_t$

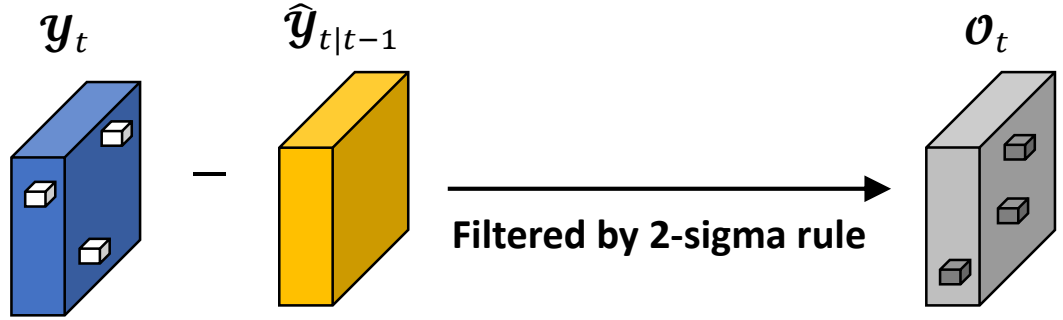
Given: $\left\{\mathbf{U}_{t-1}^{(n)}\right\}_{n=1}^{N-1}, \left\{\mathbf{u}_{\tau}^{(N)}\right\}_{\tau=t-m}^{t-1}$

For $t = 3m+1, 3m+2, \cdots$ do

- Estimate $\mathcal{O}_t$ ⬅
- Update $\left\{\mathbf{U}_t^{(n)}\right\}_{n=1}^{N-1}$
- Update $\mathbf{u}_t^{(N)}$
- Update Holt-Winters model
- Estimate missing values

Ex) 3-order tensor case





SOFIA – Step3. Dynamic Update (cont.)

Procedure:
SOFIA Dynamic Update

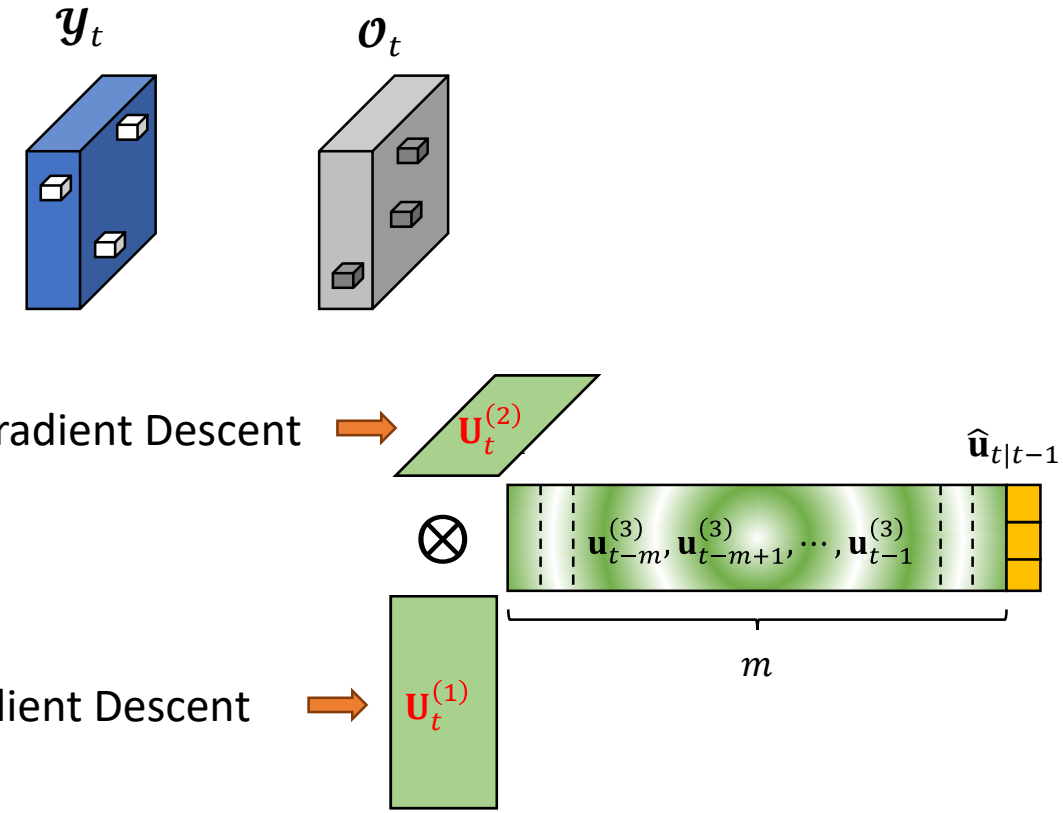
Input: $\mathcal{Y}_t$

Given: $\left\{\mathbf{U}_{t-1}^{(n)}\right\}_{n=1}^{N-1}, \left\{\mathbf{u}_{\tau}^{(N)}\right\}_{\tau=t-m}^{t-1}$

For $t = 3m+1, 3m+2, \cdots$ do

- Estimate $\mathcal{O}_t$
- Update $\left\{\mathbf{U}_t^{(n)}\right\}_{n=1}^{N-1}$ ⬅
- Update $\mathbf{u}_t^{(N)}$
- Update Holt-Winters model
- Estimate missing values

Ex) 3-order tensor case



$$min_{\{\mathbf{U}^{(} \quad \sum_{t}}$$

$$min_{\left\{\mathbf{U}_t^{(n)}\right\}_{n=1}^{N-1}, \mathbf{u}_t^{(N)}, \mathcal{O}_t} \|\Omega_t \circledast (\mathcal{Y}_t - \mathcal{O}_t - \mathcal{X}_t)\|_F^2 + \lambda_1 \left\|\mathbf{u}_{t-1}^{(N)} - \mathbf{u}_t^{(N)}\right\|_F^2 + \lambda_2 \left\|\mathbf{u}_{t-m}^{(N)} - \mathbf{u}_t^{(N)}\right\|_F^2 + \lambda_3 \|\mathcal{O}_t\|_1$$

SOFIA – Step3. Dynamic Update (cont.)

Procedure:
SOFIA Dynamic Update

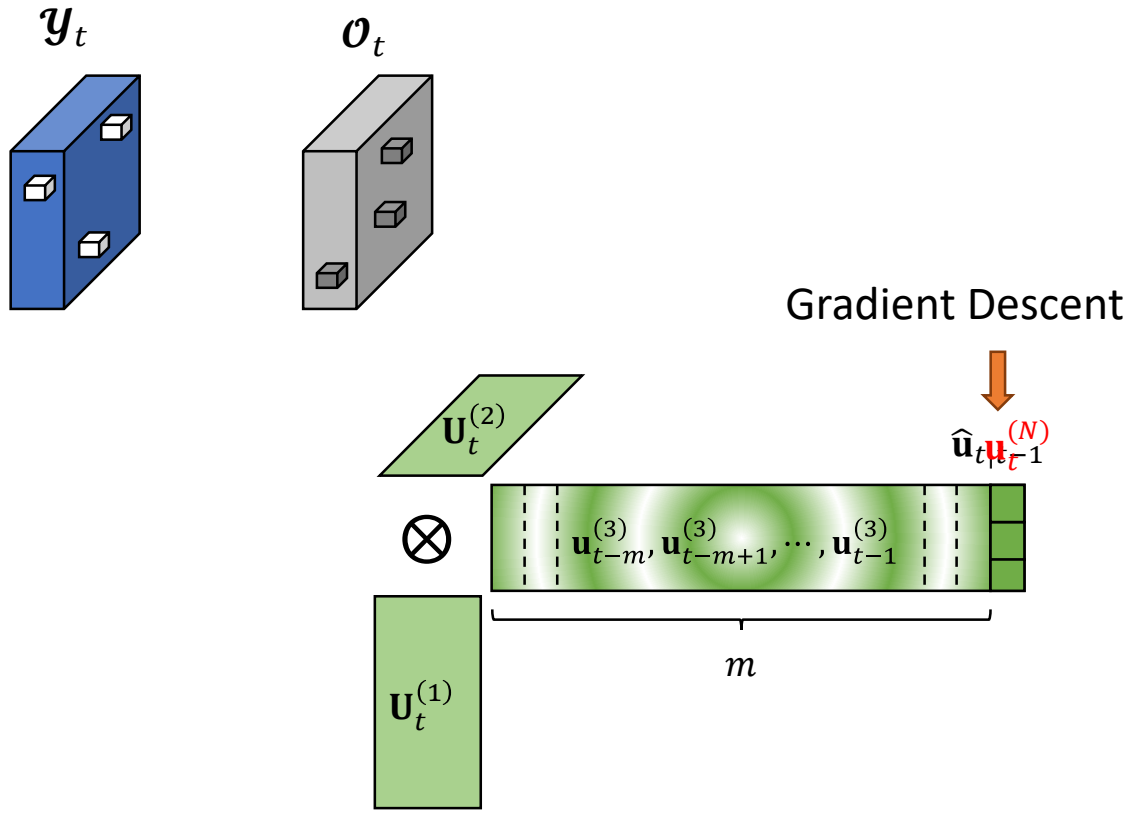
Input: $\mathcal{Y}_t$

Given: $\left\{\mathbf{U}_{t-1}^{(n)}\right\}_{n=1}^{N-1}, \left\{\mathbf{u}_{\tau}^{(N)}\right\}_{\tau=t-m}^{t-1}$

For $t = 3m+1, 3m+2, \cdots$ do

- Estimate $\mathcal{O}_t$
- Update $\left\{\mathbf{U}_t^{(n)}\right\}_{n=1}^{N-1}$
- Update $\mathbf{u}_t^{(N)}$ ⬅
- Update Holt-Winters model
- Estimate missing values

Ex) 3-order tensor case



$$min_{\left\{\mathbf{U}_t^{(n)}\right\}_{n=1}^{N-1}, \mathbf{u}_t^{(N)}, \mathcal{O}_t} \|\Omega_t \circledast (\mathcal{Y}_t - \mathcal{O}_t - \mathcal{X}_t)\|_F^2 + \lambda_1 \left\|\mathbf{u}_{t-1}^{(N)} - \mathbf{u}_t^{(N)}\right\|_F^2 + \lambda_2 \left\|\mathbf{u}_{t-m}^{(N)} - \mathbf{u}_t^{(N)}\right\|_F^2 + \lambda_3 \|\mathcal{O}_t\|_1$$

SOFIA – Step3. Dynamic Update (cont.)

Procedure:
SOFIA Dynamic Update

Input: $\mathcal{Y}_t$

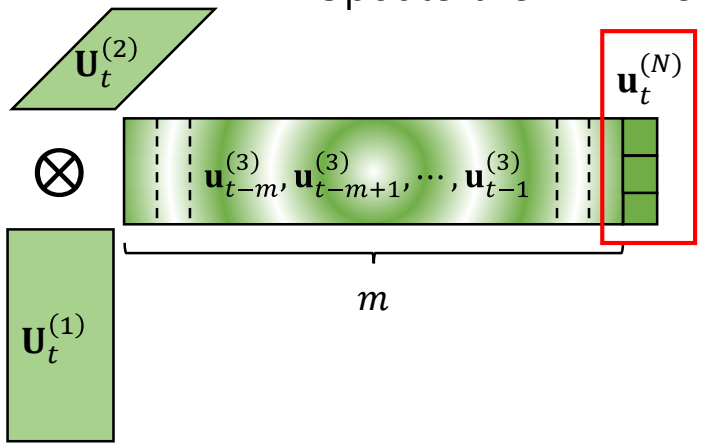
Given: $\left\{\mathbf{U}_{t-1}^{(n)}\right\}_{n=1}^{N-1}, \left\{\mathbf{u}_{\tau}^{(N)}\right\}_{\tau=t-m}^{t-1}$

For $t = 3m+1, 3m+2, \cdots$ do

- Estimate $\mathcal{O}_t$
- Update $\left\{\mathbf{U}_t^{(n)}\right\}_{n=1}^{N-1}$
- Update $\mathbf{u}_t^{(N)}$
- Update Holt-Winters model ⬅
- Estimate missing values

Ex) 3-order tensor case

Update the HW model with $\mathbf{u}_t^{(N)}$



We obtain slightly updated
- **Level (l)**
- **Trend (b)**
- **Seasonality (s)**

components.

SOFIA – Step3. Dynamic Update (cont.)

Procedure:
SOFIA Dynamic Update

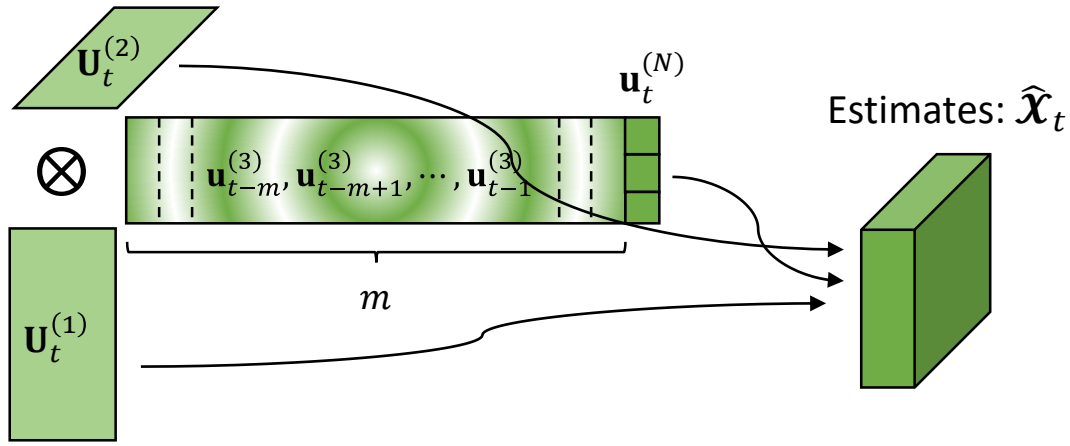
Input: $\mathcal{Y}_t$

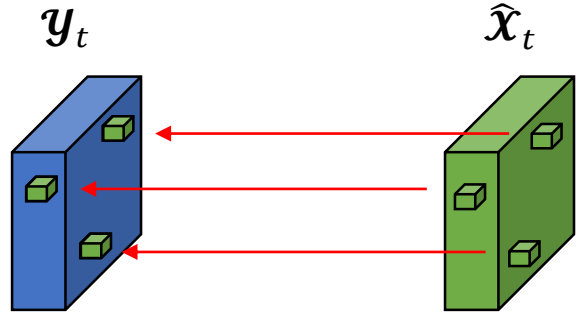
Given: $\left\{\mathbf{U}_{t-1}^{(n)}\right\}_{n=1}^{N-1}, \left\{\mathbf{u}_{\tau}^{(N)}\right\}_{\tau=t-m}^{t-1}$

For $t = 3m+1, 3m+2, \cdots$ do

- Estimate $\mathcal{O}_t$
- Update $\left\{\mathbf{U}_t^{(n)}\right\}_{n=1}^{N-1}$
- Update $\mathbf{u}_t^{(N)}$
- Update Holt-Winters model
- Estimate missing values ⬅

Ex) 3-order tensor case



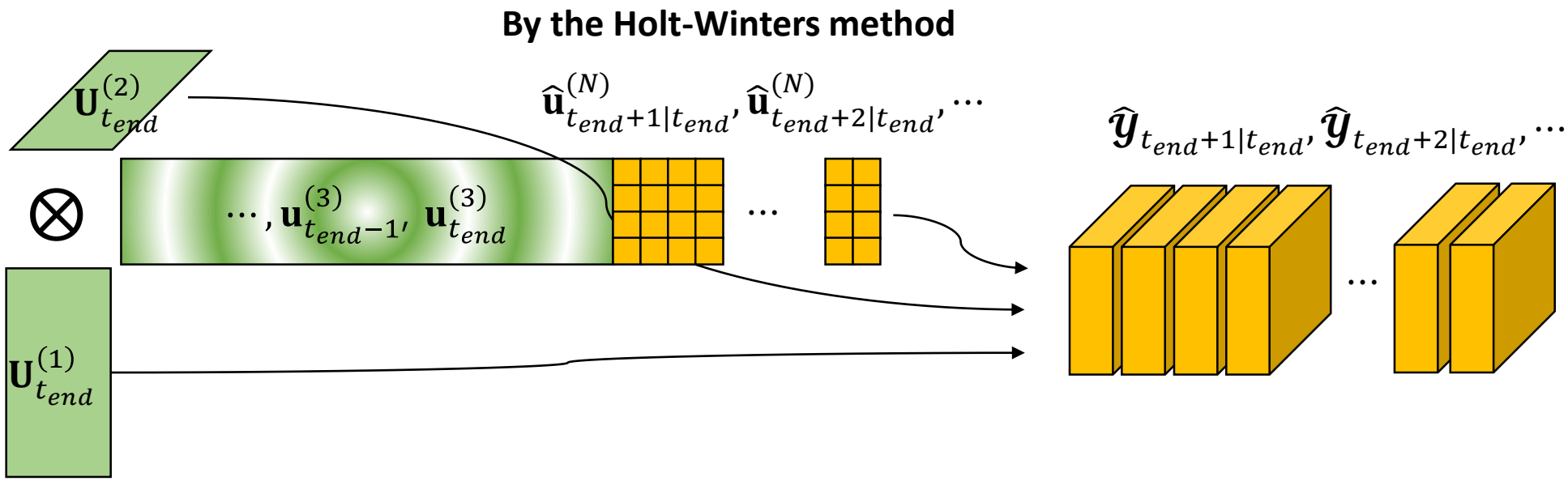


Forecasting Future Evolution of Tensor Streams

- Given any $t = t_{end} + h$, we can forecast future subtensors.

$$\hat{\mathcal{Y}}_{t|t_{end}} = \left[\!\left[\left\{ \mathbf{U}^{(n)}_{t_{end}} \right\}_{n=1}^{N-1}, \hat{\mathbf{u}}_{t|t_{end}} \right]\!\right]$$

Ex) 3-order tensor case



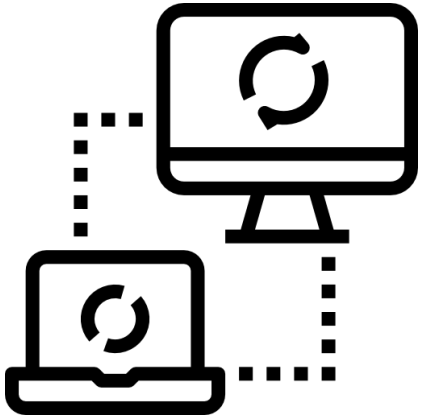
Road Map

- Introduction
- Problem Definition
- Proposed Method: **SOFIA**
  - Tensor Factorization Model
  - Optimization Algorithm
- **Experimental Results** ⬅
- Conclusions

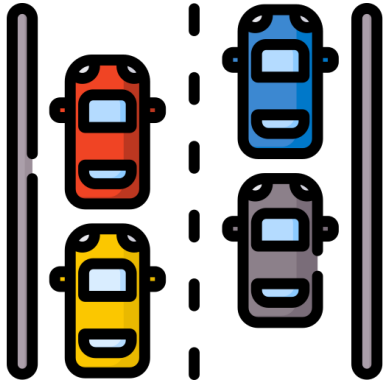
Experimental Settings

- 4 real-world time series datasets modeled by a tensor stream

Intel Lab Sensor



Network Traffic



Chicago Taxi Traffic

NYC Taxi Traffic

- A tuple (X, Y, Z) denotes the experimental setting.

(X, Y, Z)

missing ratio (%) — X

outlier ratio (%) — Y

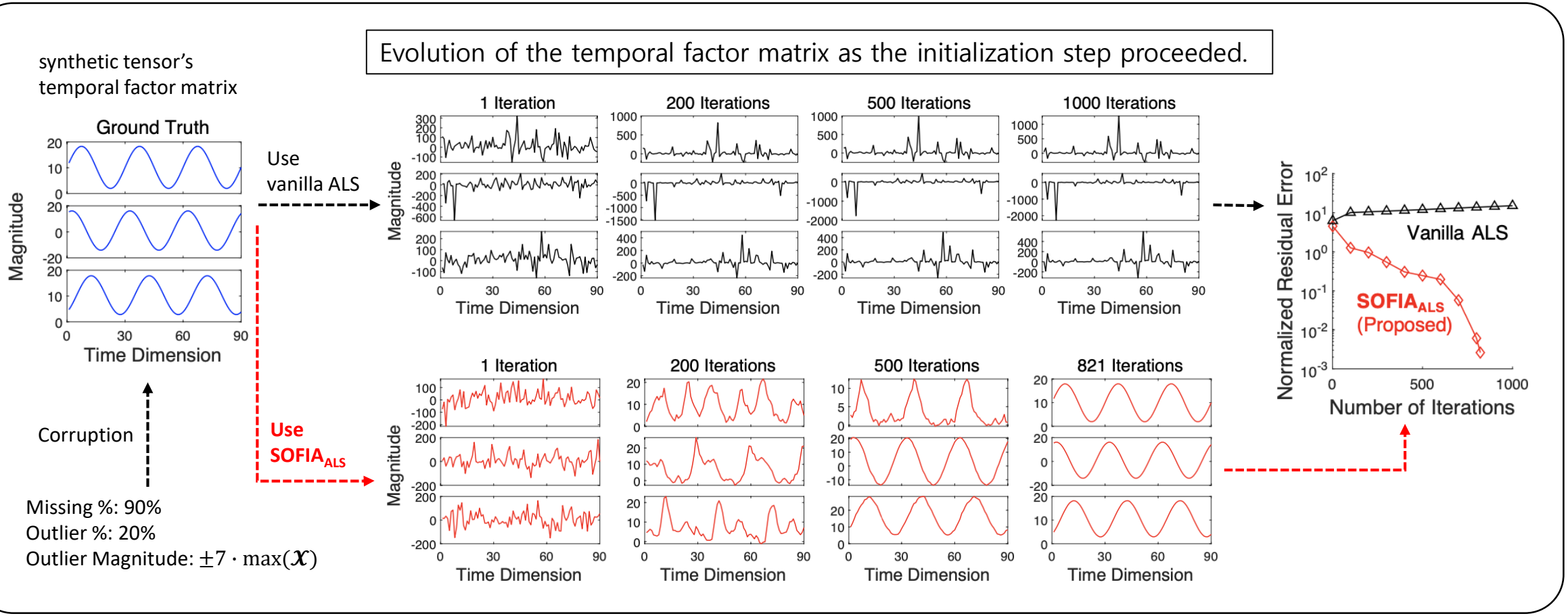
outlier magnitude — Z

ex) $(20, 10, 2)$

- 20% of entries are missing
- 10% of entries are outliers
- outlier magnitude is $\pm 2 \cdot \max(\mathcal{X})$ with equal probability

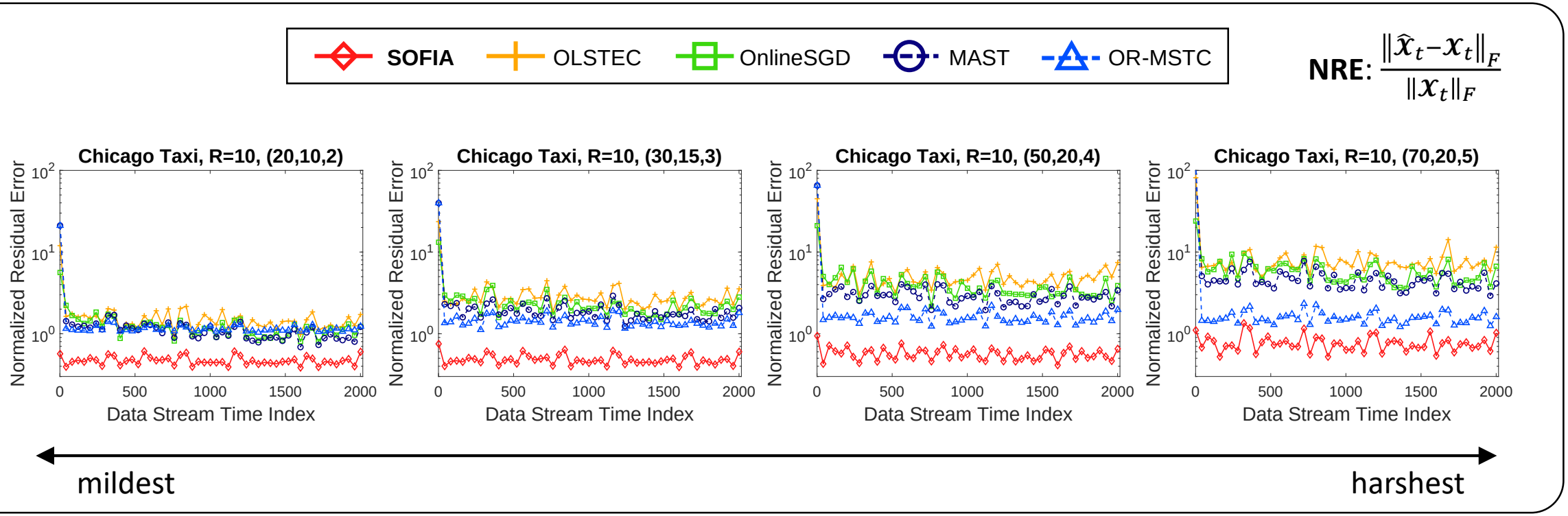
Exp1. Initialization Accuracy

- **SOFIA$_{ALS}$ accurately captured temporal patterns** from an incomplete and noisy tensor in the initialization step.



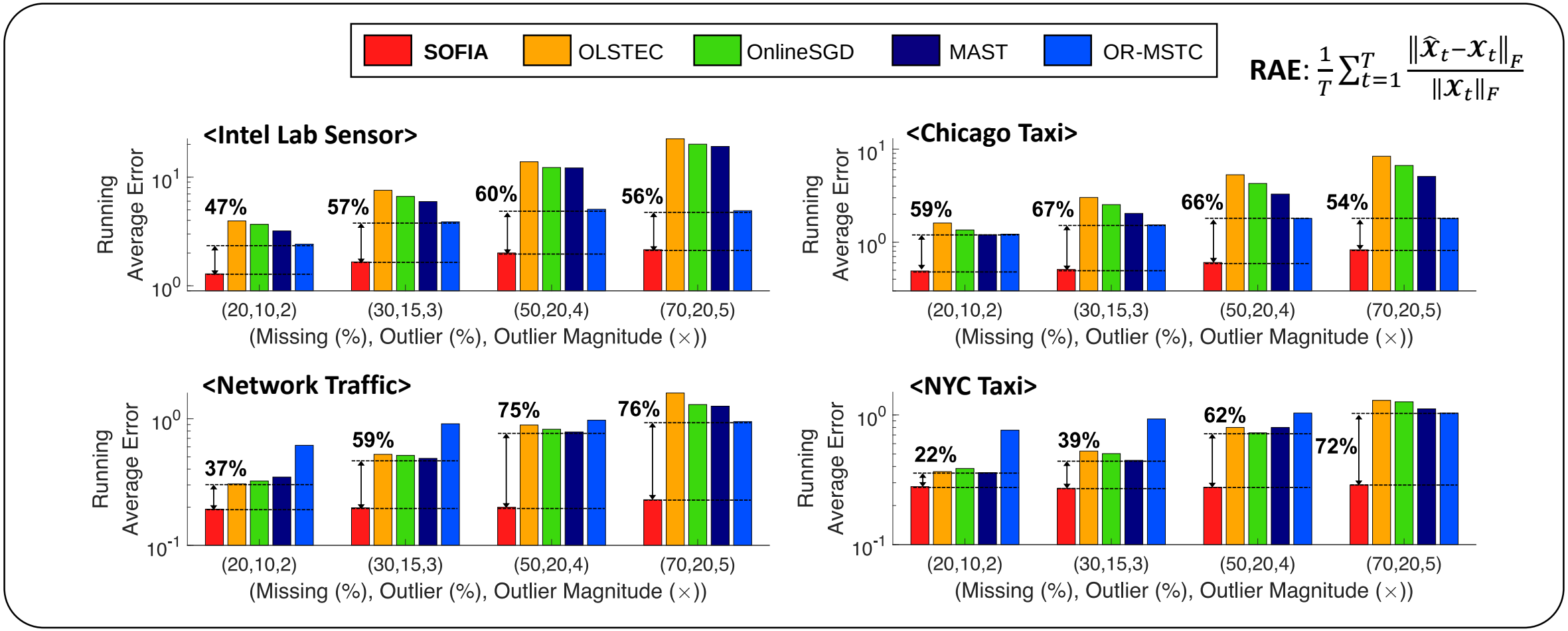
Exp2. Imputation Accuracy

- The normalized residual error (NRE) under 4 experimental settings from the mildest (leftmost) to the harshest (rightmost).
- **SOFIA was the most accurate** in all the tensor streams.



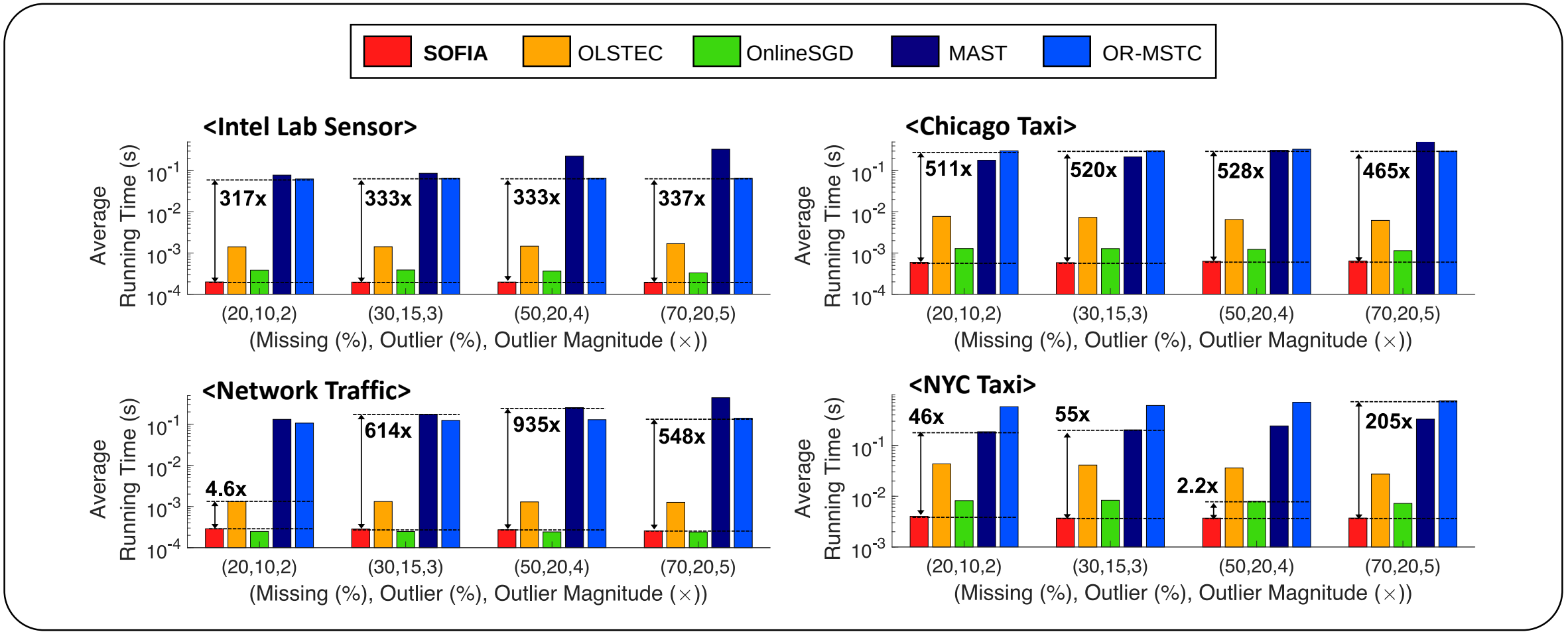
Exp2. Imputation Accuracy (cont.)

- **<u>SOFIA gave up to 76% smaller running average error (RAE)</u>** than the second-most accurate approach.



Exp3. Speed

- **SOFIA was up to 935× faster** than the second-most accurate algorithm.



Exp4. Forecasting Accuracy

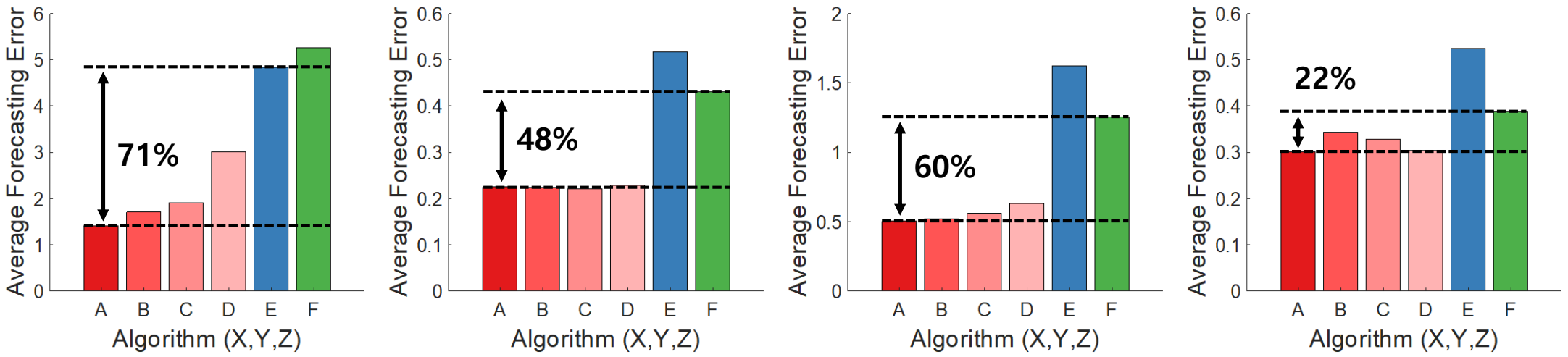
- **<u>Forecasting with SOFIA was the most accurate</u>**, despite the presence of missing values.

AFE: $\frac{1}{t_f}\sum_{h=1}^{t_f}\frac{\|\widehat{\mathcal{X}}_{t+h|t}-\mathcal{X}_{t+h}\|_F}{\|\mathcal{X}_{t+h}\|_F}$

SOFIA can operate on various levels of missing ratios.

Assume that all entries are observed.

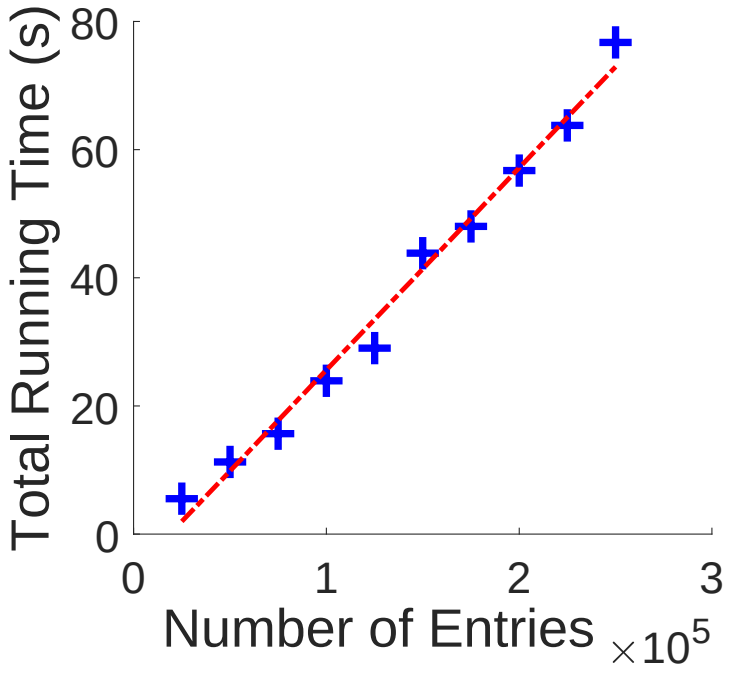
A **SOFIA** (0,20,5) B **SOFIA** (30,20,5) C **SOFIA** (50,20,5) D **SOFIA** (70,20,5) E SMF (0,20,5) F CPHW (0,20,5)

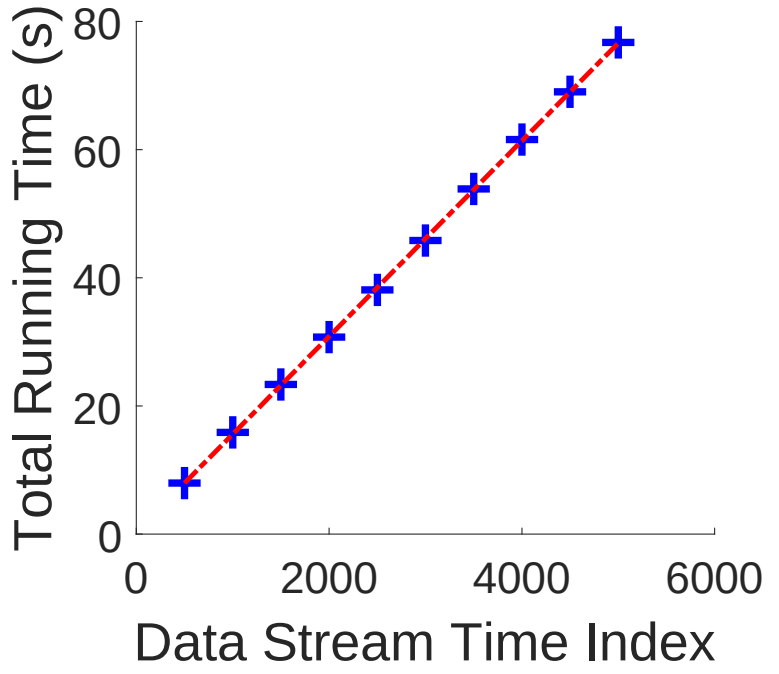


<Intel Lab Sensor> **<Network Traffic>** **<Chicago Taxi>** **<NYC Taxi>**

Exp5. Scalability

- Elapsed time taken by SOFIA per time step was **almost constant** regardless of the number of subtensors processed so far.





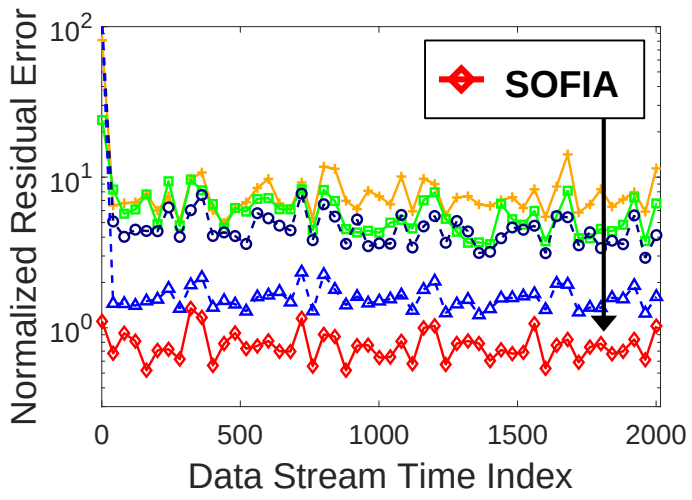
Road Map

- Introduction
- Problem Definition
- Proposed Method: **SOFIA**
  - Tensor Factorization Model
  - Optimization Algorithm
- Experiment Results
- **Conclusions** ⬅

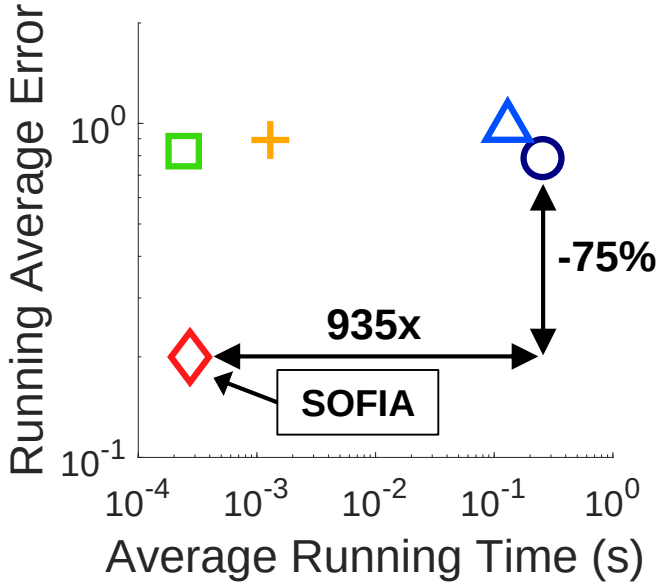
Conclusions

- We propose **SOFIA**, a streaming factorization algorithm for real-world tensors with missing entries and outliers.

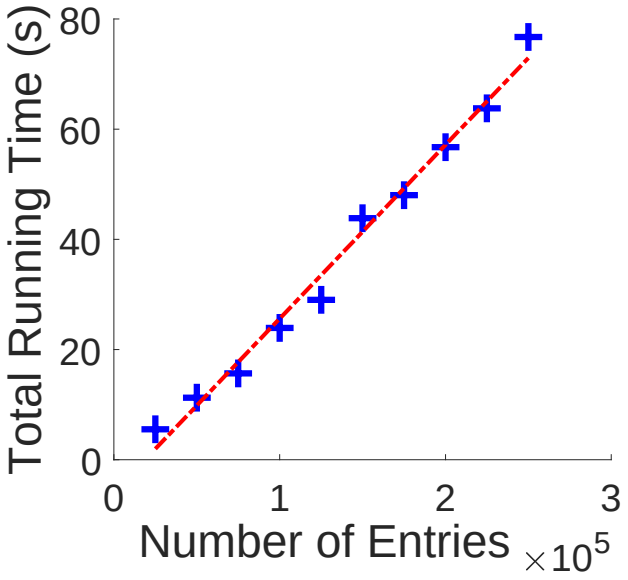
Robust and Accurate



Fast

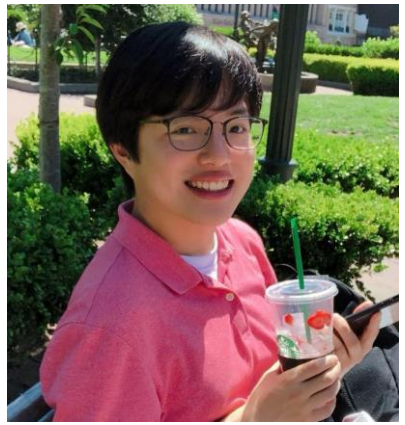


Scalable



Robust Factorization of Real-world Tensor Streams with Patterns, Missing Values, and Outliers

Dongjin Lee



Kijung Shin