SCED: A General Framework for Sparse Tensor Decomposition with Constraints and Elementwise Dynamic Learning

Shuo Zhou, Sarah M. Erfani and James Bailey

School of Computing and Information Systems

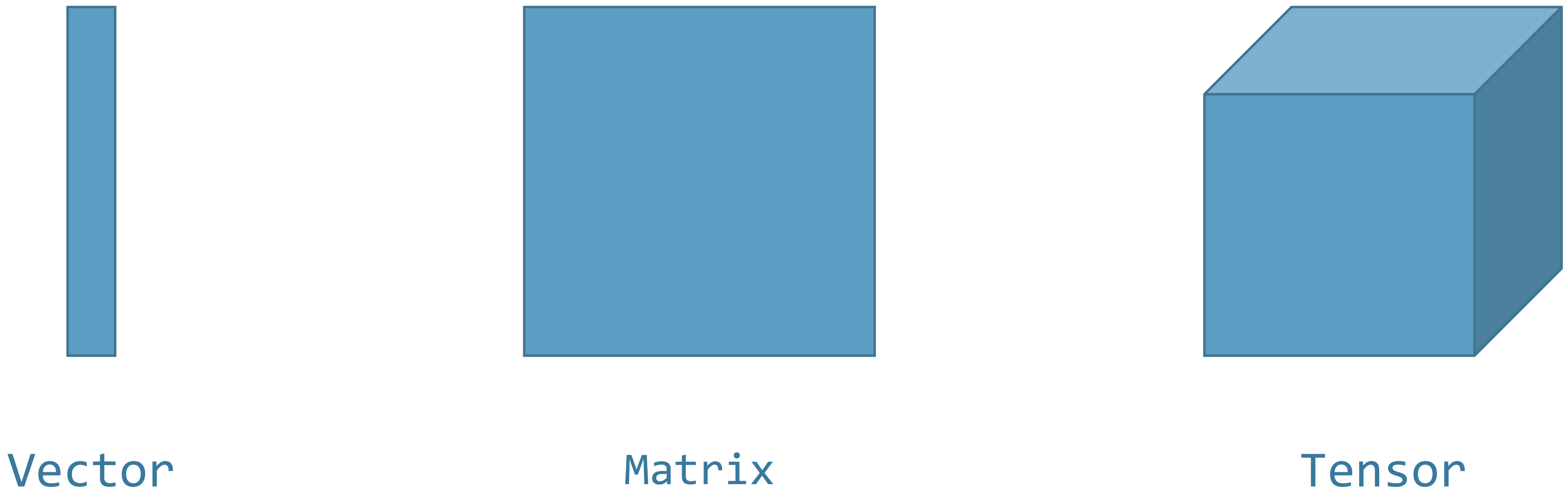
The University of Melbourne

ICDM’17 New Orleans

Outline

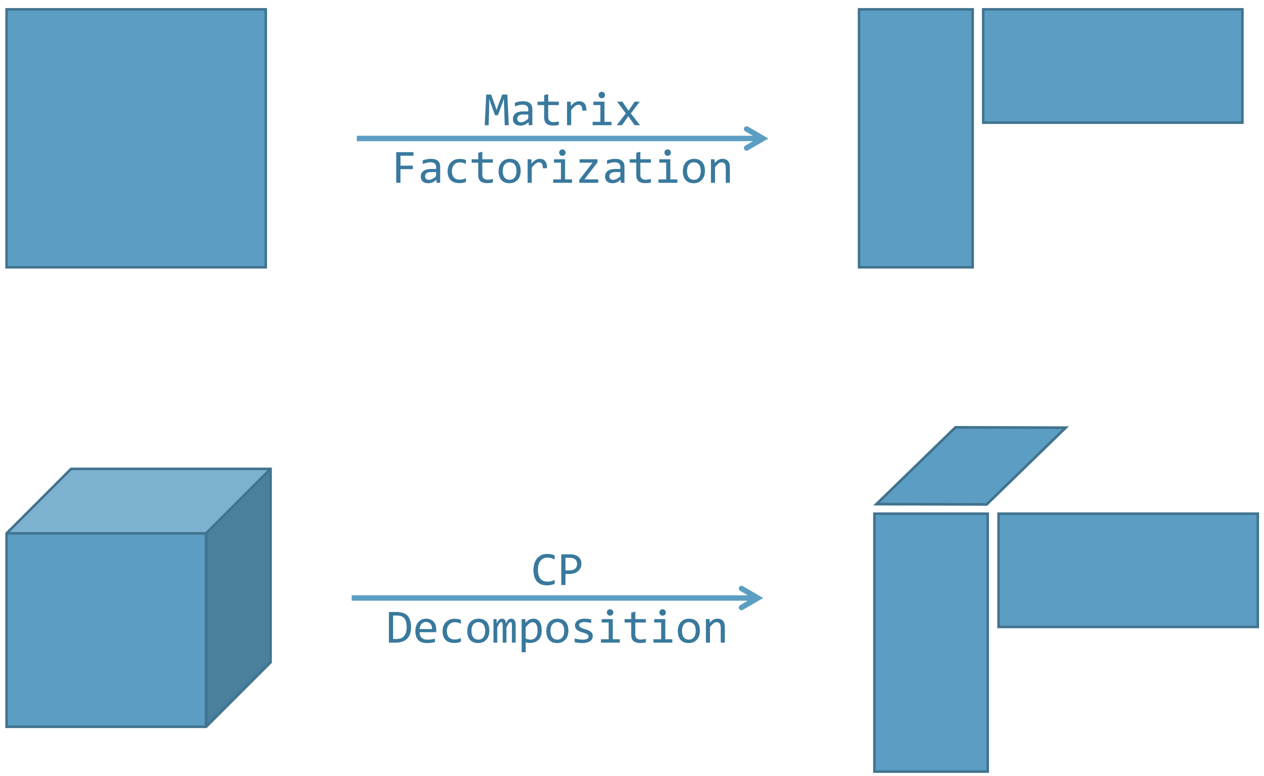
Introduction

Tensor



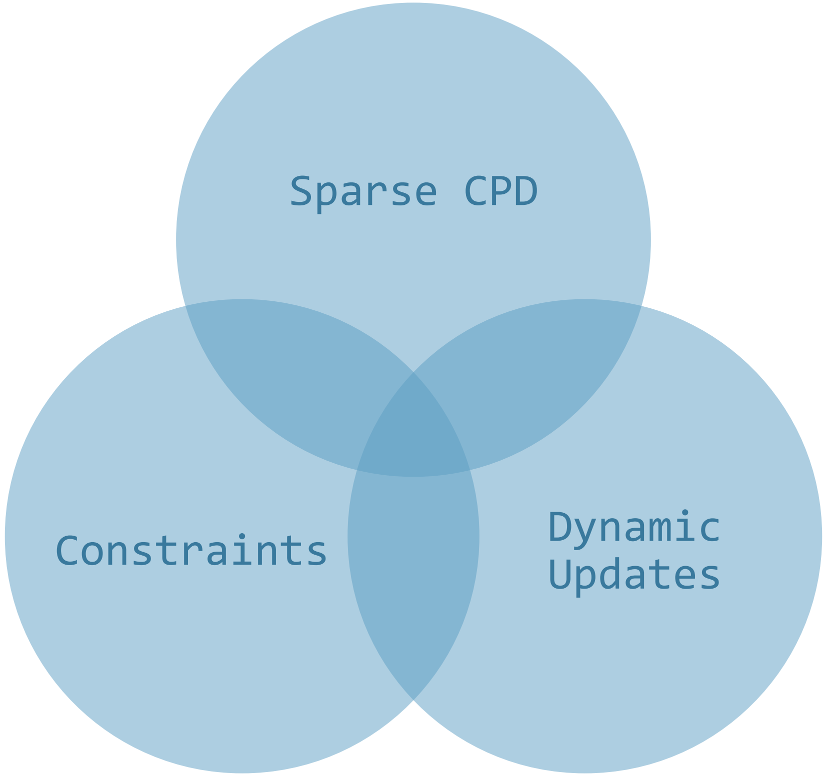
Tensors (multi-way array) are a natural representation for multi-dimensional data, e.g., videos, time-evolving networks

CP Decomposition



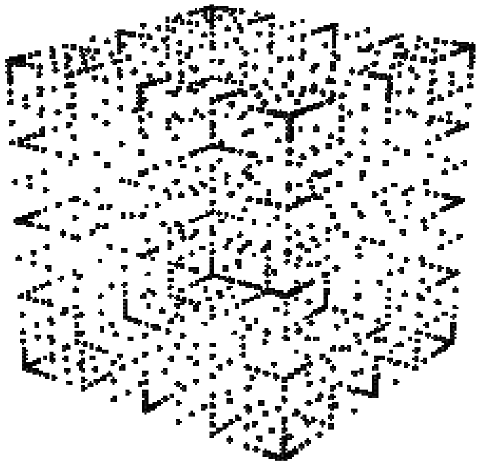
CP decomposition is a method to simplify and summarize tensors

Research Question



Desirable properties of a tensor decomposition

Sparse Tensors



- Role of zero: 3 possible cases
  1) True observations: equally important to non-zeros

     e.g., intersection × intersection × time
  2) Missing values: no contribution

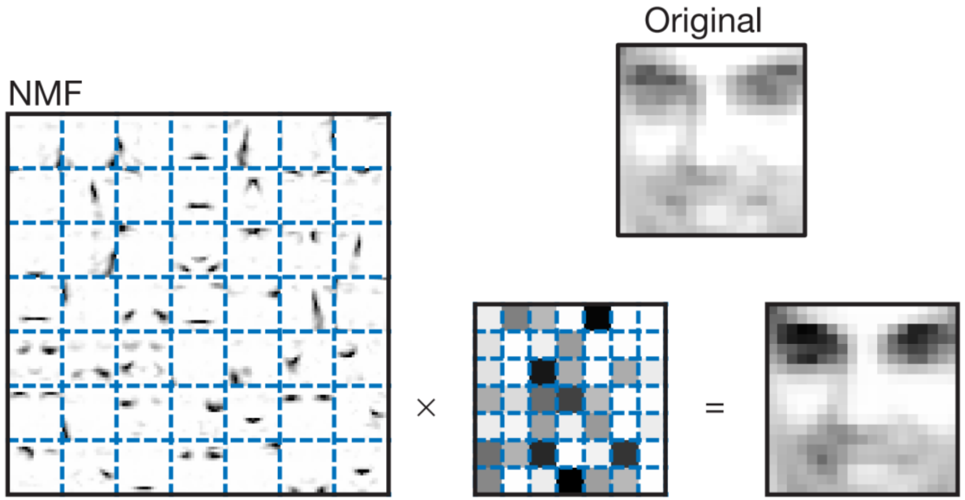
     e.g., context-aware recommender systems with explicit ratings
  3) Implicit information: less important than observations

     e.g., context-aware recommender systems with implicit ratings (clicks on ads)

Image source: http://wissrech.ins.uni-bonn.de/research/projects/zumbusch/sparse.html

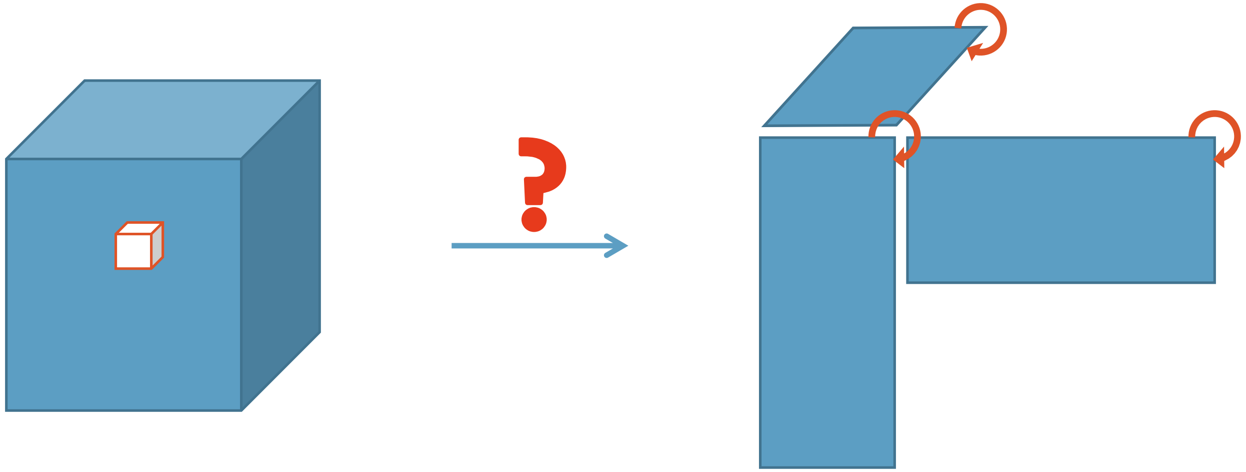
Constrained Decomposition

- Usually provides interpretable and meaningful results
- Non-negativity
  - Additive contributions of factors
- Regularizations
  - Sparseness: L1 norm
  - Prevent overfitting: Frobenius norm



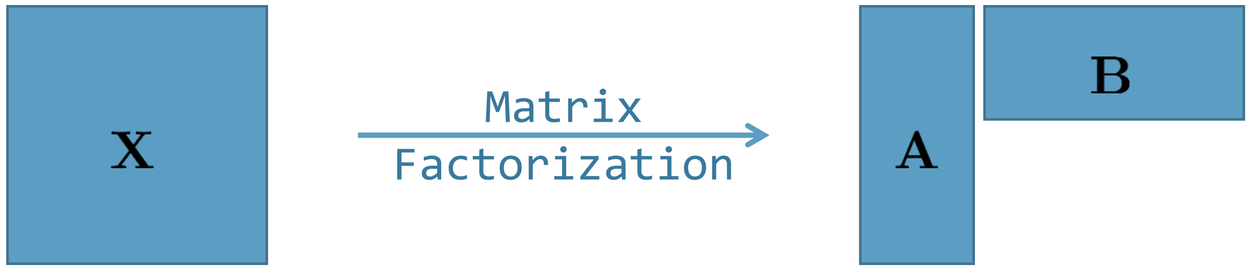
Dynamic Learning

- When new ***cell entries*** arrive, how to efficiently update an existing decomposition?
  - e.g., recommender systems, new ratings



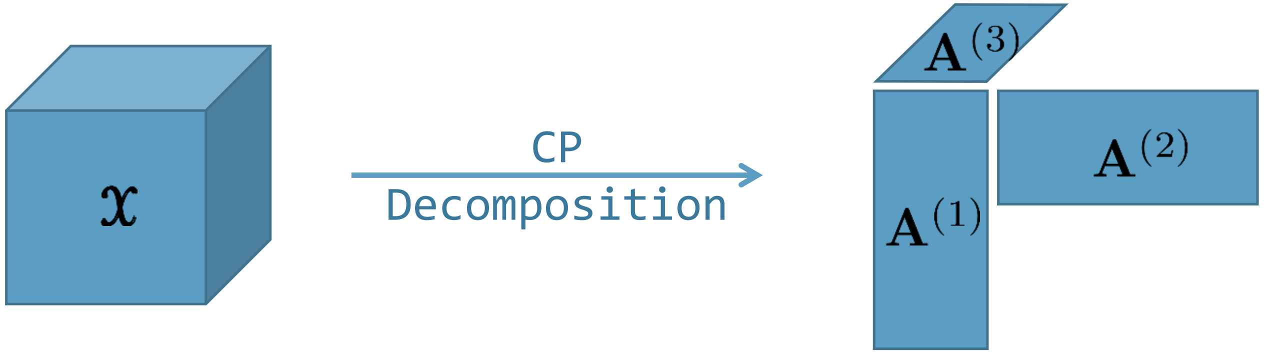
Preliminaries

CP Decomposition



$$\hat{x}_{ij} = \sum_{r=1}^{R} a_{ir} b_{jr},$$

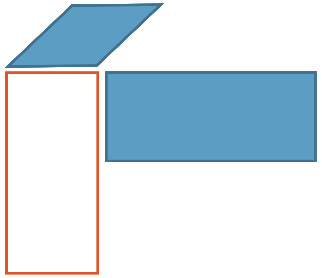
$$Loss = \frac{1}{2} \sum\nolimits_{\Omega} (x_{ij} - \hat{x}_{ij})^2$$



$$\hat{x}_{i_1,\ldots,i_N} = \sum_{r=1}^{R} \prod_{n=1}^{N} a^{(n)}_{i_n r},$$

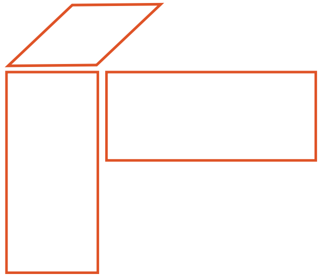
$$Loss = \frac{1}{2} \sum\nolimits_{\Omega} (x_{i_1,\ldots,i_N} - \hat{x}_{i_1,\ldots,i_N})^2$$

Existing Methods



ALS

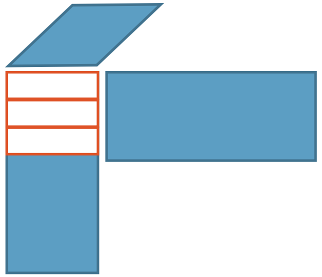
$$Loss = \frac{1}{2}\sum\nolimits_{\Omega}(x_{\mathbf{i}} - \hat{x}_{\mathbf{i}})^2$$



WOPT

$$Loss = \frac{1}{2}\sum\nolimits_{\Omega} w_{\mathbf{i}}(x_{\mathbf{i}} - \hat{x}_{\mathbf{i}})^2$$

$$w_{\mathbf{i}} = \begin{cases} 0 & \text{if } x_{\mathbf{i}} \text{ is missing} \\ 1 & \text{otherwise} \end{cases}$$



iTALS

$$Loss = \frac{1}{2}\sum\nolimits_{\Omega} w_{\mathbf{i}}(x_{\mathbf{i}} - \hat{x}_{\mathbf{i}})^2$$

$$w_{\mathbf{i}} = \begin{cases} 1 & \text{if } x_{\mathbf{i}} = 0 \\ \alpha > 1 & \text{otherwise} \end{cases}$$

	ALS	WOPT	iTALS
Efficiency & Scalability	✓	✓	
General	True Observations	Missing Values	Implicit Information
Constraints	✓		
Dynamic			✓

Our Approach

A uniform formulation that covers all 3 cases

True Observations
($\alpha = 1$)

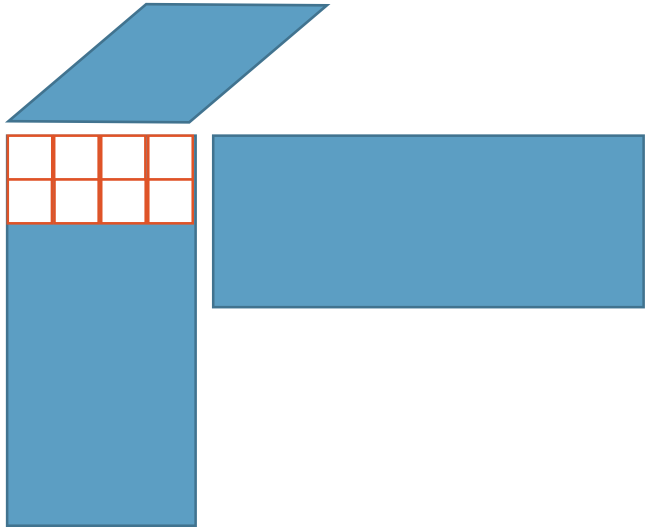
$$Loss = \frac{1}{2}\sum_{\Omega} w_{\mathbf{i}}(x_{\mathbf{i}} - \hat{x}_{\mathbf{i}})^2$$

$$w_{\mathbf{i}} = \begin{cases} \alpha \in [0,1] & \text{if } x_{\mathbf{i}} = 0 \\ 1 & \text{otherwise} \end{cases}$$

Missing Values
($\alpha = 0$)

Implicit
Information
($\alpha \in (0, 1)$)

SCED Algorithm



- Update one parameter at a time by fixing all others
- Good convergence
- Linear complexity w.r.t. the number of non-zeros *and decomposition rank*
- Easy to incorporate constraints
- Suitable for dynamic learning

Time Complexity

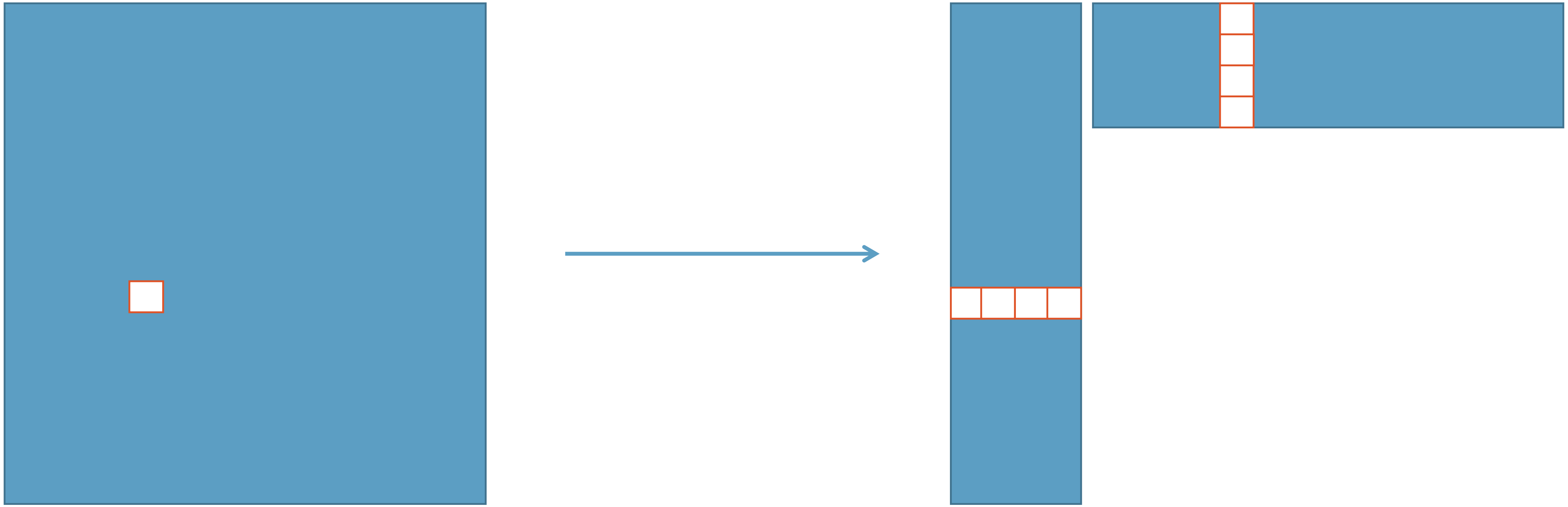
	SCED	Baseline
True Observations ($\alpha = 1$)	$\mathcal{O}(\|\Omega^+\|R)$	
Missing Values ($\alpha = 0$)		
Implicit Information ($\alpha \in (0, 1)$)	$\mathcal{O}(\|\Omega^+\|R)$	$\mathcal{O}(\|\Omega^+\|R^2)$

$|\Omega^+|$: the number of non-zeros
R: the decomposition rank

Constraints

- Non-negativity
  - set all negative updates to ε (ε is non-negative and close to zero)
- Regularizations
  - $$Loss + \sum_{n=1}^{N} \lambda_n \phi_n(\mathbf{A}^{(n)})$$

Dynamic Learning



A new entry will only significantly affect its corresponding rows in loading matrices

Experiments

Setup

- E1: Performance under **static** setting
  - 90% training, 10% testing
- E2: Performance under **dynamic** setting
  - 80% training, 10% dynamic, 10% testing
- E3: Scalability Performance
- Evaluation metrics
  - Efficiency
    - Running time in seconds
  - Effectiveness
    - RMSE and Fitness

$$fitness \triangleq \left(1 - \frac{\left\|\hat{\mathcal{X}} - \mathcal{X}\right\|_F}{\left\|\mathcal{X}\right\|_F}\right)$$

Datasets

- Synthetic datasets
  - SYN-TO, SYN-MV, SYN-II
  - 200*200*200
  - R = 20
  - Density = 1%
- Real-world datasets

Datasets	Size	$\lvert\Omega^+\rvert$	Density
MovieLens[1]	6040*3952*1040	$1 * 10^6$	$4 * 10^{-5}$
LastFM[2]	991*1000*168	$2.9 * 10^6$	$1.7 * 10^{-2}$
MathOverflow[3]	24818*24818*2351	$4 * 10^5$	$2.7 * 10^{-7}$

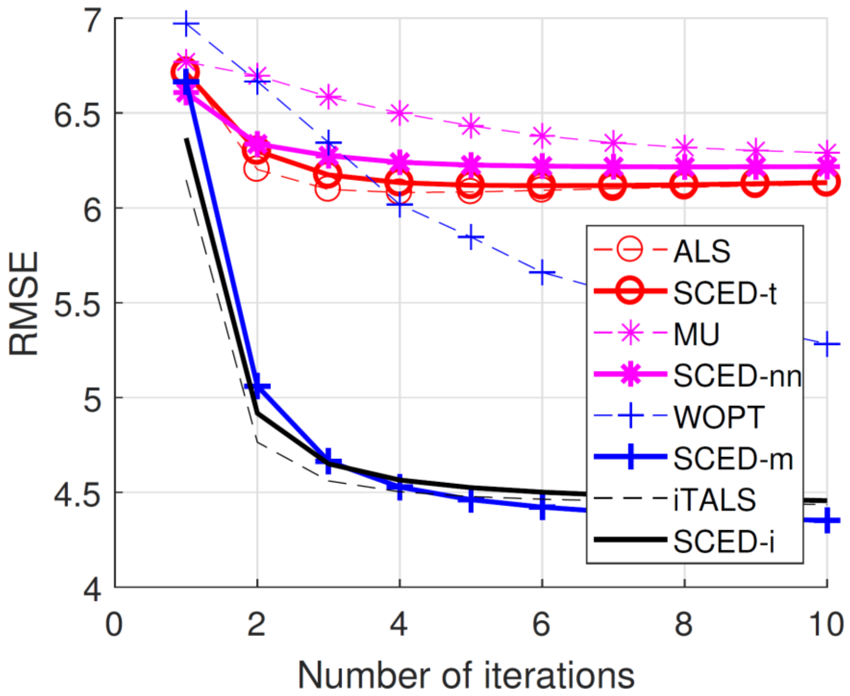
1 www.movielens.org
2 www.last.fm
3 mathoverflow.net

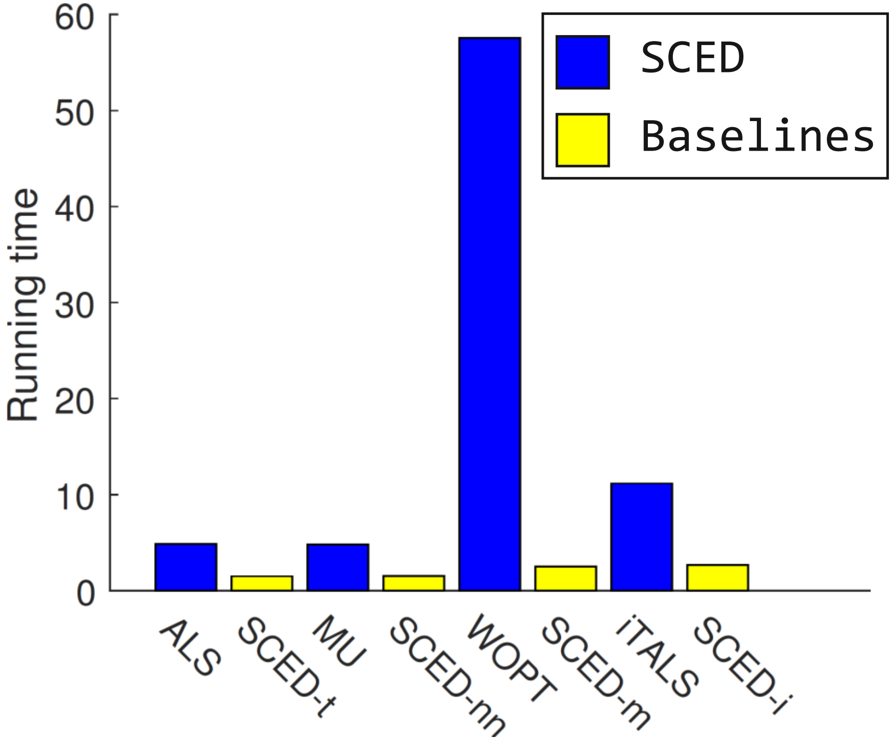
Baselines

- ALS (Bader & Kolda *et al.*, 2015)
  - For TO case
  - Update one loading matrix by fixing others
- MU (Bader & Kolda *et al.*, 2015)
  - For TO case
  - Non-negative CP decomposition based on multiplicative update rule
- WOPT (Acar *et al.*, 2011)
  - For MV case
  - Update all parameter by optimization
- iTALS (Hidasi *et al.*, 2012)
  - For II case
  - Row-wise update

Results – Static Setting

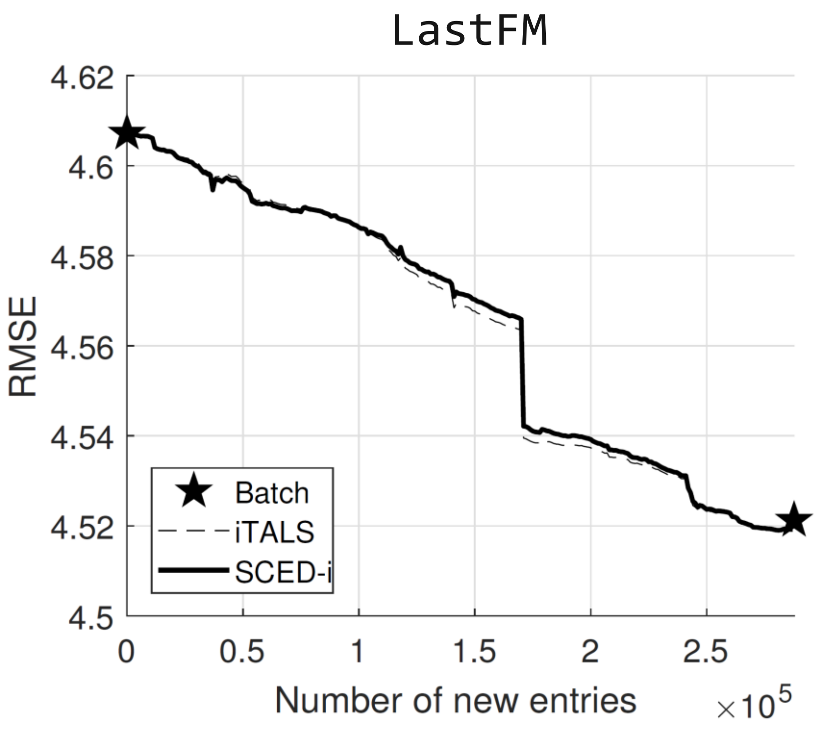
LastFM





- Better convergence over MU and WOPT
- Assigning small weight to zero entries is helpful
- Improved better efficiency compared to state-of-the-arts

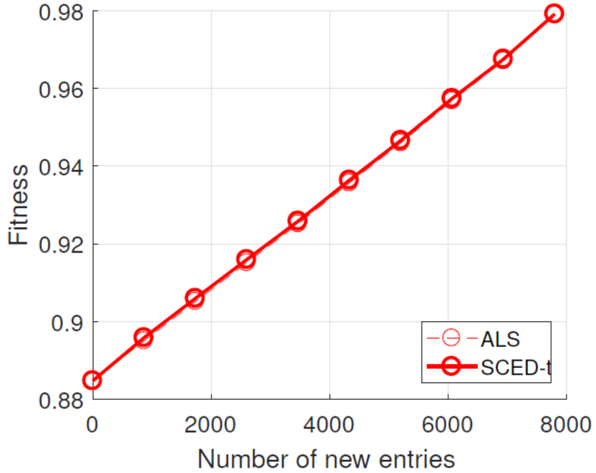
Results – Dynamic Setting



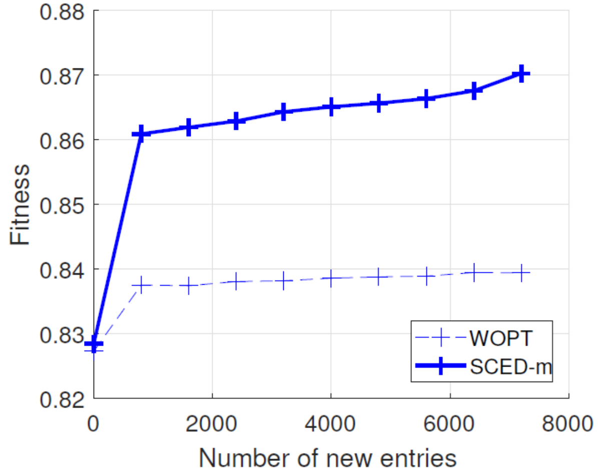
	Running Time (seconds)	Speedup
Batch	490	25,675x
iTALS	0.0484	2.6x
SCED	0.0185	

- Batch has no latest decomposition
- Highly effective dynamic update schema
- Speeding up batch method by ~26K times

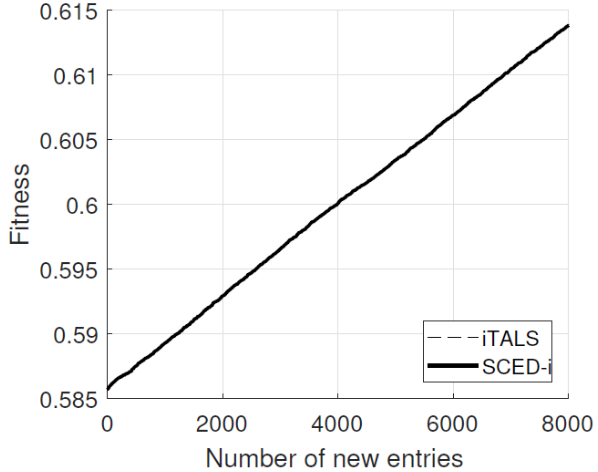
SCED can work with all type of data dynamically



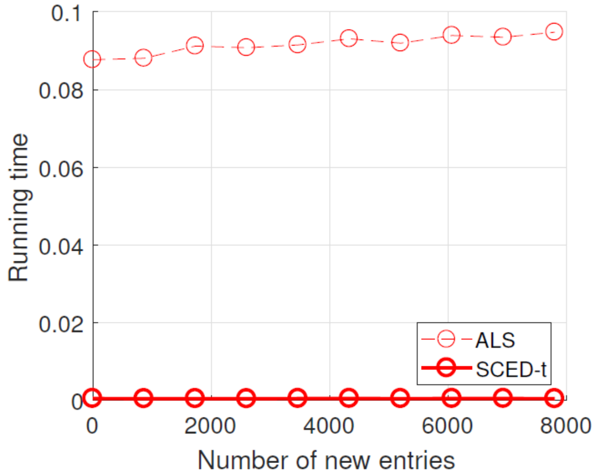
zeros are true observations

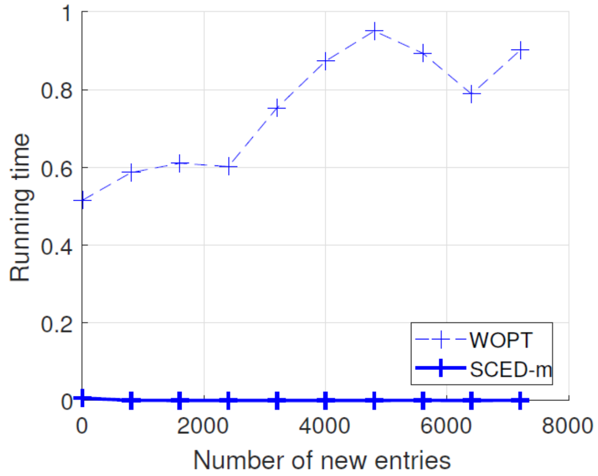


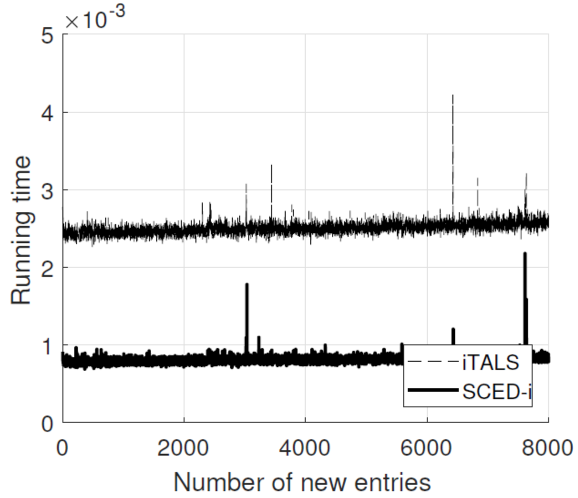
zeros are missing values



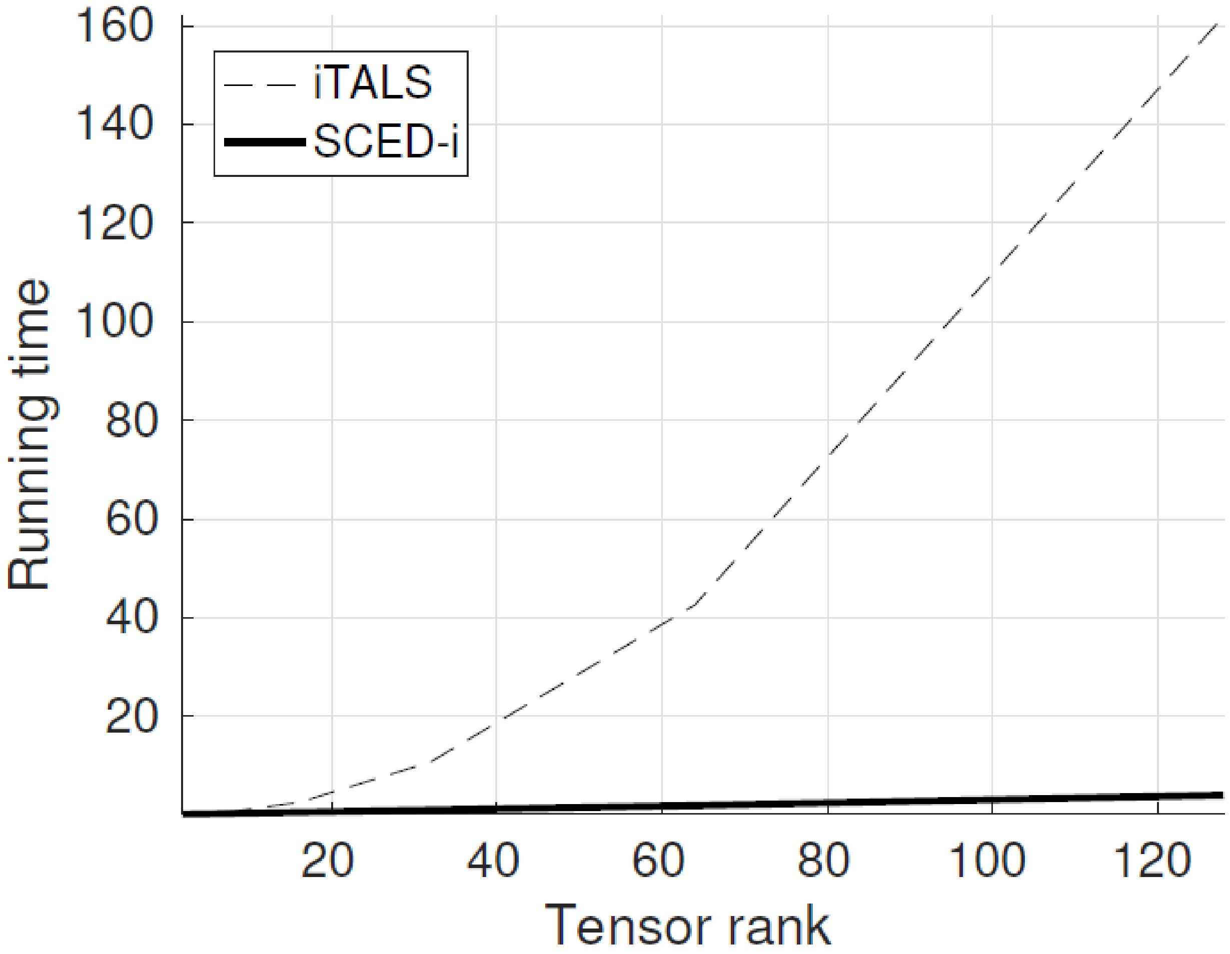
zeros are implicit information







Results - Scalability



Conclusions

Conclusions

- A unified formulation for sparse CPD
- An efficient and scalable algorithm
  - Able to incorporate constraints
  - Can dynamically track the new CPD on-the-fly.
- Significant performance gains in effectiveness, efficiency and scalability
  - Good convergence and better efficiency for the static case
  - Highly effective dynamic update schema (>20K times speedup)

If you have a sparse tensor,
try SCED!

Future Work

- Relationships to existing online tensor decomposition
- Handling other types of dynamic cases
  - e.g., deletions and updating
- Apply to real-world applications with domain-specified constraints (e.g., recommender systems)

Q & A