Hybrid-Parallel Parameter Estimation for Frequentist and Bayesian Models

Parameswaran Raman

Ph.D. Dissertation Defense

University of California, Santa Cruz

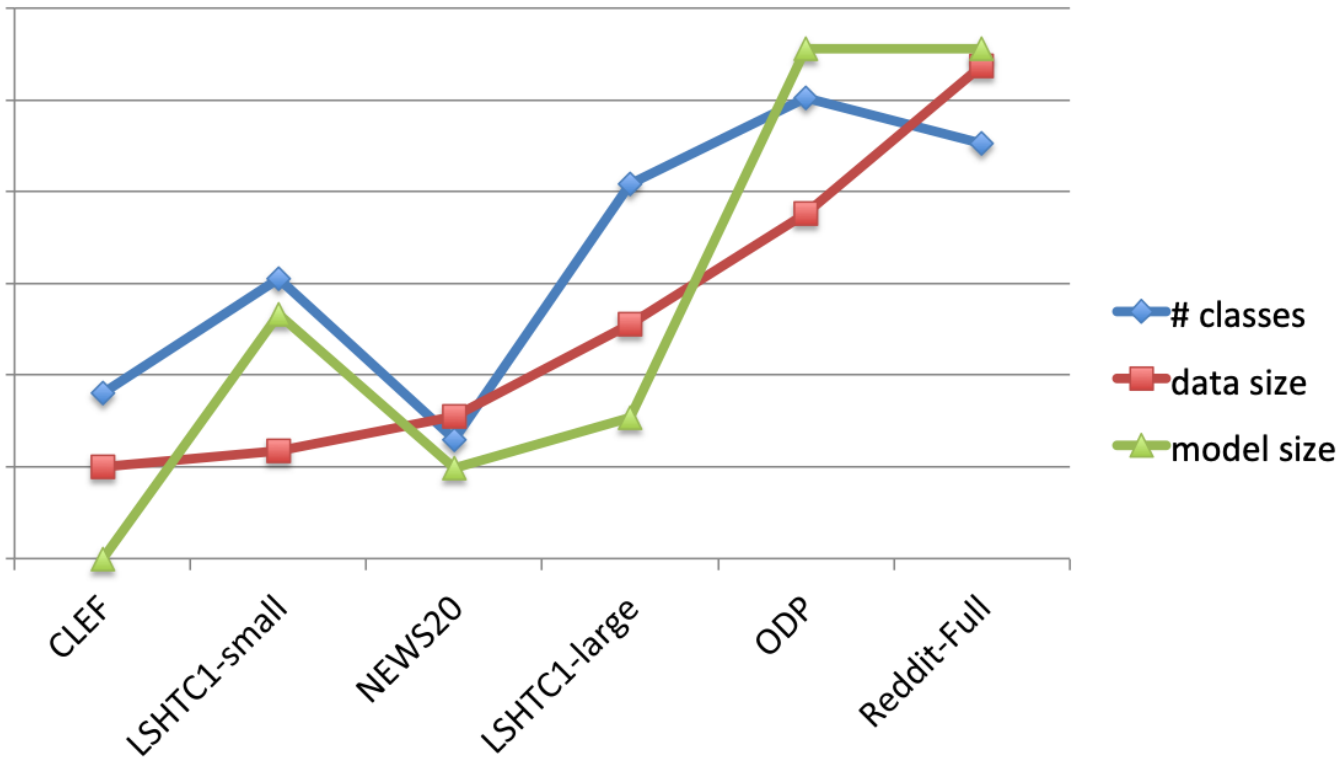
December 2, 2019

Motivation

Data and Model grow hand in hand

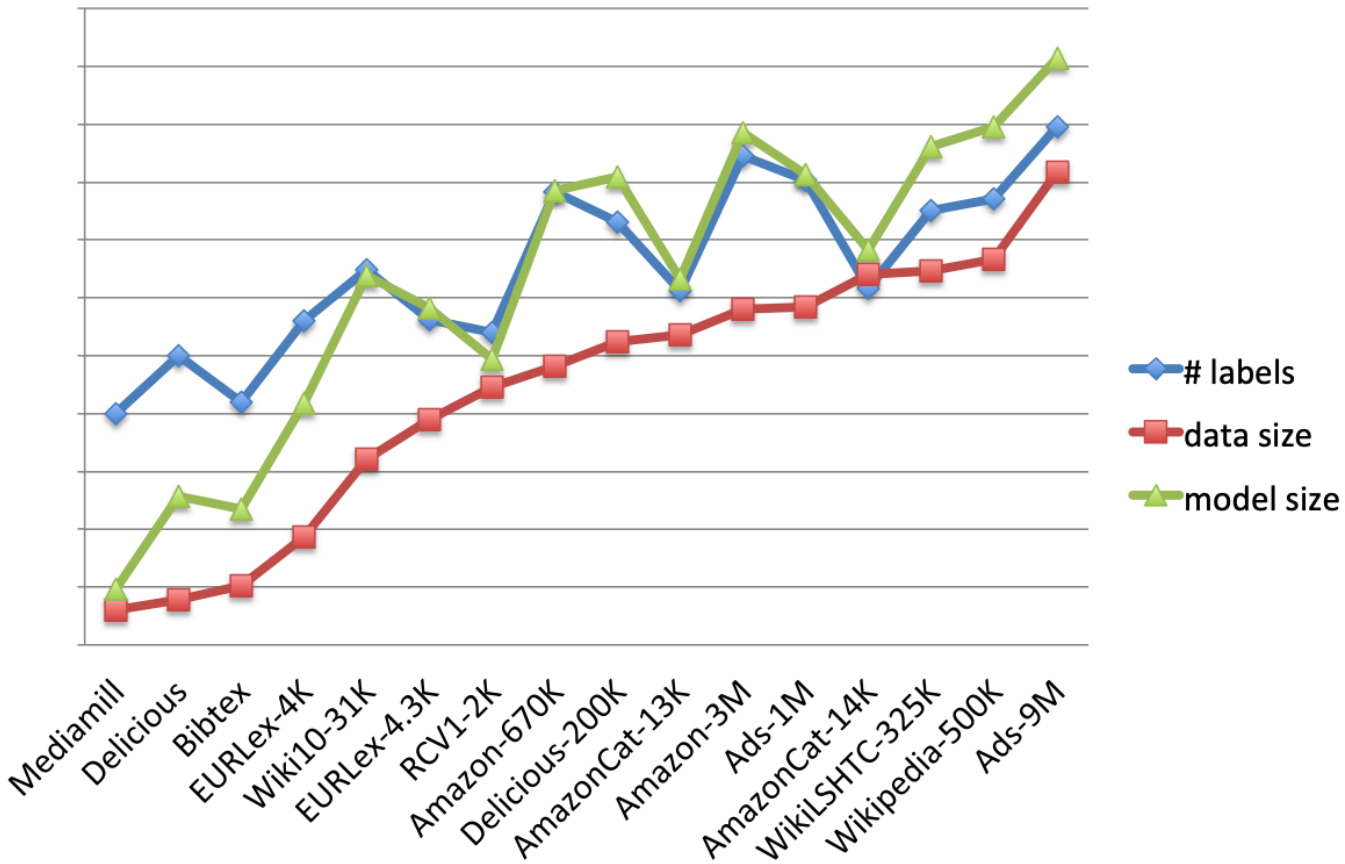
Data and Model grow hand in hand

e.g. Extreme Multi-class classification



Data and Model grow hand in hand

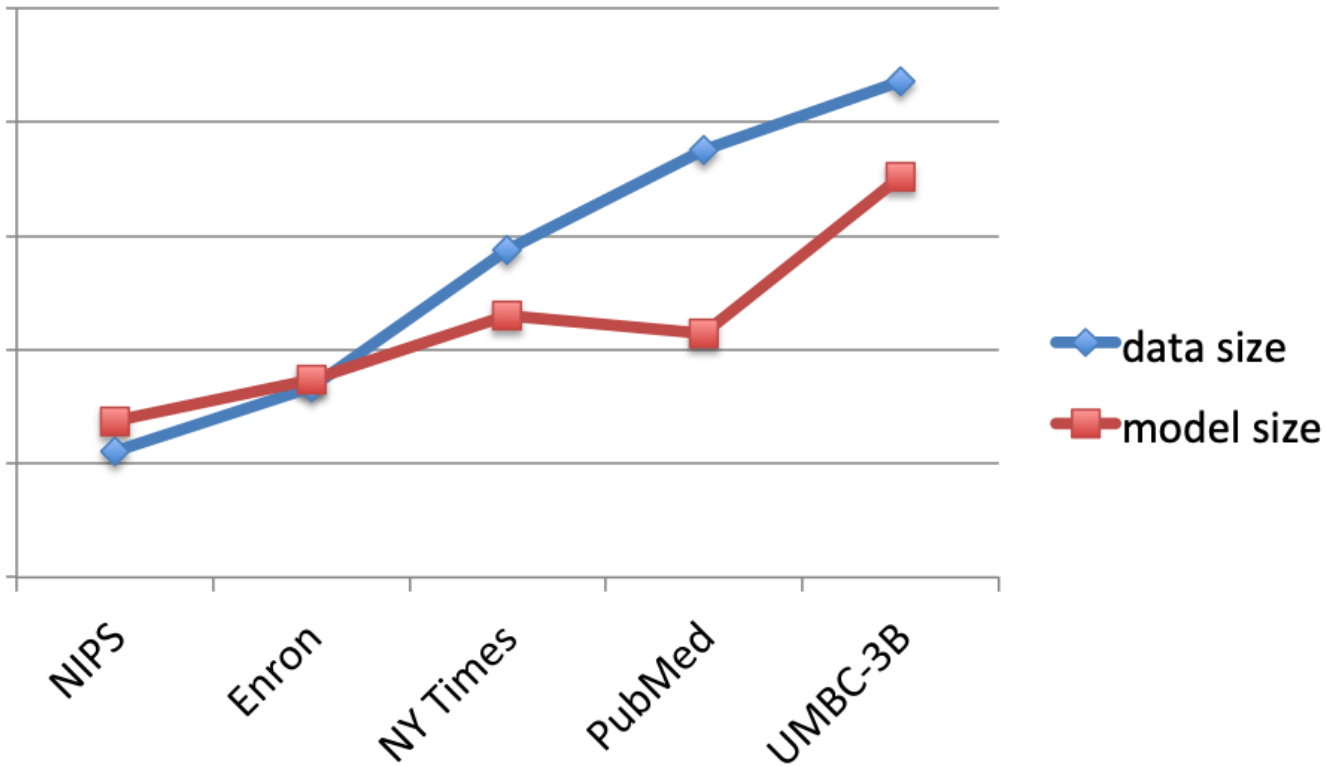
e.g. Extreme Multi-label classification



http://manikvarma.org/downloads/XC/XMLRepository.html

Data and Model grow hand in hand

e.g. Topic Models



Challenges in Parameter Estimation

Challenges in Parameter Estimation

1. Storage limitations of Data and Model

Challenges in Parameter Estimation

1. Storage limitations of Data and Model
2. Interdependence in parameter updates

Challenges in Parameter Estimation

1. Storage limitations of Data and Model
2. Interdependence in parameter updates
3. Bulk-Synchronization is expensive

Challenges in Parameter Estimation

1. Storage limitations of Data and Model
2. Interdependence in parameter updates
3. Bulk-Synchronization is expensive
4. Synchronous communication is inefficient

Challenges in Parameter Estimation

1. Storage limitations of Data and Model
2. Interdependence in parameter updates
3. Bulk-Synchronization is expensive
4. Synchronous communication is inefficient

Traditional distributed machine learning approaches fall short.

Challenges in Parameter Estimation

1. Storage limitations of Data and Model
2. Interdependence in parameter updates
3. Bulk-Synchronization is expensive
4. Synchronous communication is inefficient

Traditional distributed machine learning approaches fall short.

Hybrid-Parallel algorithms for parameter estimation!

Outline

1. Introduction
2. Distributed Parameter Estimation
3. Thesis Roadmap
   - Multinomial Logistic Regression (KDD 2019)
   - Mixture Models (AISTATS 2019)
   - Latent Collaborative Retrieval (NIPS 2014)
4. Conclusion and Future Directions
5. Appendix

Regularized Risk Minimization

Goals in machine learning

Regularized Risk Minimization

Goals in machine learning

- We want to build a model using observed (training) data

Regularized Risk Minimization

Goals in machine learning

- We want to build a model using observed (training) data
- Our model must generalize to unseen (test) data

Regularized Risk Minimization

Goals in machine learning

- We want to build a model using observed (training) data
- Our model must generalize to unseen (test) data

$$\min_{\theta} \mathcal{L}(\theta) := \lambda \underbrace{\mathcal{R}(\theta)}_{\text{regularizer}} + \underbrace{\frac{1}{N} \sum_{i=1}^{N} loss(\mathbf{x}_i, y_i, \theta)}_{\text{empirical risk}}$$

Regularized Risk Minimization

Goals in machine learning

- We want to build a model using observed (training) data
- Our model must generalize to unseen (test) data

$$\min_{\theta} \mathcal{L}(\theta) := \lambda \underbrace{\mathcal{R}(\theta)}_{\text{regularizer}} + \underbrace{\frac{1}{N} \sum_{i=1}^{N} loss(\mathbf{x}_i, y_i, \theta)}_{\text{empirical risk}}$$

- $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, $\mathbf{y} = \{y_1, \ldots, y_N\}$ is the observed training data
- θ are the model parameters

Regularized Risk Minimization

Goals in machine learning

- We want to build a model using observed (training) data
- Our model must generalize to unseen (test) data

$$\min_{\theta} \mathcal{L}(\theta) := \lambda \underbrace{\mathcal{R}(\theta)}_{\text{regularizer}} + \underbrace{\frac{1}{N} \sum_{i=1}^{N} loss(\mathbf{x}_i, y_i, \theta)}_{\text{empirical risk}}$$

- $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, $\mathbf{y} = \{y_1, \ldots, y_N\}$ is the observed training data
- θ are the model parameters
- $loss(\cdot)$ to quantify model's performance

Regularized Risk Minimization

Goals in machine learning

- We want to build a model using observed (training) data
- Our model must generalize to unseen (test) data

$$\min_{\theta} \mathcal{L}(\theta) := \lambda \underbrace{\mathcal{R}(\theta)}_{\text{regularizer}} + \underbrace{\frac{1}{N} \sum_{i=1}^{N} loss(\mathbf{x}_i, y_i, \theta)}_{\text{empirical risk}}$$

- $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, $\mathbf{y} = \{y_1, \ldots, y_N\}$ is the observed training data
- θ are the model parameters
- $loss(\cdot)$ to quantify model's performance
- regularizer $\mathcal{R}(\theta)$ to avoid over-fitting (penalizes complex models)

Frequentist Models

Frequentist Models

- Binary Classification/Regression
- Multinomial Logistic Regression
- Matrix Factorization
- Latent Collaborative Retrieval
- Polynomial Regression, Factorization Machines

Bayesian Models

Bayesian Models

$$\underbrace{p(\theta|\mathbf{X})}_{\textit{posterior}} = \frac{\overbrace{p(\mathbf{X}|\theta)}^{\textit{likelihood}} \cdot \overbrace{p(\theta)}^{\textit{prior}}}{\underbrace{\int p(\mathbf{X}, \theta) d\theta}_{\text{marginal likelihood (model evidence)}}}$$

- prior plays the role of regularizer $\mathcal{R}(\theta)$
- likelihood plays the role of empirical risk

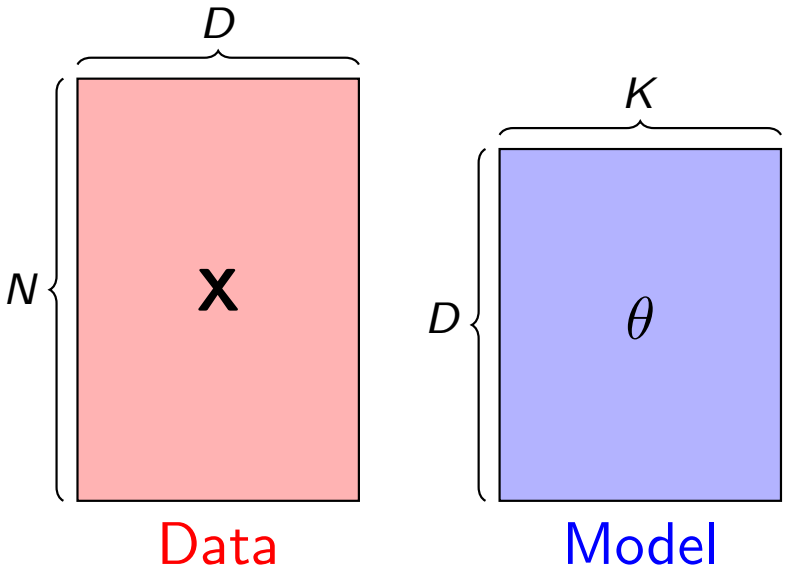
Bayesian Models

- Gaussian Mixture Models (GMM)
- Latent Dirichlet Allocation (LDA)

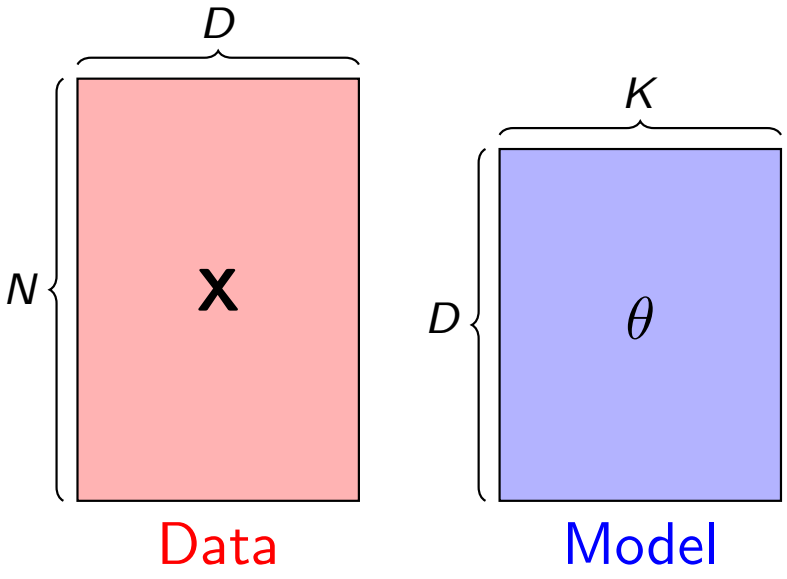
$$\underbrace{p(\theta|\mathbf{X})}_{\text{posterior}} = \frac{\overbrace{p(\mathbf{X}|\theta)}^{\text{likelihood}} \cdot \overbrace{p(\theta)}^{\text{prior}}}{\underbrace{\int p(\mathbf{X}, \theta) d\theta}_{\text{marginal likelihood (model evidence)}}}$$

- prior plays the role of regularizer $\mathcal{R}(\theta)$
- likelihood plays the role of empirical risk

Focus on Matrix Parameterized Models



Focus on Matrix Parameterized Models



What if these matrices do not fit in memory?

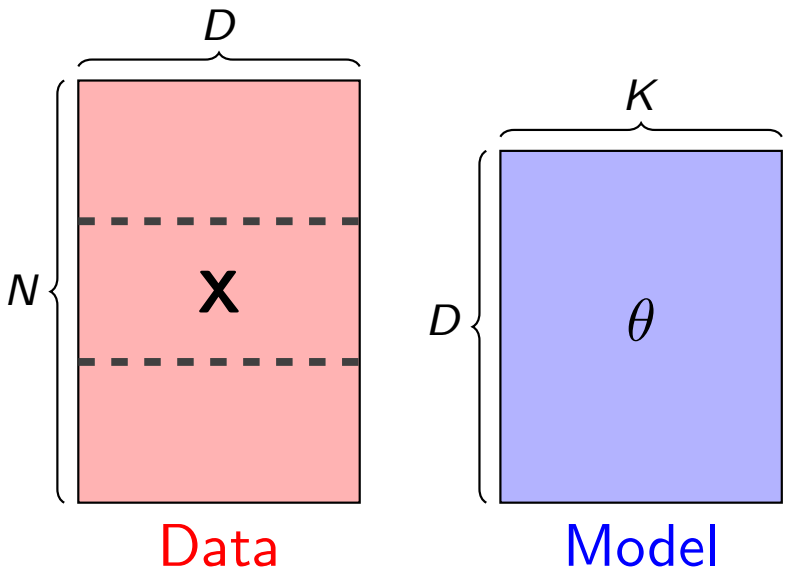
Outline

1. Introduction
2. Distributed Parameter Estimation
3. Thesis Roadmap
   - Multinomial Logistic Regression (KDD 2019)
   - Mixture Models (AISTATS 2019)
   - Latent Collaborative Retrieval (NIPS 2014)
4. Conclusion and Future Directions
5. Appendix

Distributed Parameter Estimation

Distributed Parameter Estimation

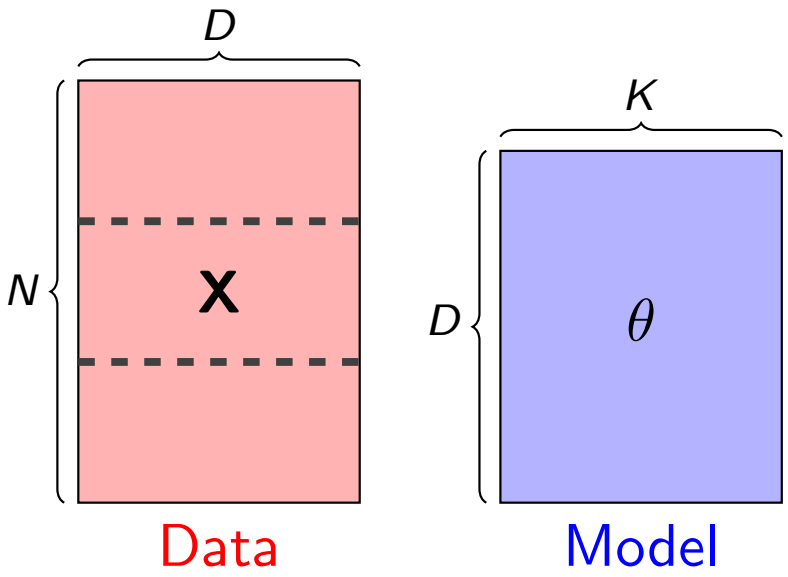
Data parallel



e.g. L-BFGS

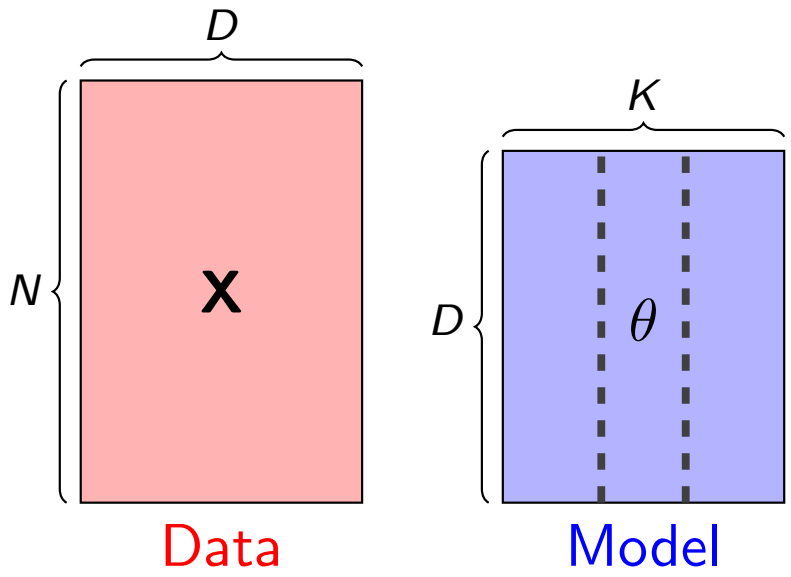
Distributed Parameter Estimation

Data parallel



e.g. L-BFGS

Model parallel



e.g. LC [Gopal et al., 2013]

Distributed Parameter Estimation

Good

- Easy to implement using map-reduce
- Scales as long as Data or Model fits in memory

Distributed Parameter Estimation

Good

- Easy to implement using map-reduce
- Scales as long as Data or Model fits in memory

Bad

Either Data or the Model is replicated on each worker.

Distributed Parameter Estimation

Good

- Easy to implement using map-reduce
- Scales as long as Data or Model fits in memory

Bad

Either Data or the Model is replicated on each worker.

- Data Parallel: Each worker requires $\mathcal{O}\left(\frac{N \times D}{P}\right) + \underbrace{\mathcal{O}\left(K \times D\right)}_{\text{bottleneck}}$

Distributed Parameter Estimation

Good

- Easy to implement using map-reduce
- Scales as long as Data or Model fits in memory

Bad

Either Data or the Model is replicated on each worker.

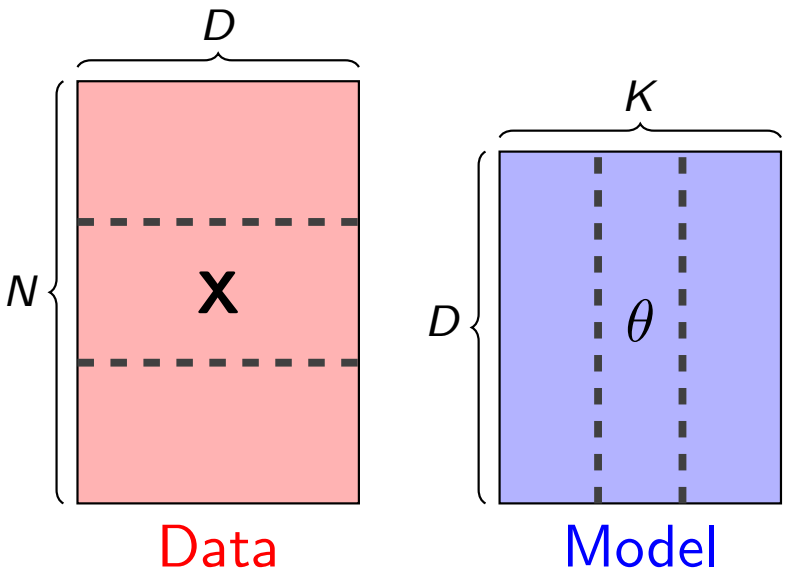
- Data Parallel: Each worker requires $\mathcal{O}\left(\frac{N \times D}{P}\right) + \underbrace{\mathcal{O}\left(K \times D\right)}_{\text{bottleneck}}$
- Model Parallel: Each worker requires requires $\underbrace{\mathcal{O}\left(N \times D\right)}_{\text{bottleneck}} + \mathcal{O}\left(\frac{K \times D}{P}\right)$

Distributed Parameter Estimation

Question

Can we get the best of both worlds?

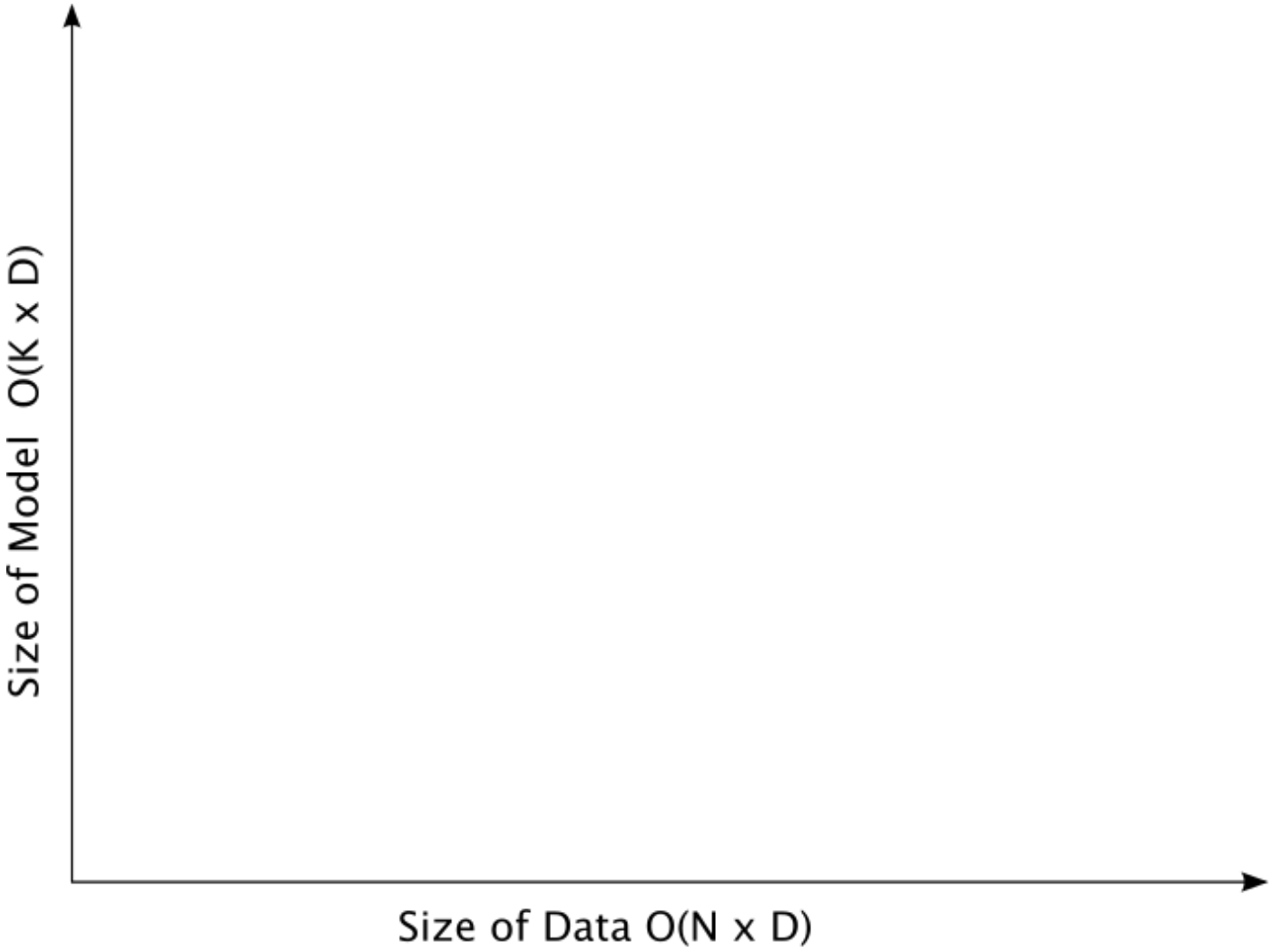
Hybrid Parallelism



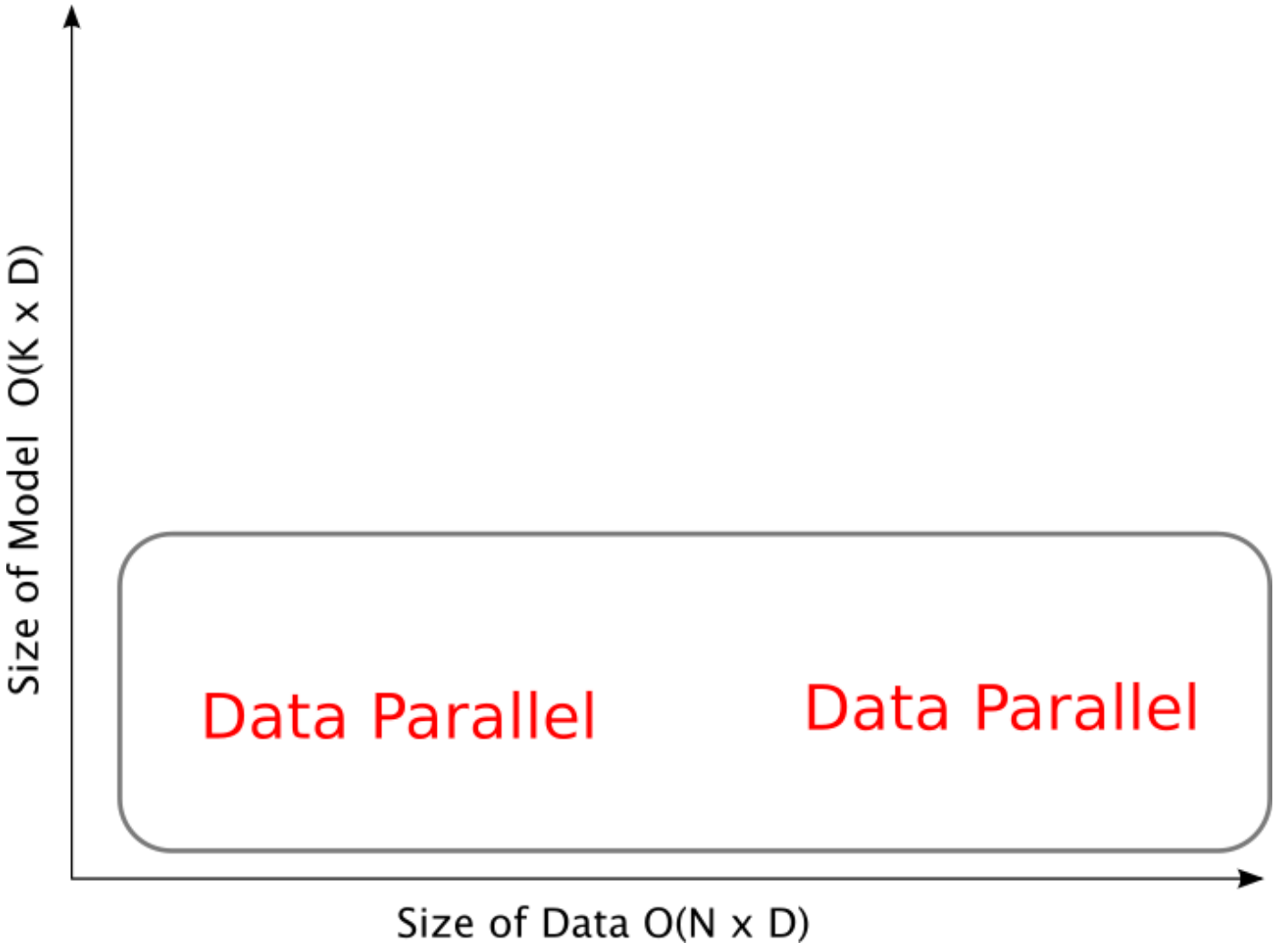
Benefits of Hybrid-Parallelism

1. **One versatile method** for all regimes of data and model parallelism

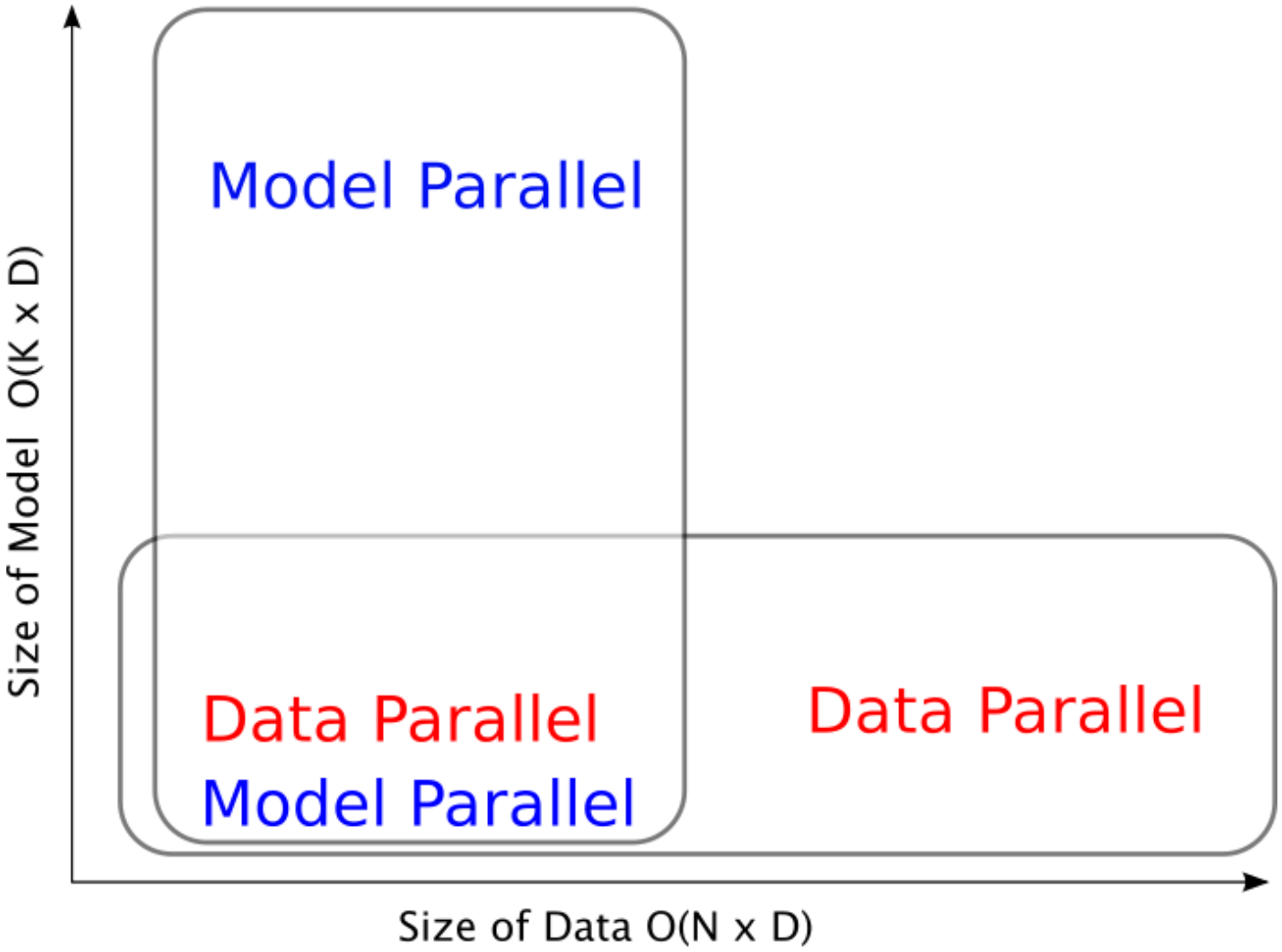
Why Hybrid Parallelism?



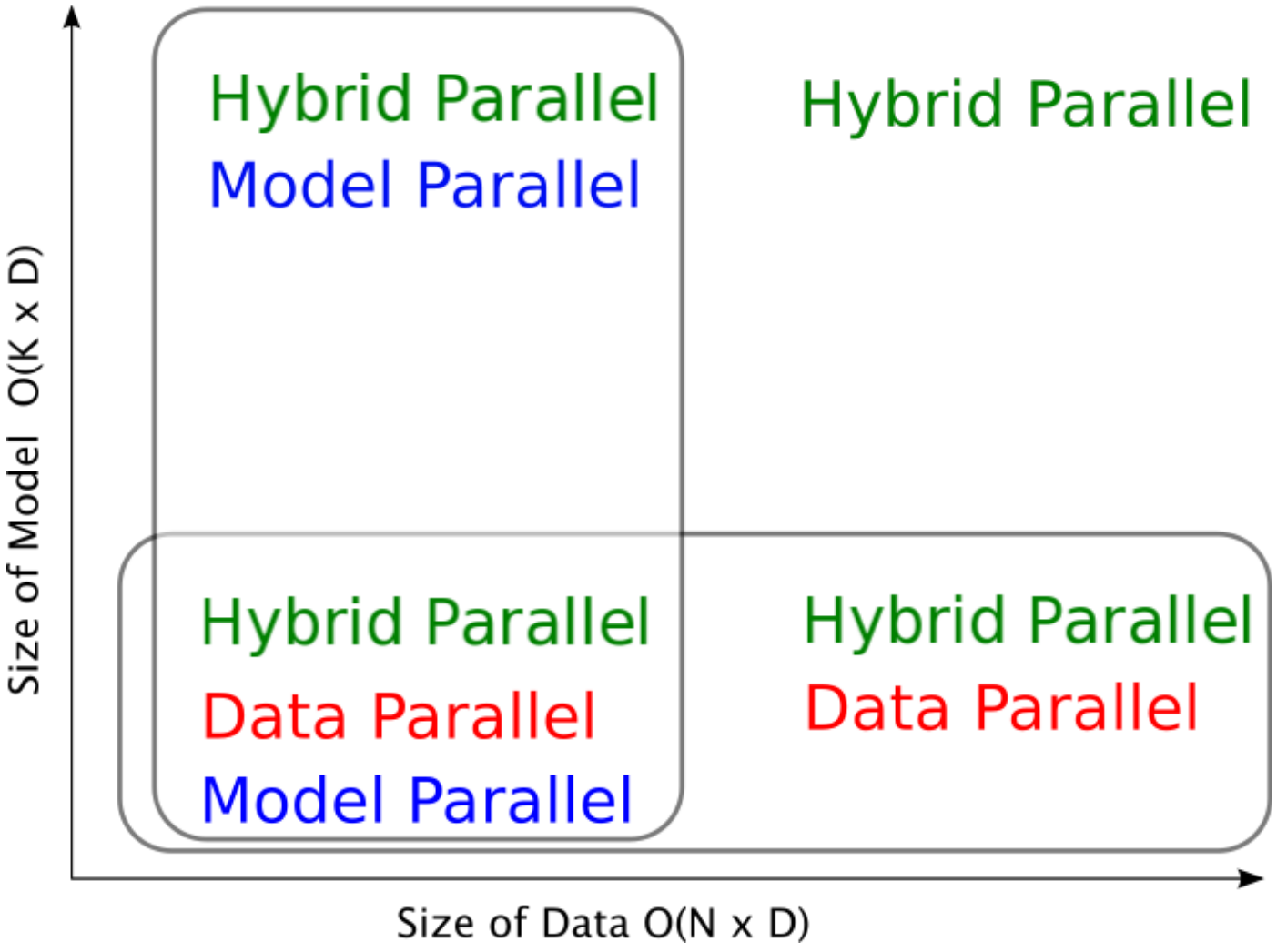
Why Hybrid Parallelism?



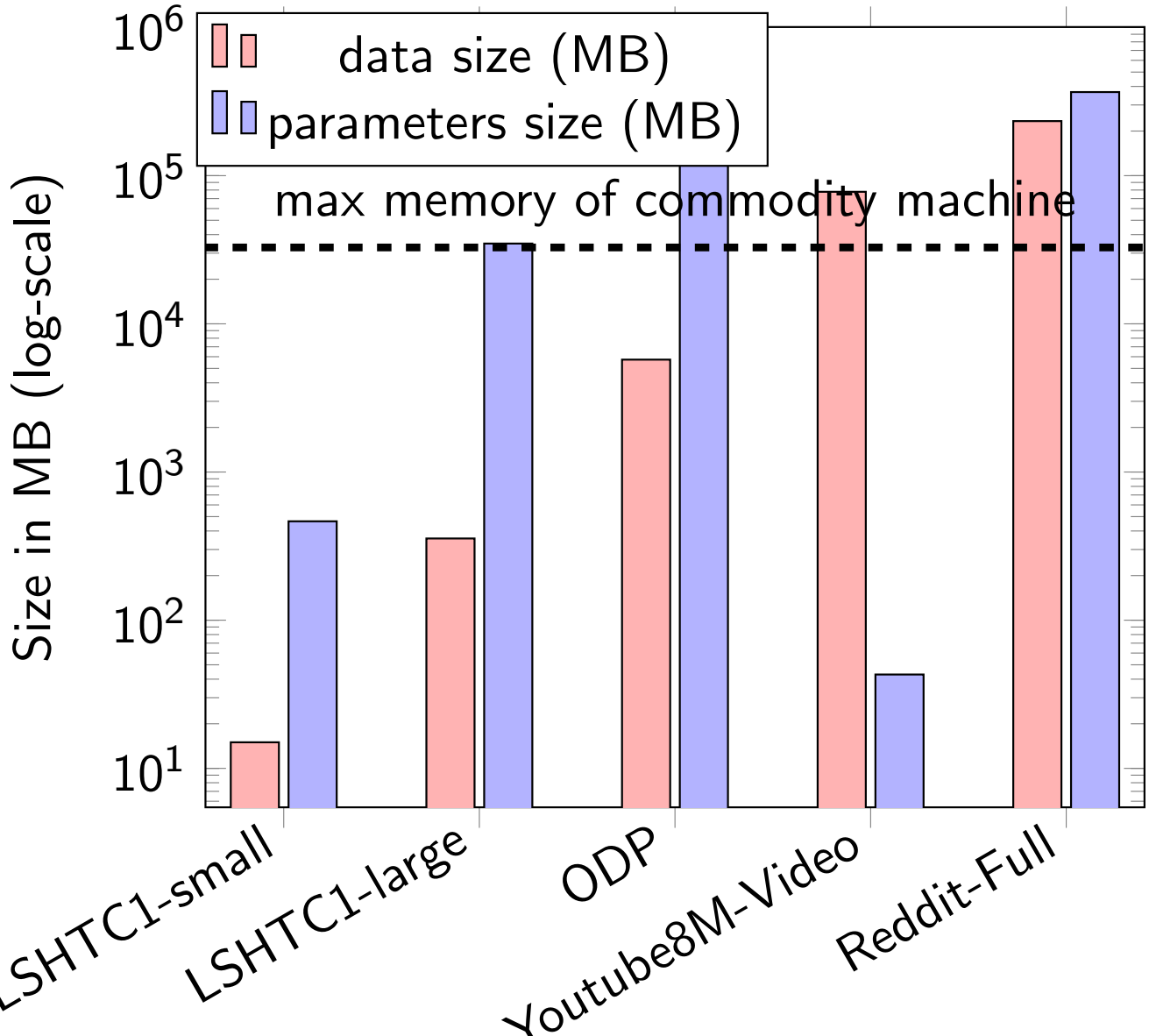
Why Hybrid Parallelism?



Why Hybrid Parallelism?



Why Hybrid Parallelism?



Benefits of Hybrid-Parallelism

1. **One versatile method** for all regimes of data and model parallelism
2. **Independent parameter updates** on each worker

Benefits of Hybrid-Parallelism

1. **One versatile method** for all regimes of data and model parallelism
2. **Independent parameter updates** on each worker
3. **Fully de-centralized** and **asynchronous** optimization algorithms

How do we achieve **Hybrid Parallelism** in machine learning models?

Double-Separability

Hybrid Parallelism

Double-Separability

Definition

Let $\{\mathbb{S}_i\}_{i=1}^{m}$ and $\{\mathbb{S}'_j\}_{j=1}^{m'}$ be two families of sets of parameters. A function $f : \prod_{i=1}^{m} \mathbb{S}_i \times \prod_{j=1}^{m'} \mathbb{S}'_j \to \mathbb{R}$ is doubly separable if $\exists\ f_{ij} : \mathbb{S}_i \times \mathbb{S}'_j \to \mathbb{R}$ for each $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, m'$ such that:

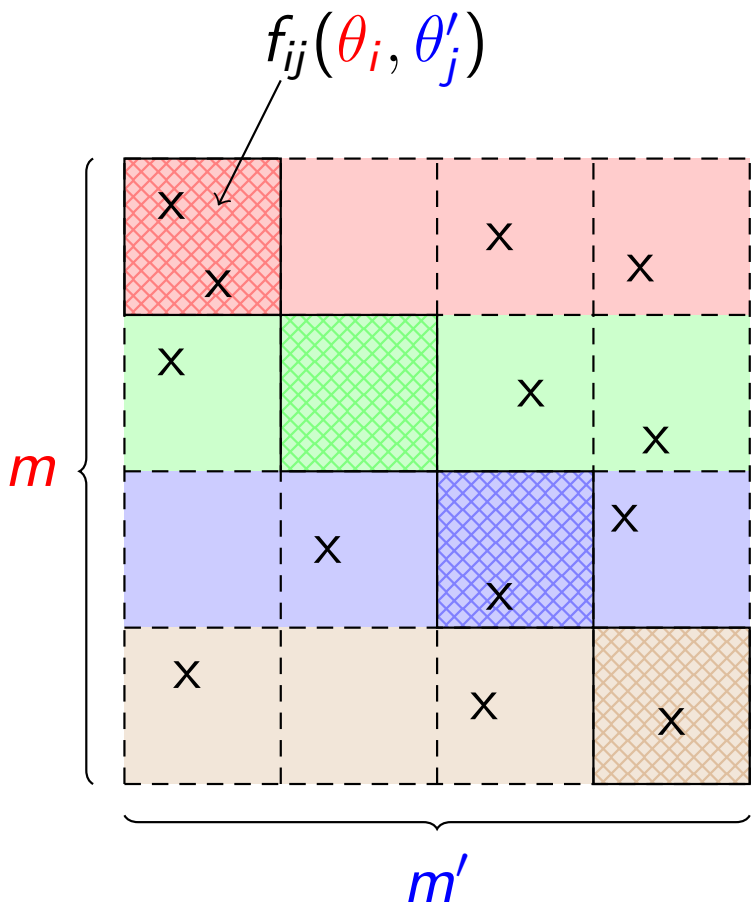
$$f(\theta_1, \theta_2, \ldots, \theta_m, \theta'_1, \theta'_2, \ldots, \theta'_{m'}) = \sum_{i=1}^{m} \sum_{j=1}^{m'} f_{ij}(\theta_i, \theta'_j)$$

Double-Separability

$$f(\theta_1, \theta_2, \ldots, \theta_m, \theta'_1, \theta'_2, \ldots, \theta'_{m'}) = \sum_{i=1}^{m} \sum_{j=1}^{m'} f_{ij}(\theta_i, \theta'_j)$$

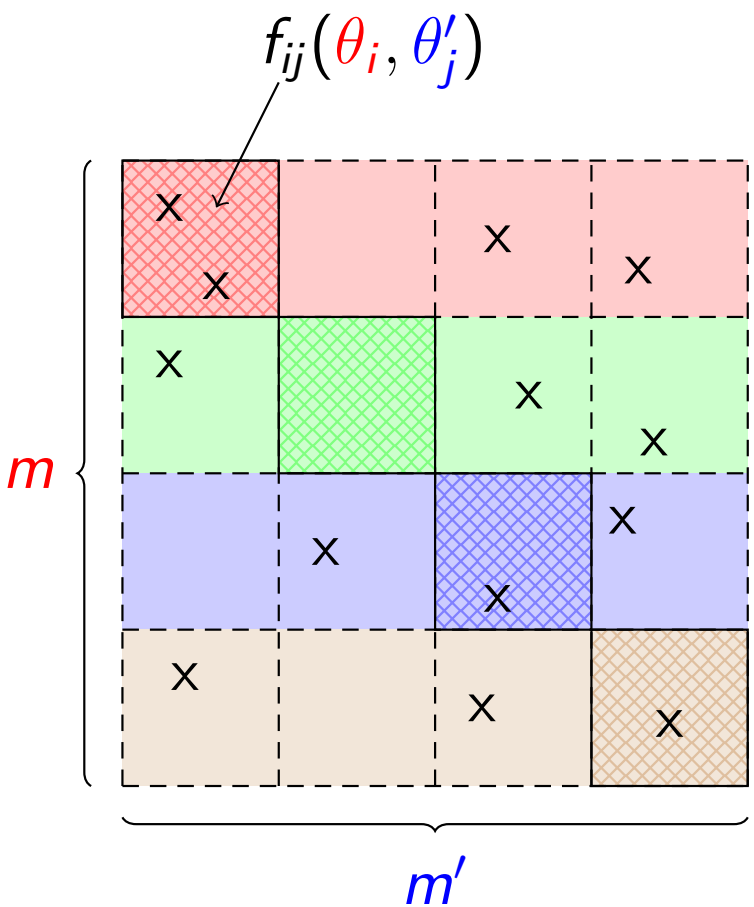
Double-Separability

$$f(\theta_1, \theta_2, \ldots, \theta_m, \theta'_1, \theta'_2, \ldots, \theta'_{m'}) = \sum_{i=1}^{m} \sum_{j=1}^{m'} f_{ij}(\theta_i, \theta'_j)$$



Double-Separability

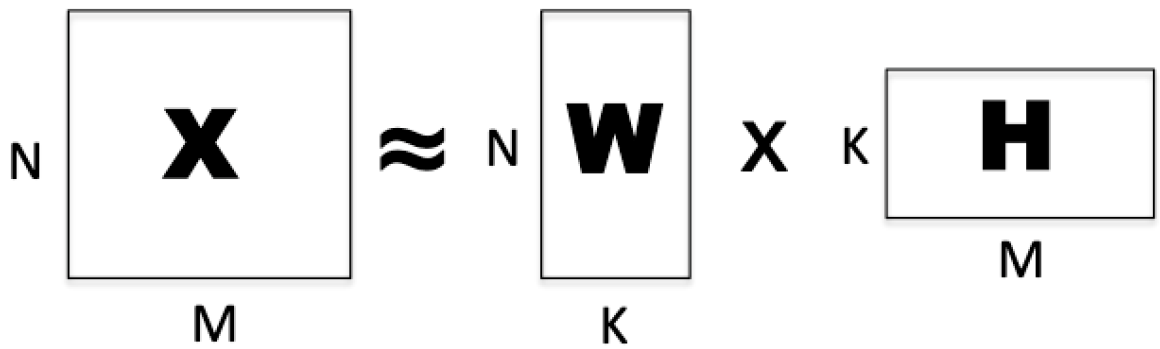
$$f(\theta_1, \theta_2, \ldots, \theta_m, \theta'_1, \theta'_2, \ldots, \theta'_{m'}) = \sum_{i=1}^{m} \sum_{j=1}^{m'} f_{ij}(\theta_i, \theta'_j)$$



f_{ij} corresponding to highlighted diagonal blocks can be computed **independently** and in **parallel**

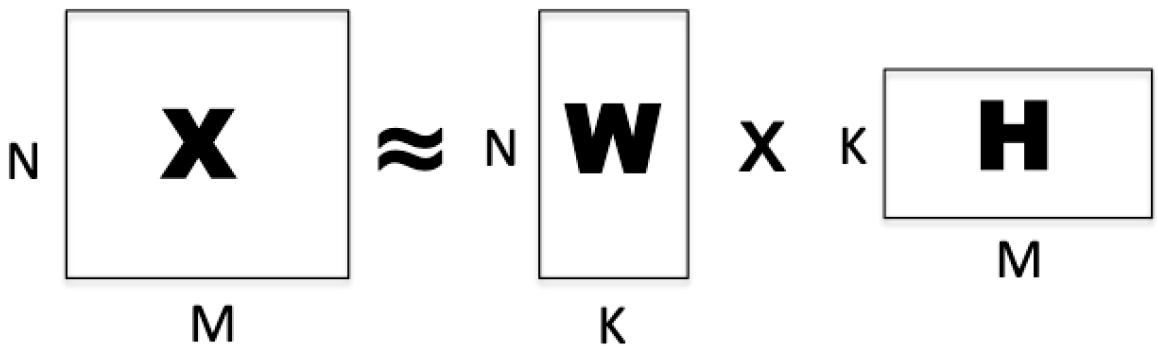
Direct Double-Separability

e.g. Matrix Factorization



Direct Double-Separability

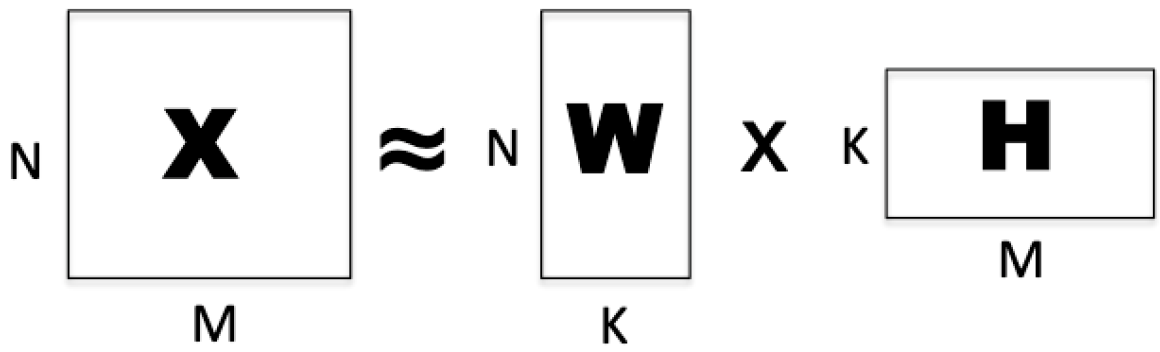
e.g. Matrix Factorization



$$\mathcal{L}(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_N, \mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_M) = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{M} (X_{ij} - \langle \mathbf{w}_i, \mathbf{h}_j \rangle)^2$$

Direct Double-Separability

e.g. Matrix Factorization



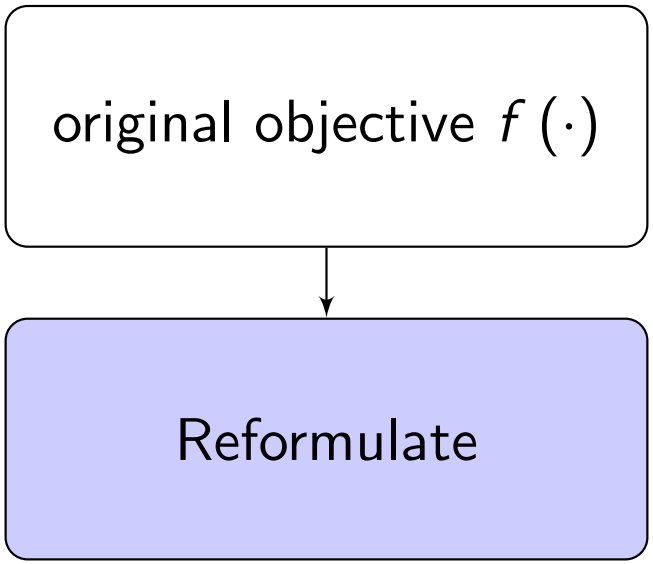
$$\mathcal{L}(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_N, \mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_M) = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{M} (X_{ij} - \langle \mathbf{w}_i, \mathbf{h}_j \rangle)^2$$

Objective function is trivially doubly-separable! [Yun et al 2014]

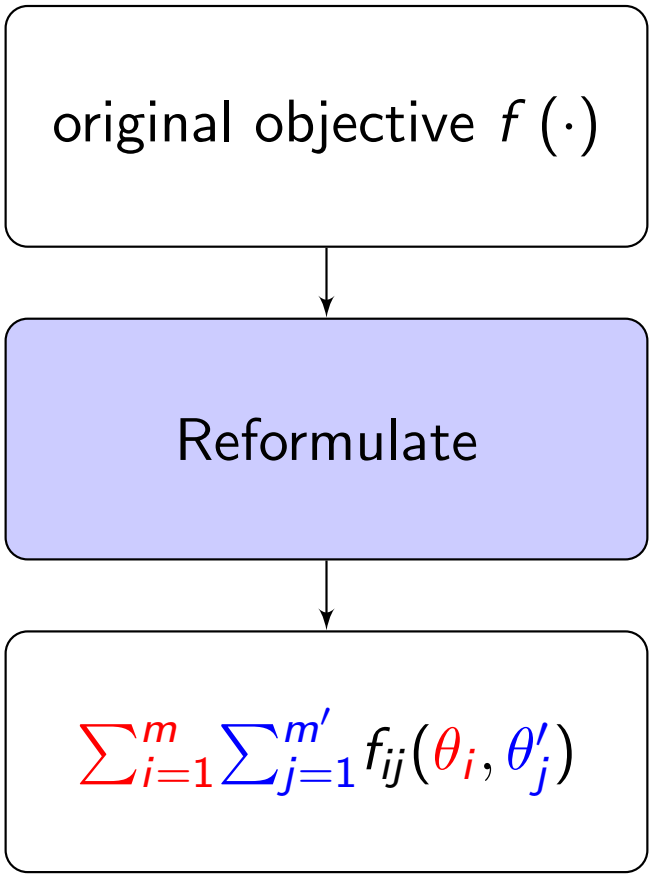
Others require Reformulations

original objective $f(\cdot)$

Others require Reformulations



Others require Reformulations



Technical Contributions of my thesis

Study frequentist and bayesian models which **do not have direct double-separability** in their parameter estimation,

Technical Contributions of my thesis

Study frequentist and bayesian models which **do not have direct double-separability** in their parameter estimation,

1. Propose reformulations that lead to Hybrid-Parallelism

Technical Contributions of my thesis

Study frequentist and bayesian models which **do not have direct double-separability** in their parameter estimation,

1. Propose reformulations that lead to Hybrid-Parallelism
2. Design asynchronous distributed optimization algorithms

Outline

1 Introduction

2 Distributed Parameter Estimation

3 Thesis Roadmap
- Multinomial Logistic Regression (KDD 2019)
- Mixture Models (AISTATS 2019)
- Latent Collaborative Retrieval (NIPS 2014)

4 Conclusion and Future Directions

5 Appendix

Outline

1. Introduction
2. Distributed Parameter Estimation
3. Thesis Roadmap
   - Multinomial Logistic Regression (KDD 2019)
   - Mixture Models (AISTATS 2019)
   - Latent Collaborative Retrieval (NIPS 2014)
4. Conclusion and Future Directions
5. Appendix

Multinomial Logistic Regression (MLR)

Given

- Training data $(\mathbf{x}_i, y_i)_{i=1,\ldots,N}$ where $\mathbf{x}_i \in \mathbb{R}^D$
- corresp. labels $y_i \in \{1, 2, \ldots, K\}$
- N, D and K are large ($N >>> D >> K$)

Multinomial Logistic Regression (MLR)

Given

- Training data $(\mathbf{x}_i, y_i)_{i=1,\ldots,N}$ where $\mathbf{x}_i \in \mathbb{R}^D$
- corresp. labels $y_i \in \{1, 2, \ldots, K\}$
- N, D and K are large ($N >>> D >> K$)

Goal

The probability that $\mathbf{x}_i$ belongs to class k is given by:

$$p(y_i = k|\mathbf{x}_i, W) = \frac{\exp(\mathbf{w}_k{}^T \mathbf{x}_i)}{\sum_{j=1}^{K} \exp(\mathbf{w}_j{}^T \mathbf{x}_i)}$$

where $W = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K\}$ denotes the parameter for the model.

Multinomial Logistic Regression (MLR)

The corresponding l_2 **regularized negative log-likelihood loss**:

$$\min_{W} \frac{\lambda}{2} \sum_{k=1}^{K} \|\mathbf{w}_k\|^2 - \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \mathbf{w}_k{}^T \mathbf{x}_i + \frac{1}{N} \sum_{i=1}^{N} \log \left(\sum_{k=1}^{K} \exp(\mathbf{w}_k{}^T \mathbf{x}_i) \right)$$

where λ is the regularization hyper-parameter.

Multinomial Logistic Regression (MLR)

The corresponding l_2 **regularized negative log-likelihood loss**:

$$\min_{W} \frac{\lambda}{2} \sum_{k=1}^{K} \|\mathbf{w}_k\|^2 - \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \mathbf{w}_k{}^T \mathbf{x}_i + \frac{1}{N} \sum_{i=1}^{N} \underbrace{\log \left(\sum_{k=1}^{K} \exp(\mathbf{w}_k{}^T \mathbf{x}_i) \right)}_{\text{makes model parallelism hard}}$$

where λ is the regularization hyper-parameter.

Reformulation into Doubly-Separable form

Log-concavity bound [GopYan13]

$$\log(\gamma) \leq a \cdot \gamma - \log(a) - 1, \quad \forall \gamma, a > 0,$$

where a is a variational parameter. This bound is tight when $a = \frac{1}{\gamma}$.

Reformulating the objective of MLR

$$\min_{W,A} \frac{\lambda}{2} \sum_{k=1}^{K} \|\mathbf{w}_k\|^2 + \frac{1}{N} \sum_{i=1}^{N} \left(- \sum_{k=1}^{K} y_{ik} \mathbf{w}_k{}^T \mathbf{x}_i + a_i \sum_{k=1}^{K} \exp(\mathbf{w}_k{}^T \mathbf{x}_i) - \log(a_i) - 1 \right)$$

where a_i can be computed in closed form as:

$$a_i = \frac{1}{\sum_{k=1}^{K} \exp(\mathbf{w}_k{}^T \mathbf{x}_i)}$$

Doubly-Separable Multinomial Logistic Regression (DS-MLR)

Doubly-Separable form

$$\min_{W,A} \sum_{i=1}^{N}\sum_{k=1}^{K}\left(\frac{\lambda\|\mathbf{w}_k\|^2}{2N} - \frac{y_{ik}\mathbf{w}_k{}^T\mathbf{x}_i}{N} - \frac{\log a_i}{NK} + \frac{\exp(\mathbf{w}_k{}^T\mathbf{x}_i + \log a_i)}{N} - \frac{1}{NK}\right)$$

Doubly-Separable Multinomial Logistic Regression (DS-MLR)

Stochastic Gradient Updates

Each term in stochastic update depends on only data point i and class k.

Doubly-Separable Multinomial Logistic Regression (DS-MLR)

Stochastic Gradient Updates

Each term in stochastic update depends on only data point i and class k.

- $\mathbf{w}_k{}^{t+1} \leftarrow \mathbf{w}_k{}^{t} - \eta_t K \left(\lambda \mathbf{x}_i - y_{ik}\mathbf{x}_i + \exp\left(\mathbf{w}_k{}^{T}\mathbf{x}_i + \log a_i\right)\mathbf{x}_i\right)$

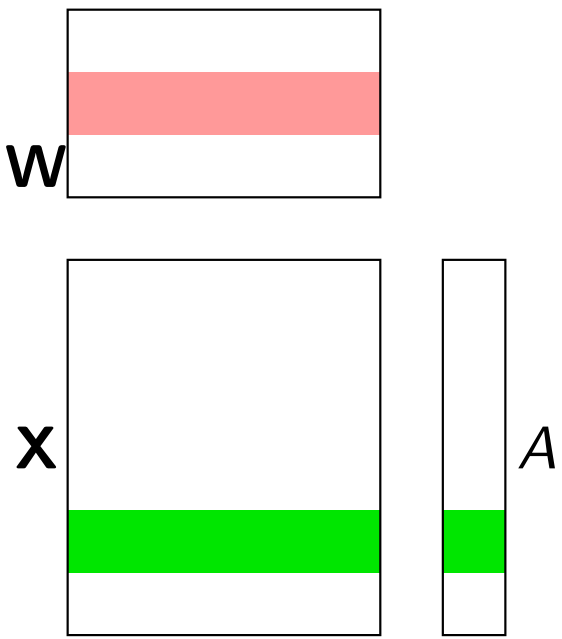
Doubly-Separable Multinomial Logistic Regression (DS-MLR)

Stochastic Gradient Updates

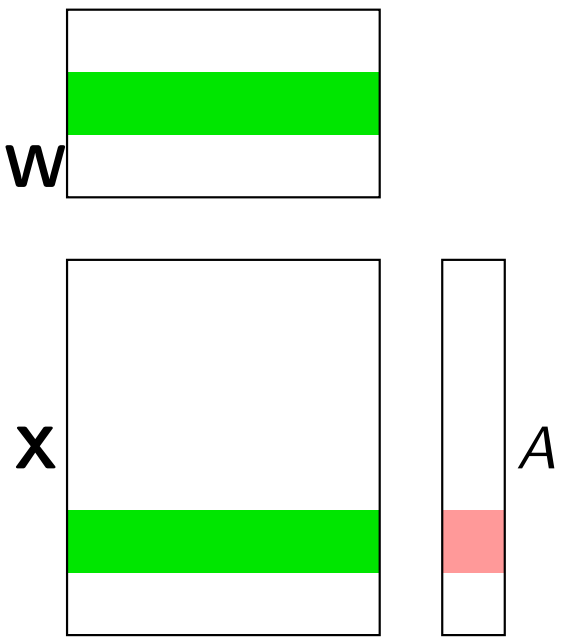
Each term in stochastic update depends on only data point i and class k.

- $\mathbf{w}_k{}^{t+1} \leftarrow \mathbf{w}_k{}^{t} - \eta_t K \left(\lambda \mathbf{x}_i - y_{ik} \mathbf{x}_i + \exp\left(\mathbf{w}_k{}^{T} \mathbf{x}_i + \log a_i \right) \mathbf{x}_i \right)$
- $\log a_i{}^{t+1} \leftarrow \log a_i{}^{t} - \eta_t K \left(\exp(\mathbf{w}_k{}^{T} \mathbf{x}_i + \log a_i) - \frac{1}{K} \right)$

Access Pattern of updates: Stoch $\mathbf{w}_k$, Stoch a_i



(a) Updating $\mathbf{w}_k$ only requires computing a_i



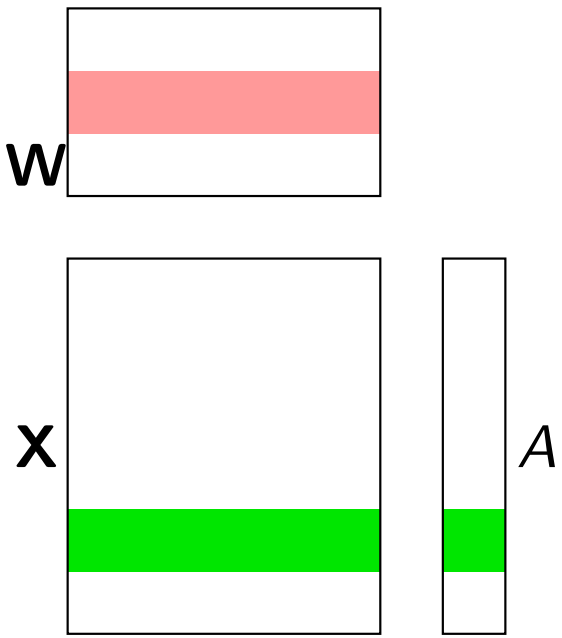
(b) Updating a_i only requires accessing $\mathbf{w}_k$ and $\mathbf{x}_i$.

Updating a_i: Closed form instead of Stoch update

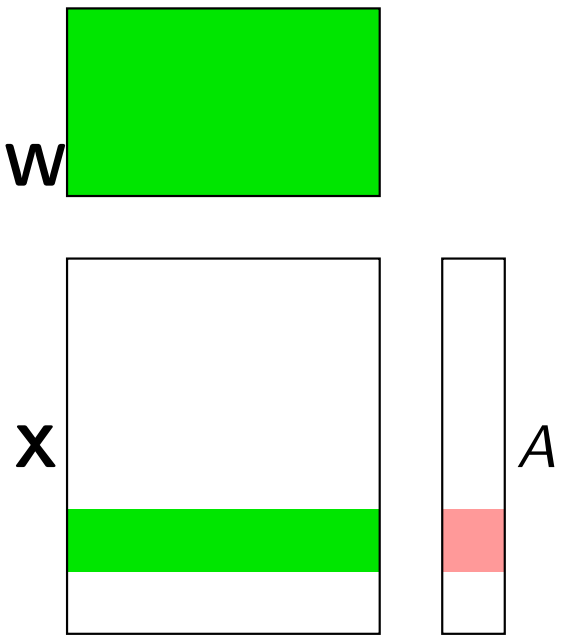
Closed-form update for a_i

$$a_i = \frac{1}{\sum_{k=1}^{K} \exp(\mathbf{w}_k{}^T \mathbf{x}_i)}$$

Access Pattern of updates: Stoch $\mathbf{w}_k$, Exact a_i



(a) Updating $\mathbf{w}_k$ only requires computing a_i



(b) Updating a_i requires accessing entire $\mathbf{W}$. Synchronization bottleneck!

Updating a_i: Avoiding bulk-synchronization

Closed-form update for a_i

$$a_i = \frac{1}{\sum_{k=1}^{K} \exp(\mathbf{w}_k{}^T \mathbf{x}_i)}$$

Updating a_i: Avoiding bulk-synchronization

Closed-form update for a_i

$$a_i = \frac{1}{\sum_{k=1}^{K} \exp(\mathbf{w}_k{}^T \mathbf{x}_i)}$$

- Each worker computes partial sum using the $\mathbf{w}_k$ it owns.

Updating a_i: Avoiding bulk-synchronization

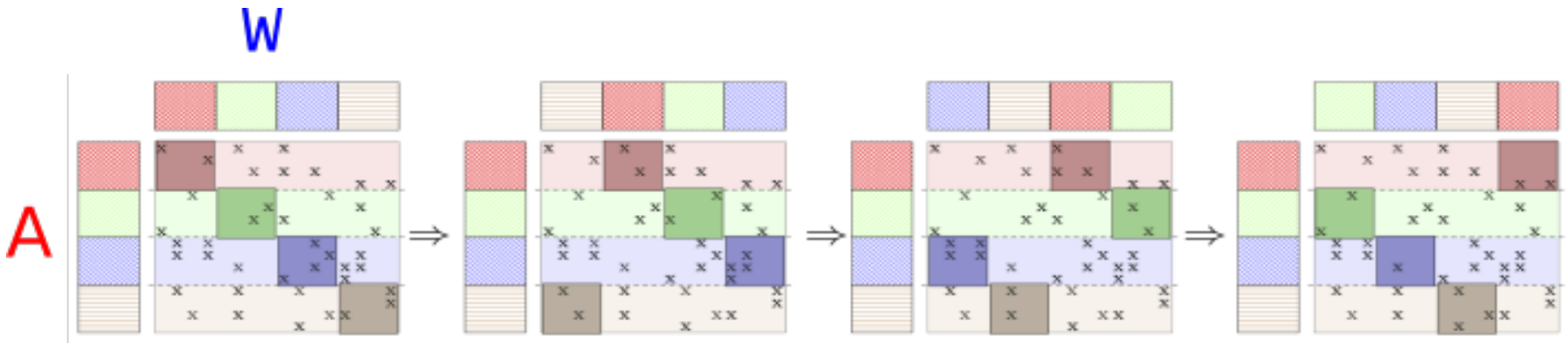
Closed-form update for a_i

$$a_i = \frac{1}{\sum_{k=1}^{K} \exp(\mathbf{w}_k{}^T \mathbf{x}_i)}$$

- Each worker computes partial sum using the $\mathbf{w}_k$ it owns.
- P workers: After P rounds the global sum is available

Parallelization: Synchronous

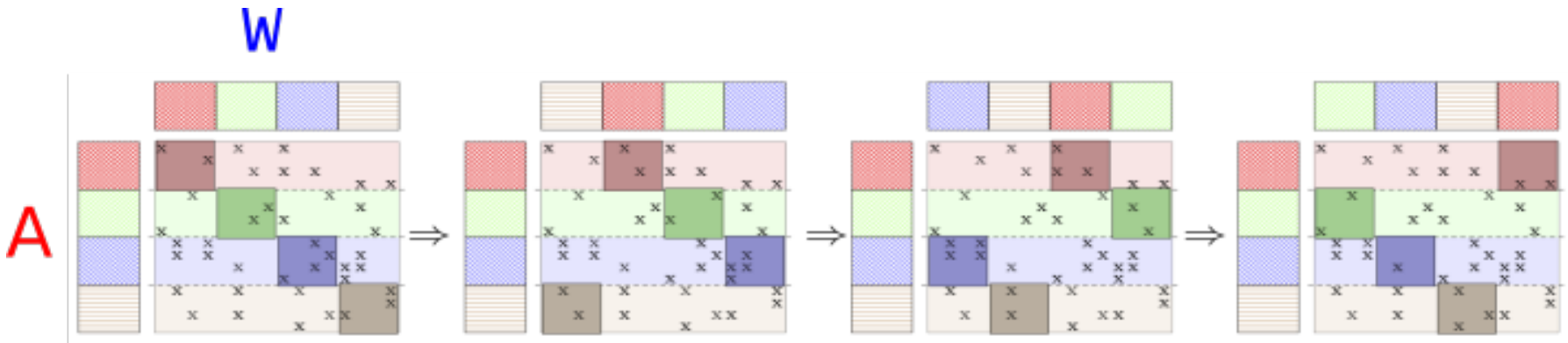
DSGD [Gemulla et al., 2011]



- **X** and local parameters **A** are partitioned horizontally $(1, \ldots, N)$

Parallelization: Synchronous

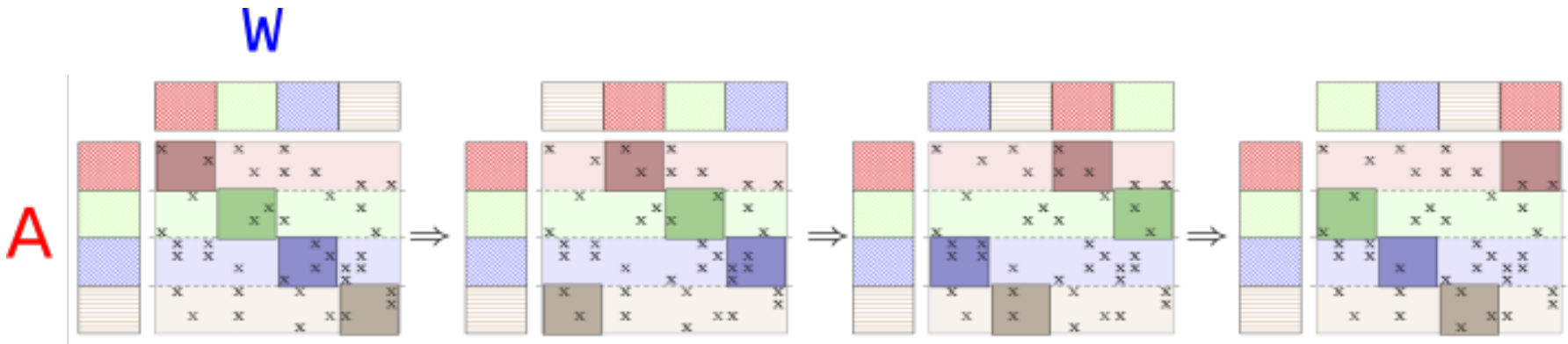
DSGD [Gemulla et al., 2011]



- **X** and local parameters **A** are partitioned horizontally $(1, \ldots, N)$
- Global model parameters **W** are partitioned vertically $(1, \ldots, K)$

Parallelization: Synchronous

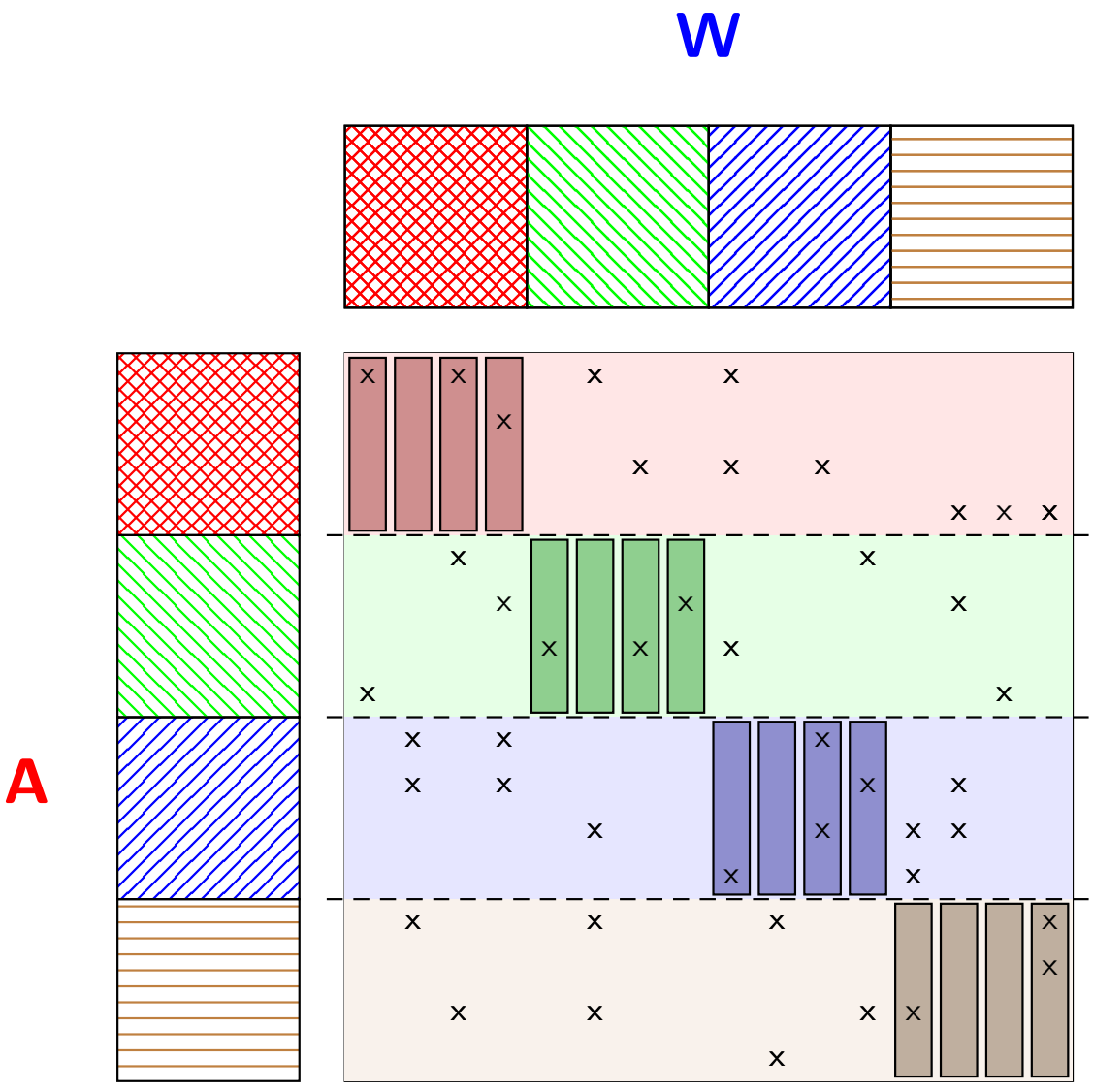
DSGD [Gemulla et al., 2011]



- **X** and local parameters **A** are partitioned horizontally $(1, \ldots, N)$
- Global model parameters **W** are partitioned vertically $(1, \ldots, K)$
- $P = 4$ workers work on mutually-exclusive blocks of **A** and **W**

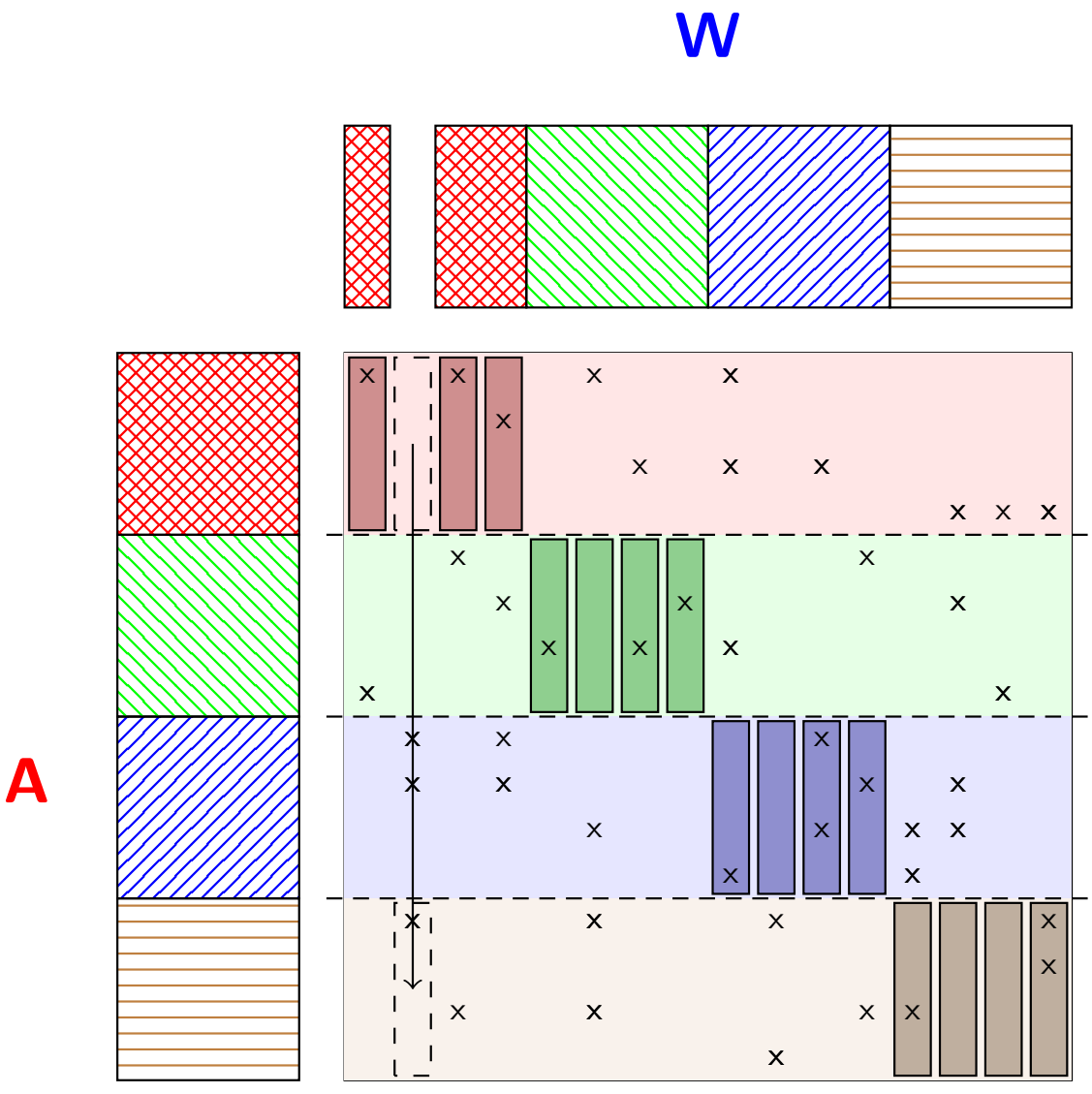
Parallelization: Asynchronous

NOMAD [Yun et al., 2014]



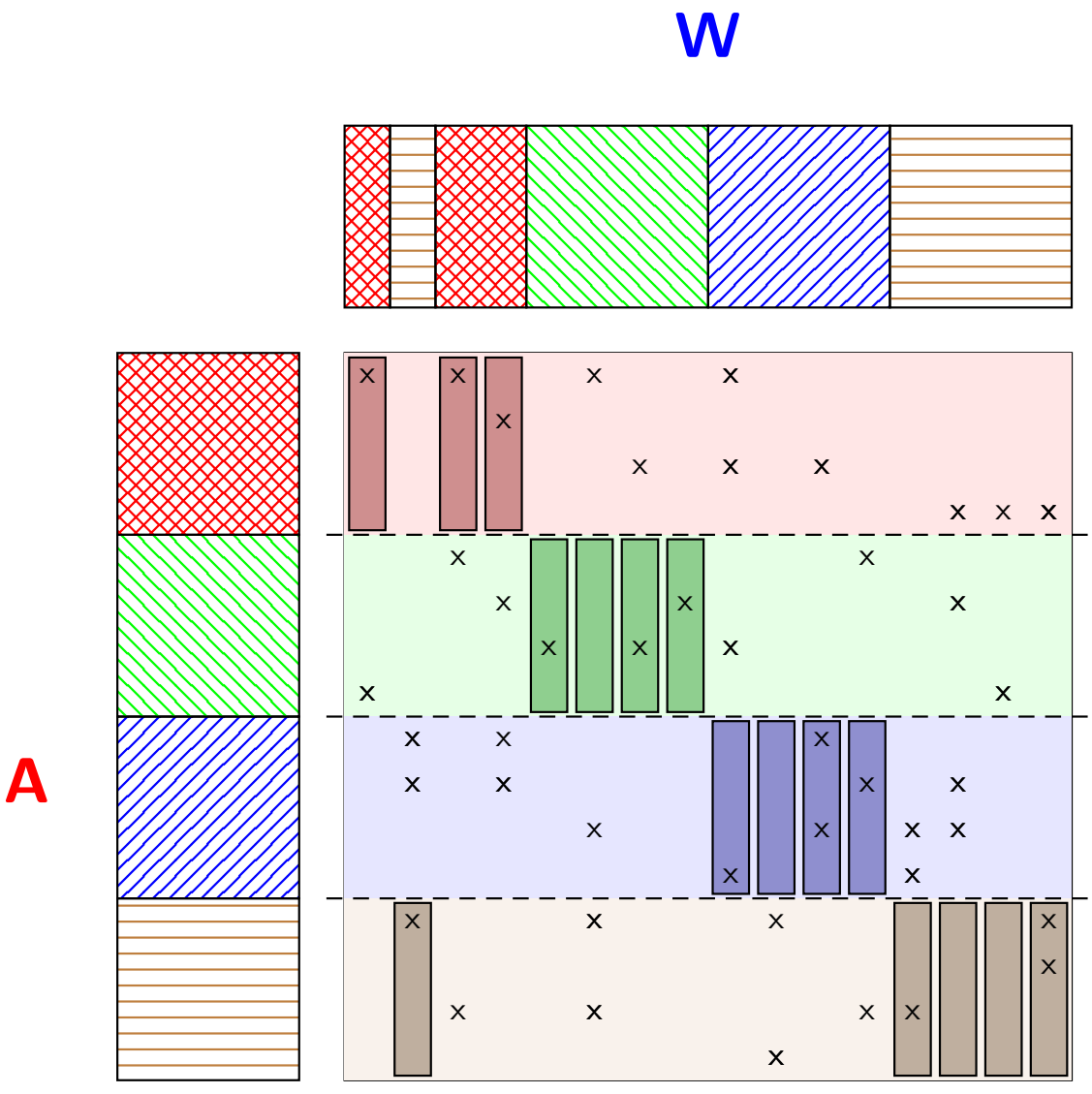
Parallelization: Asynchronous

NOMAD [Yun et al 2014]



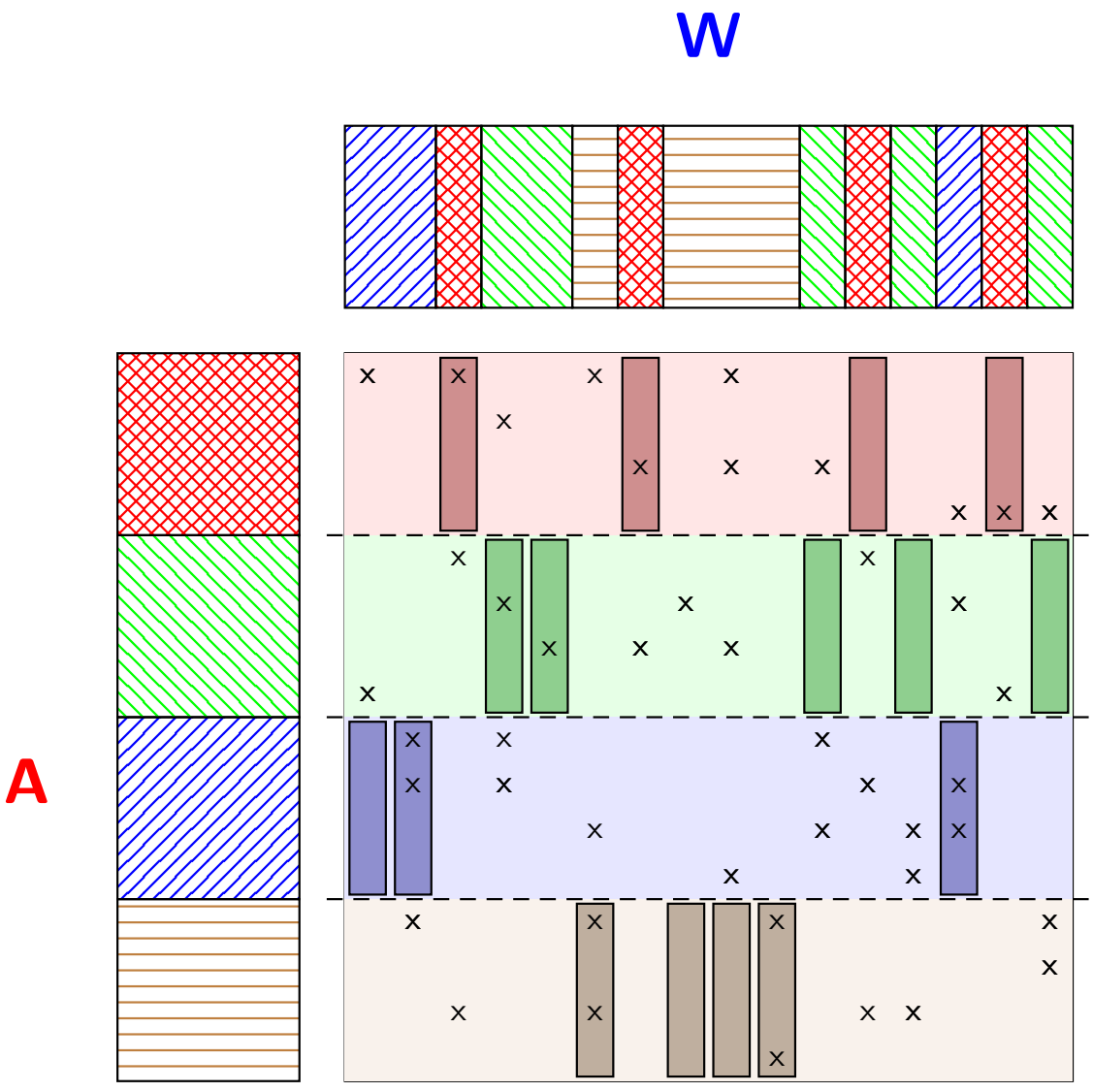
Parallelization: Asynchronous

NOMAD [Yun et al 2014]

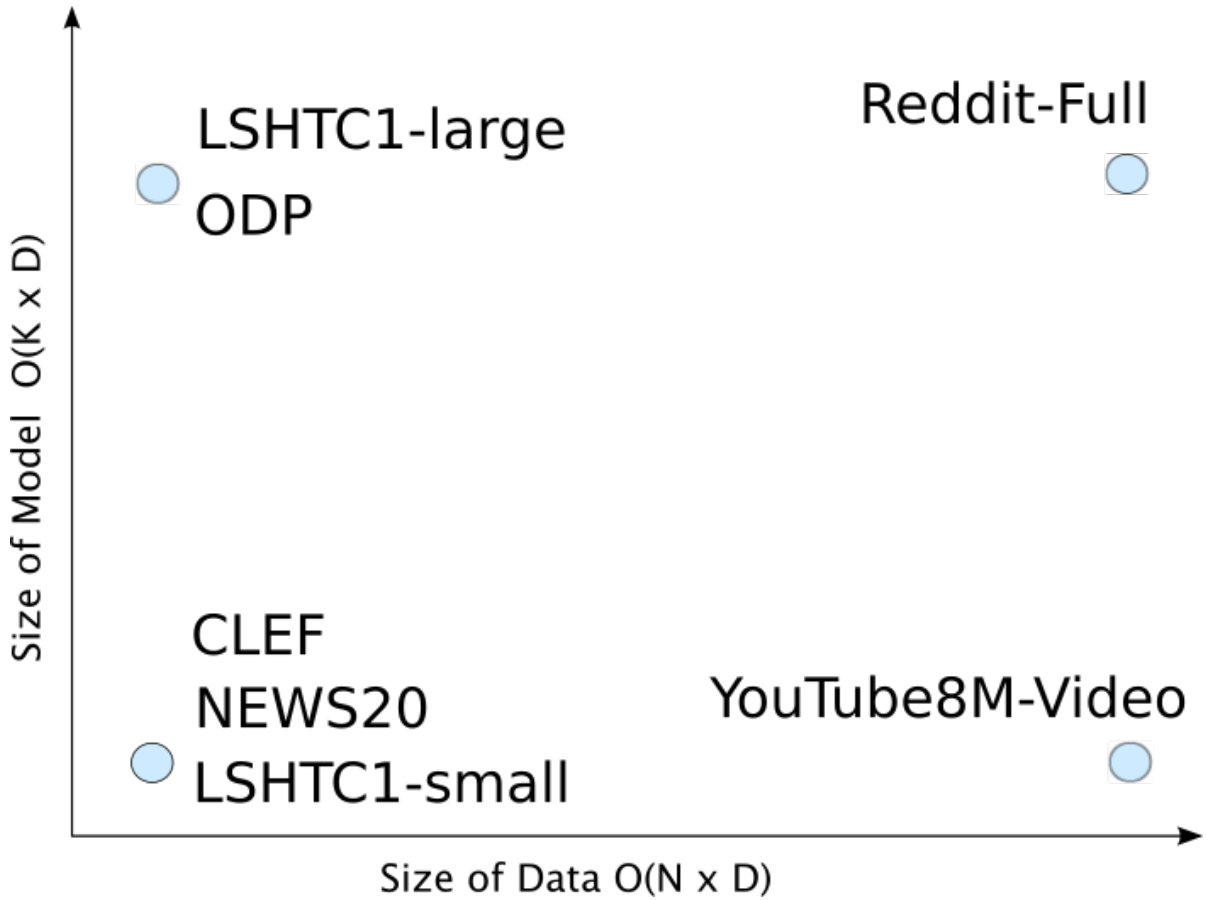


Parallelization: Asynchronous

NOMAD [Yun et al 2014]



Experiments: Datasets

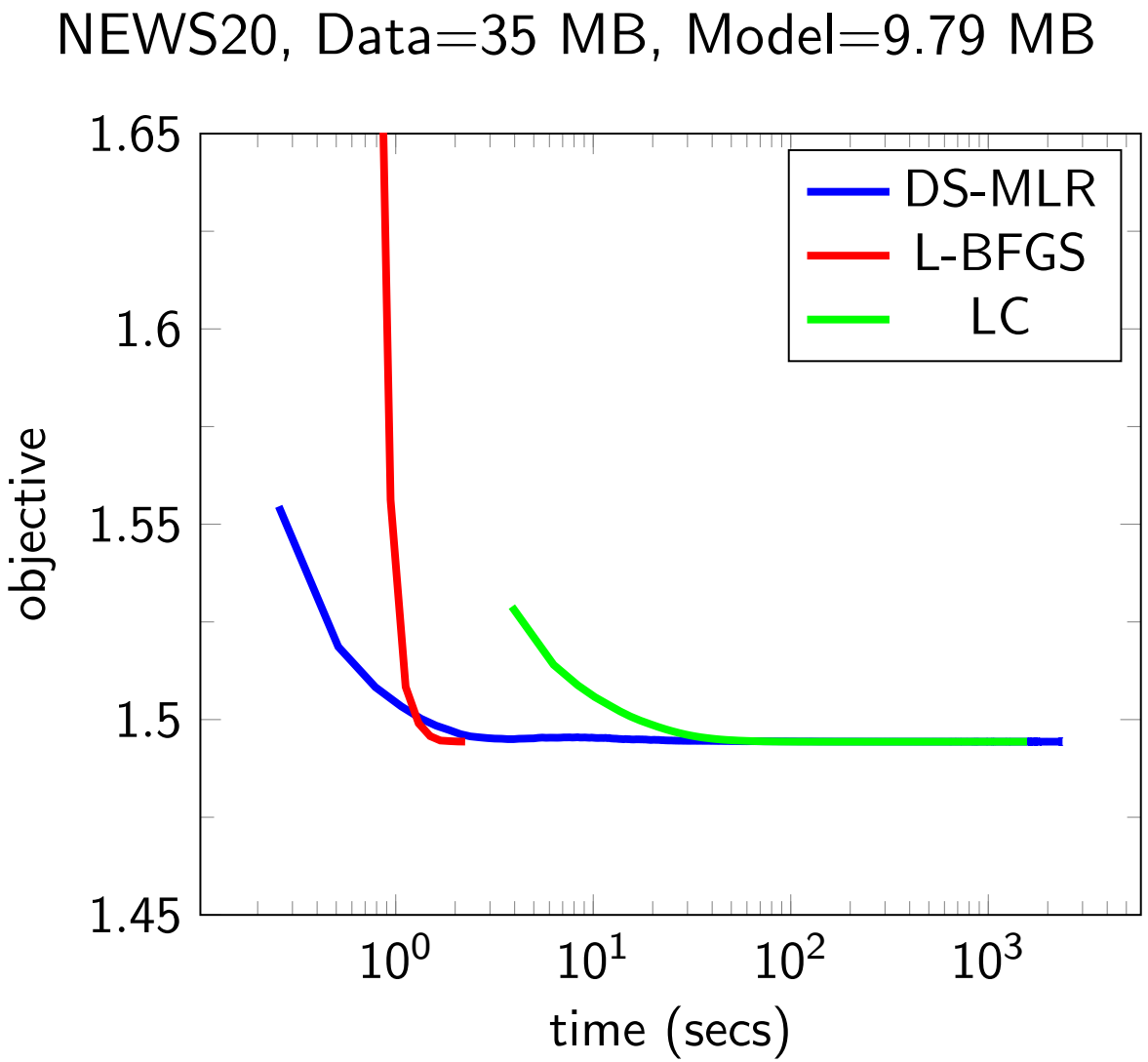


Experiments: Datasets

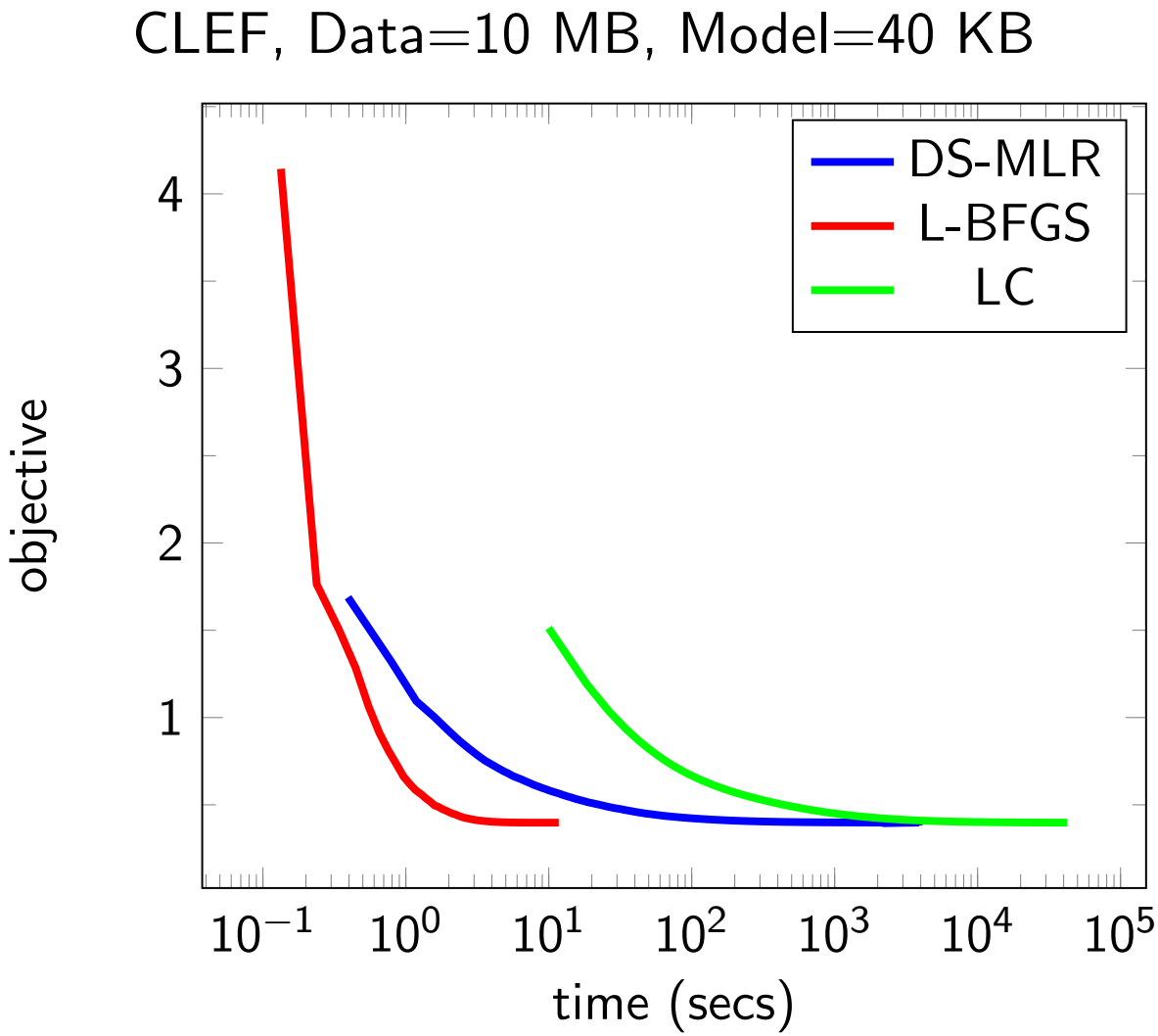
Dataset	# instances	# features	#classes	data (train + test)	parameters	sparsity (% nnz)
CLEF	10,000	80	63	9.6 MB + 988 KB	40 KB	100
NEWS20	11,260	53,975	20	21 MB + 14 MB	9.79 MB	0.21
LSHTC1-small	4,463	51,033	1,139	11 MB + 4 MB	465 MB	0.29
LSHTC1-large	93,805	347,256	12,294	258 MB + 98 MB	34 GB	0.049
ODP	1,084,404	422,712	105,034	3.8 GB + 1.8 GB	355 GB	0.0533
YouTube8M-Video	4,902,565	1,152	4,716	59 GB + 17 GB	43 MB	100
Reddit-Full	211,532,359	1,348,182	33,225	159 GB + 69 GB	358 GB	0.0036

Table: **Characteristics of the datasets used**

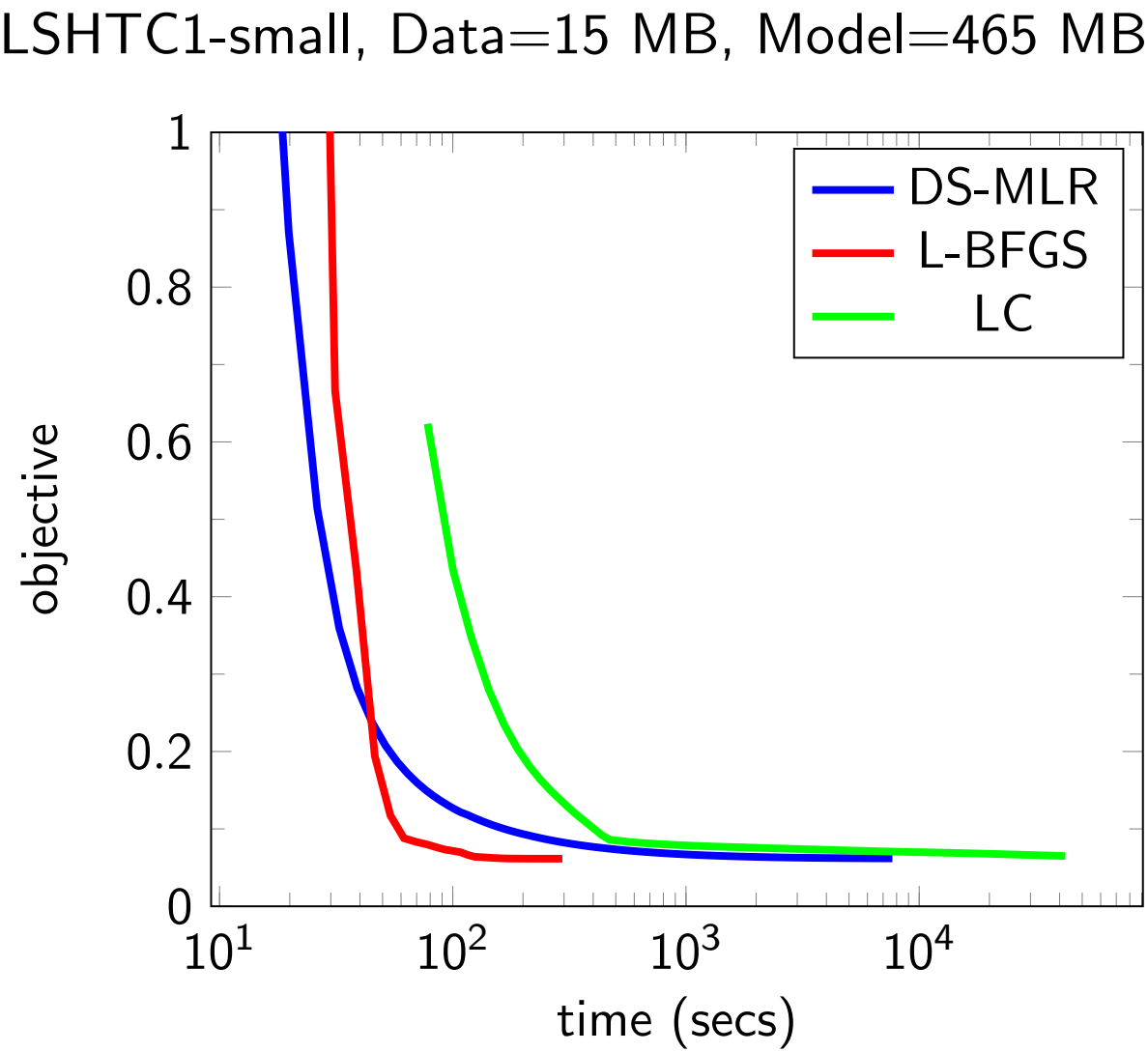
Experiments: Single Machine



Experiments: Single Machine

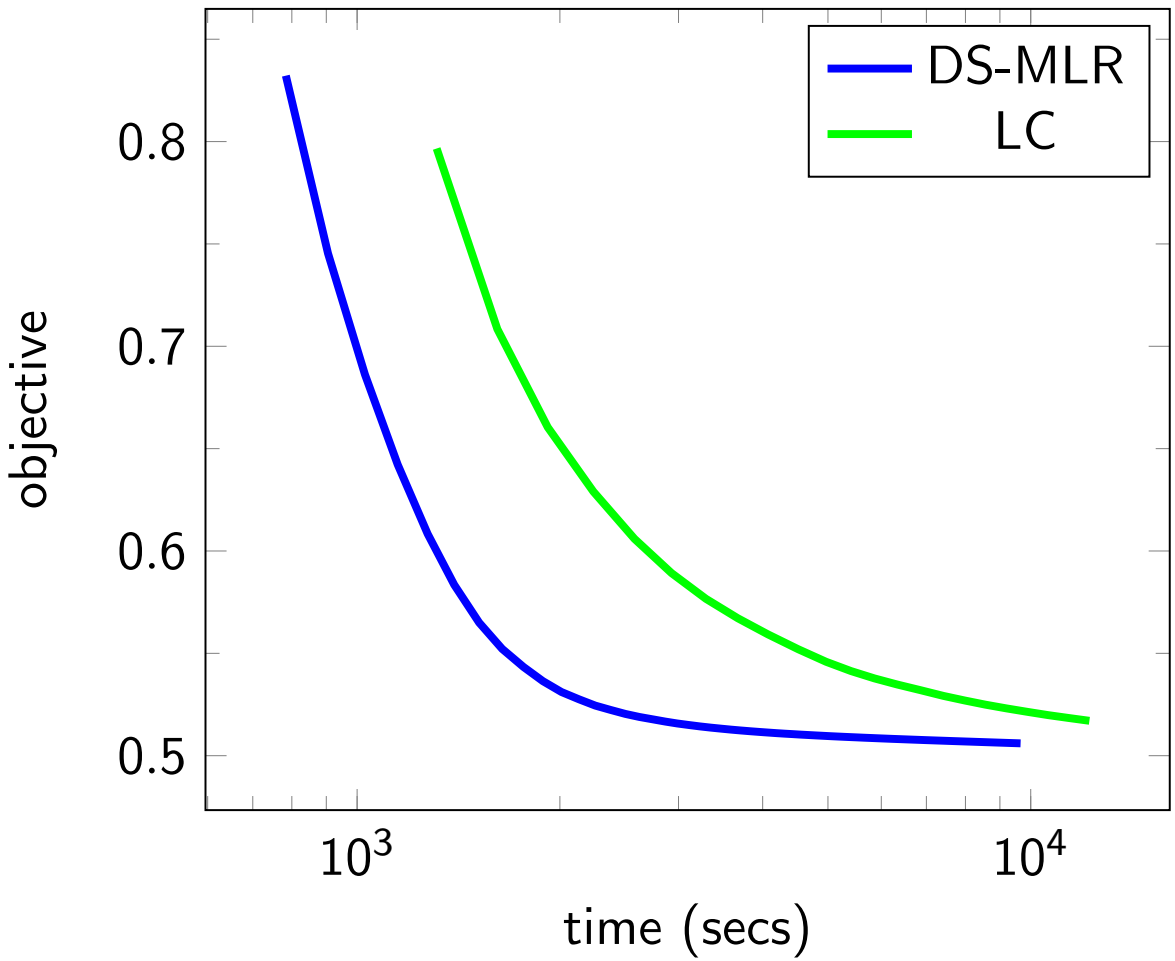


Experiments: Single Machine

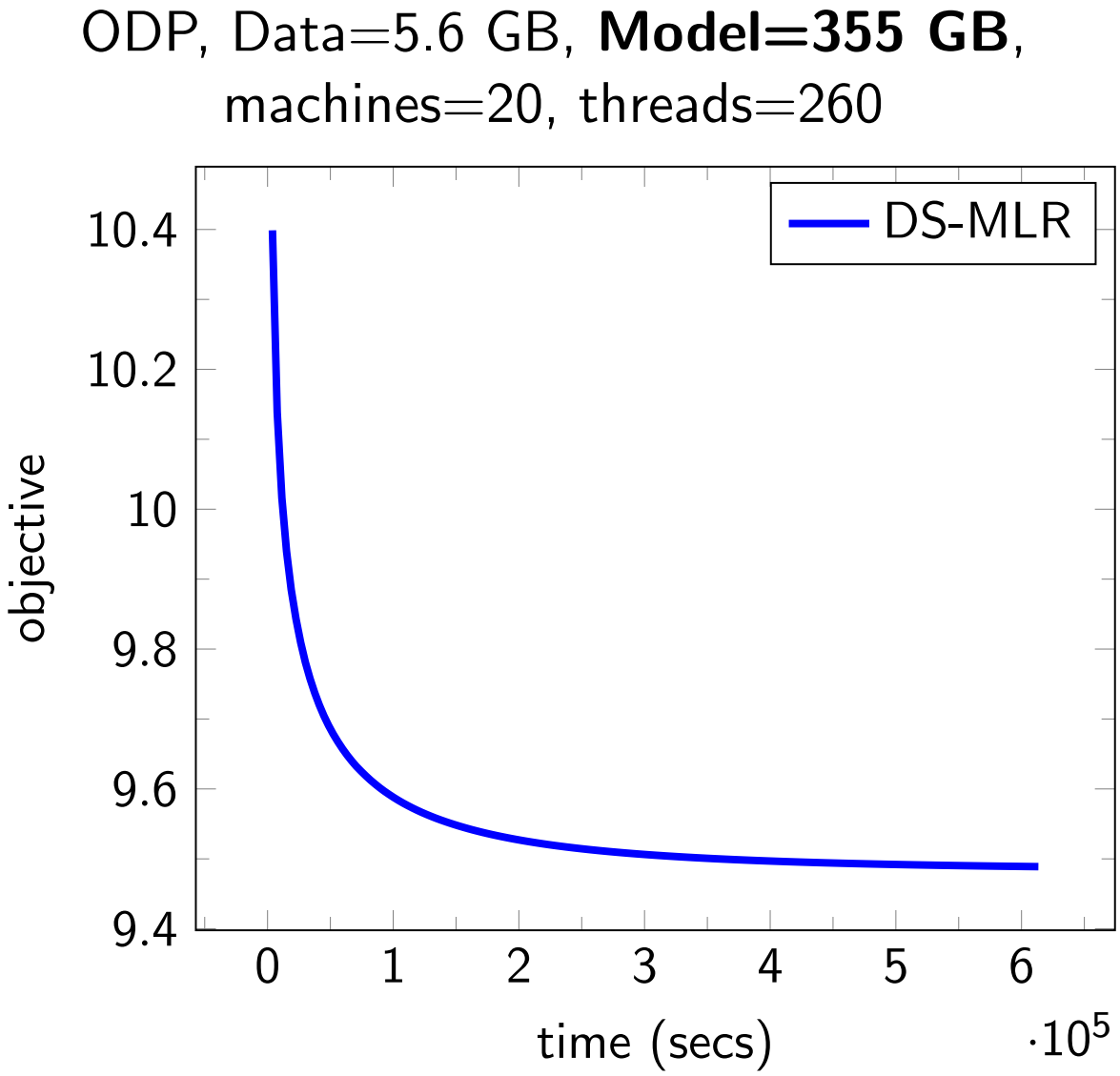


Experiments: Multi Machine

LSHTC1-large, Data=356 MB, **Model=34 GB**,
machines=4, threads=12

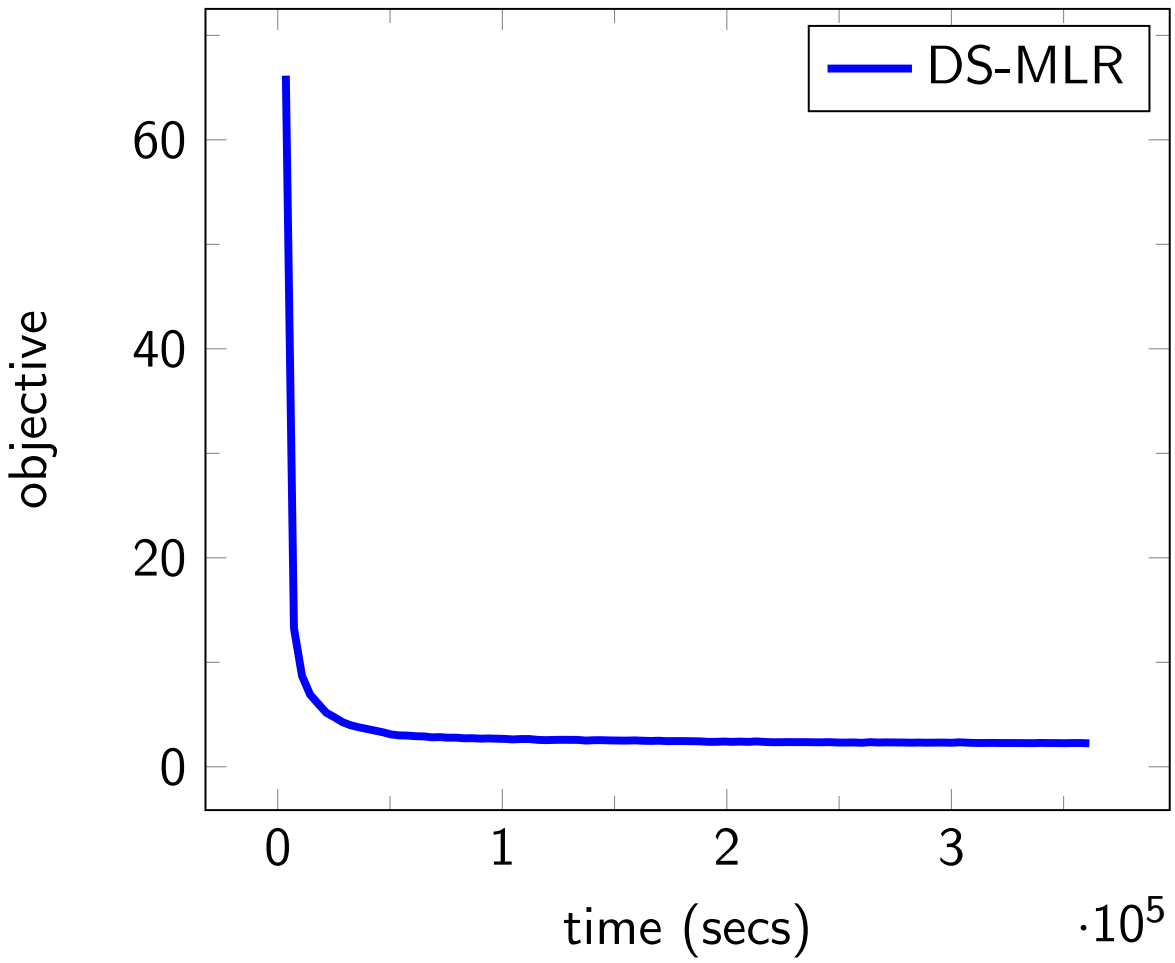


Experiments: Multi Machine



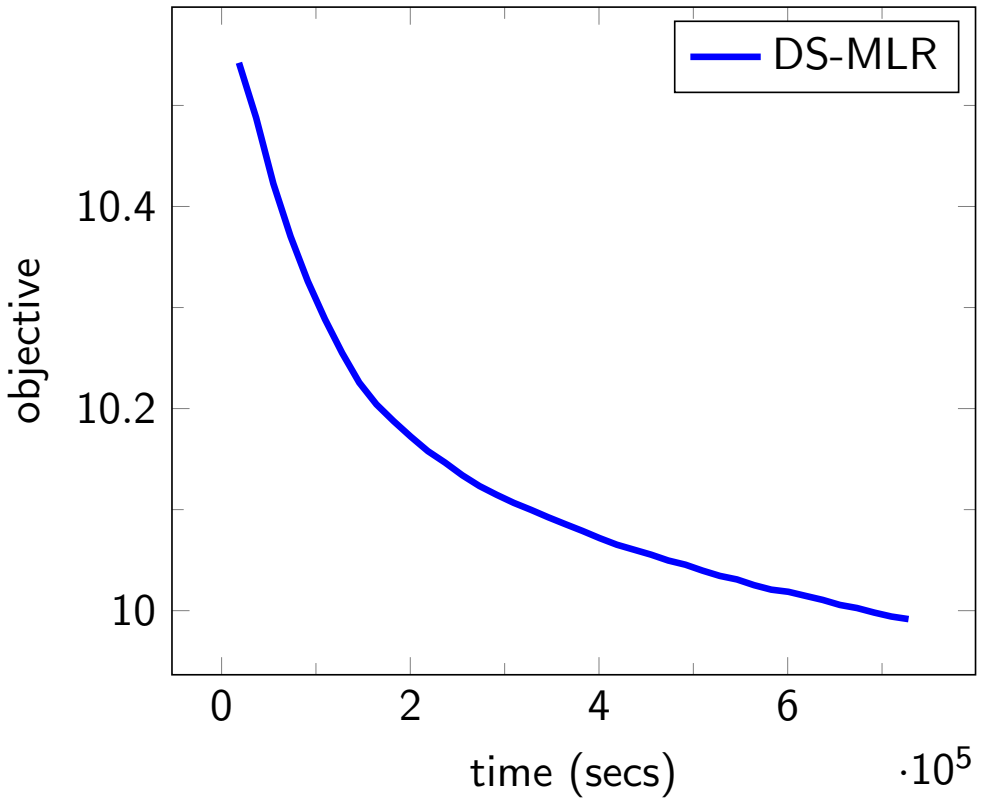
Experiments: Multi Machine Dense Dataset

YouTube-Video, **Data=76 GB**, Model=43 MB,
machines=4, threads=260



Experiments: Multi Machine (Nothing fits in memory!)

Reddit-Full, **Data=228 GB**, **Model=358 GB**,
machines=40, threads=250

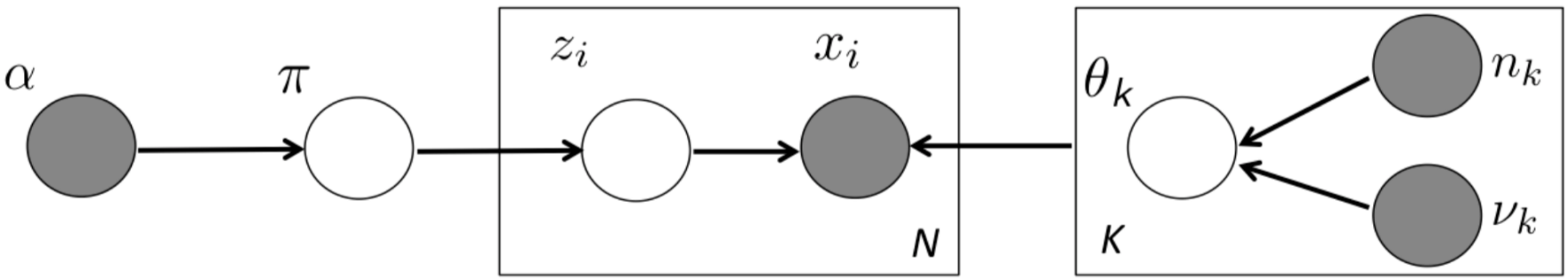


211 million examples, 44 billion parameters (# features $\times$ # classes)

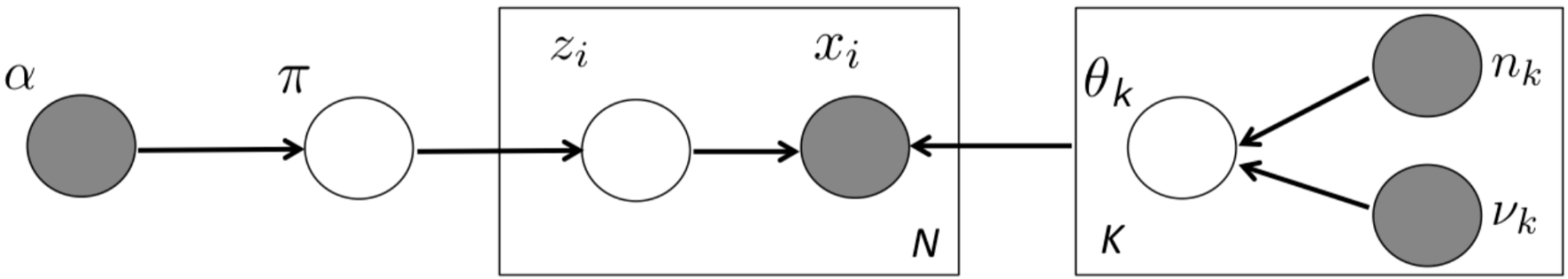
Outline

1. Introduction
2. Distributed Parameter Estimation
3. Thesis Roadmap
   - Multinomial Logistic Regression (KDD 2019)
   - Mixture Models (AISTATS 2019)
   - Latent Collaborative Retrieval (NIPS 2014)
4. Conclusion and Future Directions
5. Appendix

Mixture Of Exponential Families



Mixture Of Exponential Families



- k-means
- Gaussian Mixture Models (GMM)
- Latent Dirichlet Allocation (LDA)
- Stochastic Mixed Membership Models

Generative Model

Generative Model

- **Draw Cluster Proportions π**

$$p(\pi|\alpha) = \text{Dirichlet}(\alpha)$$

Generative Model

- **Draw Cluster Proportions** π

$$p(\pi|\alpha) = \text{Dirichlet}(\alpha)$$

- **Draw Cluster Parameters** for $k = 1, \ldots, K$

$$p(\theta_k|n_k, \nu_k) = \exp\left(\langle n_k \cdot \nu_k, \theta_k \rangle - n_k \cdot g(\theta_k) - h(n_k, \nu_k)\right)$$

Generative Model

- **Draw Cluster Proportions** π

$$p(\pi|\alpha) = \text{Dirichlet}(\alpha)$$

- **Draw Cluster Parameters** for $k = 1, \ldots, K$

$$p(\theta_k|n_k, \nu_k) = \exp\left(\langle n_k \cdot \nu_k, \theta_k \rangle - n_k \cdot g(\theta_k) - h(n_k, \nu_k)\right)$$

- **Draw Observations** for $i = 1, \ldots, N$

$$p(z_i|\pi) = \text{Multinomial}(\pi)$$

$$p(x_i|z_i, \theta) = \exp\left(\langle \phi(x_i, z_i), \theta_{z_i} \rangle - g(\theta_{z_i})\right)$$

Generative Model

- **Draw Cluster Proportions** π

$$p(\pi|\alpha) = \text{Dirichlet}(\alpha)$$

- **Draw Cluster Parameters** for $k = 1, \ldots, K$

$$p(\theta_k|n_k, \nu_k) = \exp\left(\langle n_k \cdot \nu_k, \theta_k \rangle - n_k \cdot g(\theta_k) - h(n_k, \nu_k)\right)$$

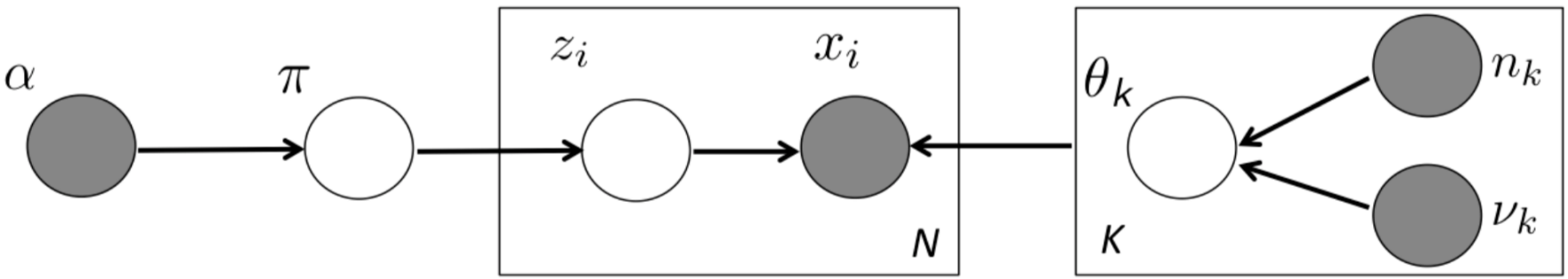
- **Draw Observations** for $i = 1, \ldots, N$

$$p(z_i|\pi) = \text{Multinomial}(\pi)$$

$$p(x_i|z_i, \theta) = \exp\left(\langle \phi(x_i, z_i), \theta_{z_i} \rangle - g(\theta_{z_i})\right)$$

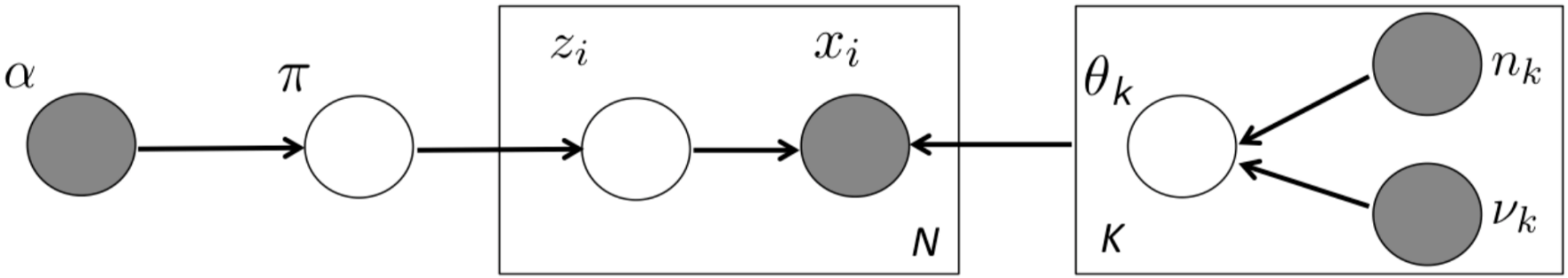
- $p(\theta_k|n_k, \nu_k)$ is conjugate to $p(x_i|z_i = k, \theta_k)$
- $p(\pi|\alpha)$ is conjugate to $p(z_i|\pi)$

Joint Distribution



$$p(\pi, \theta, z, x|\alpha, n, \nu) = p(\pi|\alpha) \cdot \prod_{k=1}^{K} p(\theta_k|n_k, \nu_k) \cdot \prod_{i=1}^{N} p(z_i|\pi) \cdot p(x_i|z_i, \theta)$$

Joint Distribution



$$p(\pi, \theta, z, x|\alpha, n, \nu) = p(\pi|\alpha) \cdot \prod_{k=1}^{K} p(\theta_k|n_k, \nu_k) \cdot \prod_{i=1}^{N} p(z_i|\pi) \cdot p(x_i|z_i, \theta)$$

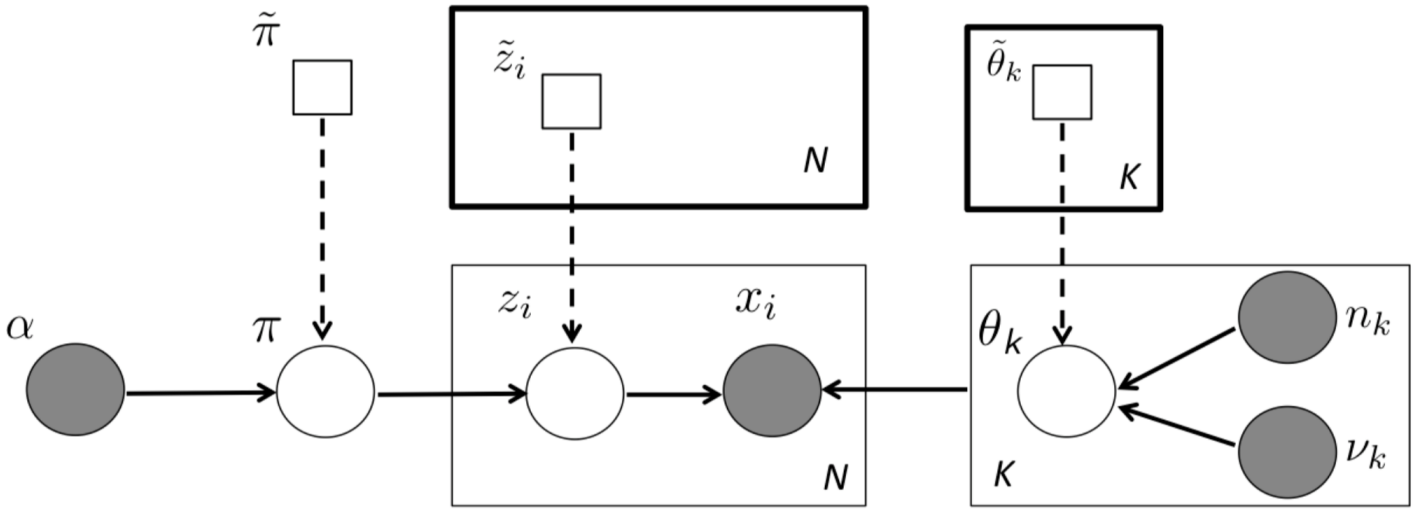
Want to estimate the posterior: $p(\pi, \theta, z|x, \alpha, n, \nu)$

Variational Inference [Blei et al., 2016]

Approximate $p(\pi, \theta, z | x, \alpha, n, \nu)$ with a simpler (factorized) distribution q

$$q\left(\pi, \theta, z | \tilde{\pi}, \tilde{\theta}, \tilde{z}\right) = q(\pi|\tilde{\pi}) \cdot \prod_{k=1}^{K} q\left(\theta_k | \tilde{\theta}_k\right) \cdot \prod_{i=1}^{N} q(z_i | \tilde{z}_i)$$

Variational Inference



Evidence Based Lower Bound (ELBO)

$$\mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}\right) = \mathbb{E}_{q\left(\pi,\theta,z|\tilde{\pi},\tilde{\theta},\tilde{z}\right)}\left[\log p\left(\pi, \theta, z, x|\alpha, n, \nu\right)\right] - \mathbb{E}_{q\left(\pi,\theta,z|\tilde{\pi},\tilde{\theta},\tilde{z}\right)}\left[\log q\left(\pi, \theta, z|\tilde{\pi}, \tilde{\theta}, \tilde{z}\right)\right]$$

Evidence Based Lower Bound (ELBO)

$$\mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}\right) = \mathbb{E}_{q\left(\pi,\theta,z|\tilde{\pi},\tilde{\theta},\tilde{z}\right)}\left[\log p\left(\pi, \theta, z, x|\alpha, n, \nu\right)\right] - \mathbb{E}_{q\left(\pi,\theta,z|\tilde{\pi},\tilde{\theta},\tilde{z}\right)}\left[\log q\left(\pi, \theta, z|\tilde{\pi}, \tilde{\theta}, \tilde{z}\right)\right]$$

Variational Inference (VI)

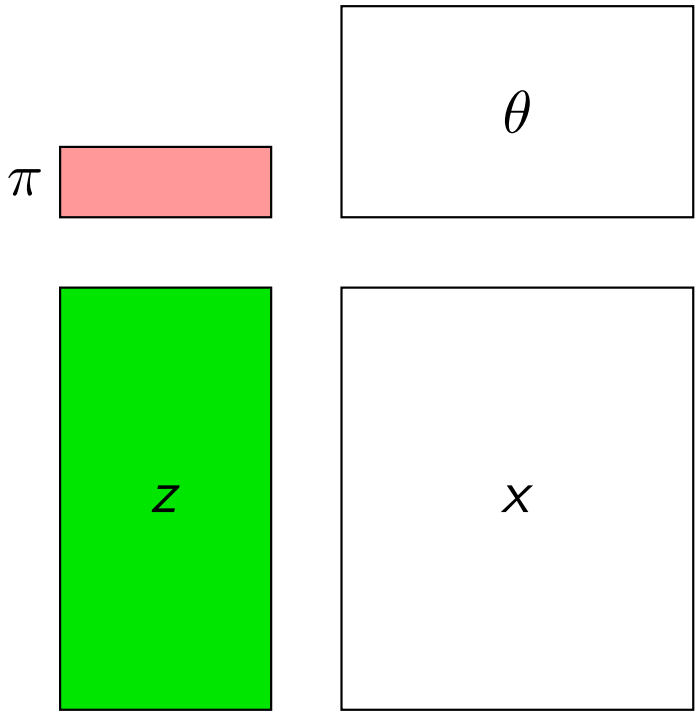
- M-step
  - $\max_{\tilde{\pi}} \mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}\right)$
  - $\max_{\tilde{\theta}} \mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}\right)$
- E-step
  - $\max_{\tilde{z}} \mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}\right)$

Variational Inference: M-step

- $\max_{\tilde{\pi}} \mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}\right)$

$$\tilde{\pi}_k = \alpha + \sum_{i=1}^{N} \tilde{z}_{i,k}$$

Access Pattern of Variables: $\tilde{\pi}$ update



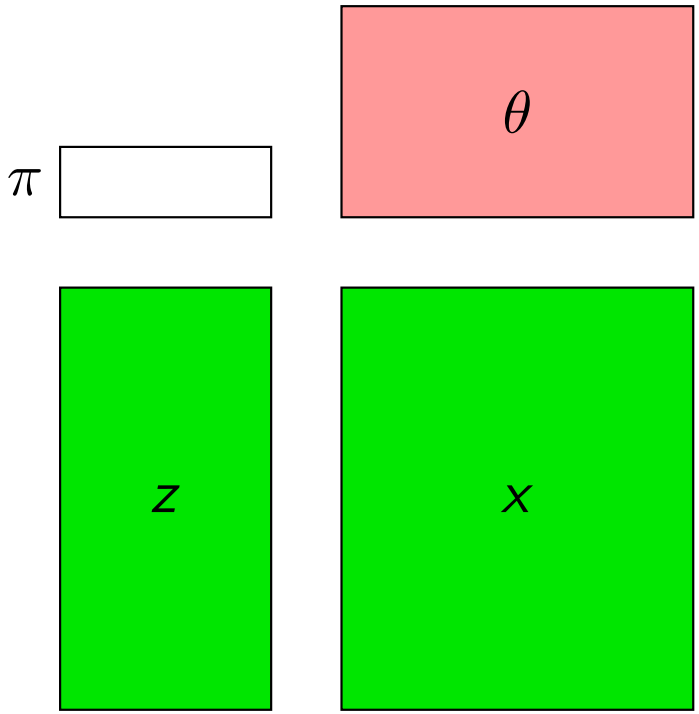
Variational Inference: M-step

- $\max_{\tilde{\theta}} \mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}\right)$

$$\tilde{n}_k = n_k + \sum_{i=1}^{N} \tilde{z}_{i,k}$$

$$\tilde{\nu}_k = n_k \cdot \nu_k + \sum_{i=1}^{N} \tilde{z}_{i,k} \cdot \phi\left(x_i, k\right)$$

Access Pattern of Variables: $\tilde{\theta}$ update

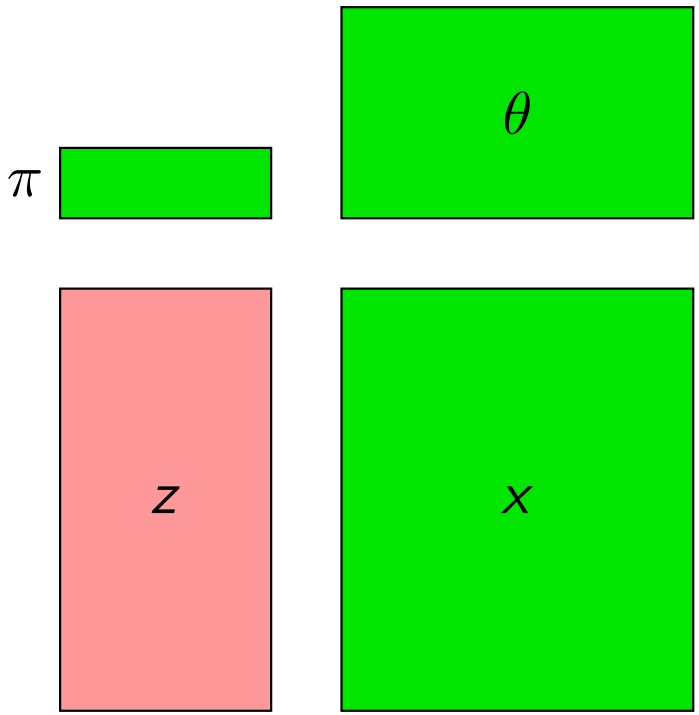


Variational Inference: E-step

- $\max_{\tilde{z}} \mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}\right)$

$$\tilde{z}_{i,k} \propto \exp\left(\psi\left(\tilde{\pi}_k\right) - \psi\left(\sum_{k'=1}^{K} \tilde{\pi}_{k'}\right) + \left\langle \phi\left(x_i, k\right), \mathbb{E}_{q\left(\theta_k|\tilde{\theta}_k\right)}\left[\theta_k\right]\right\rangle - \mathbb{E}_{q\left(\theta_k|\tilde{\theta}_k\right)}\left[g\left(\theta_k\right)\right]\right)$$

Access Pattern of Variables: $\tilde{z}$ update



Stochastic Variational Inference (SVI) [Hoffman et al., 2013]

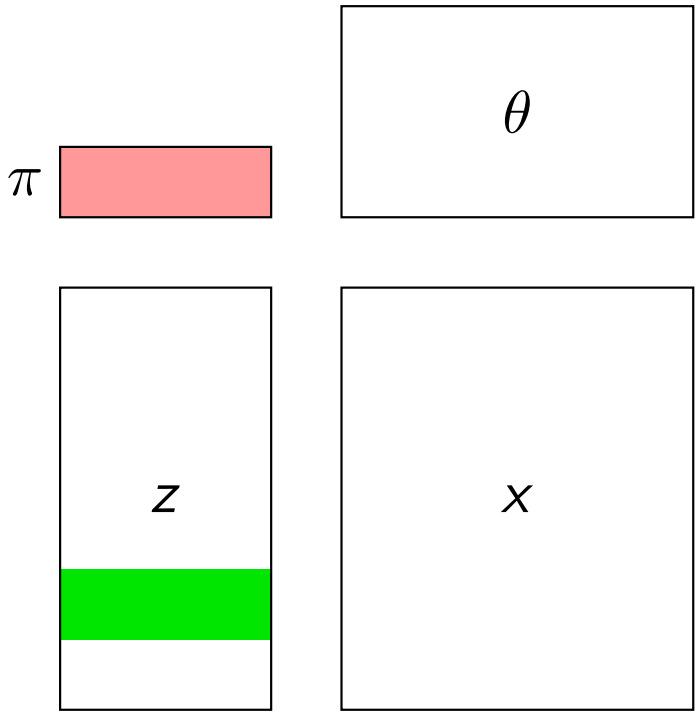
- M-step
  - $\max_{\tilde{\pi}} \mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}\right)$
  - $\max_{\tilde{\theta}} \mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}\right)$
- E-step
  - $\max_{\tilde{z}_i} \mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}_{\setminus i}, \tilde{z}_i\right)$

Stochastic Variational Inference (SVI) [Hoffman et al., 2013]

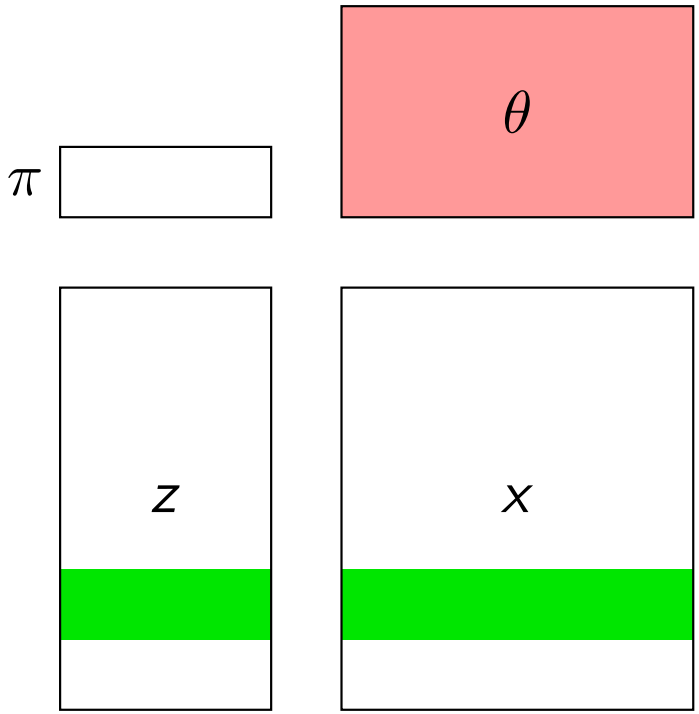
- M-step
  - $\max_{\tilde{\pi}} \mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}\right)$
  - $\max_{\tilde{\theta}} \mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}\right)$
- E-step
  - $\max_{\tilde{z}_i} \mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}_{\setminus i}, \tilde{z}_i\right)$

More frequent updates to global variables $\Longrightarrow$ Faster convergence

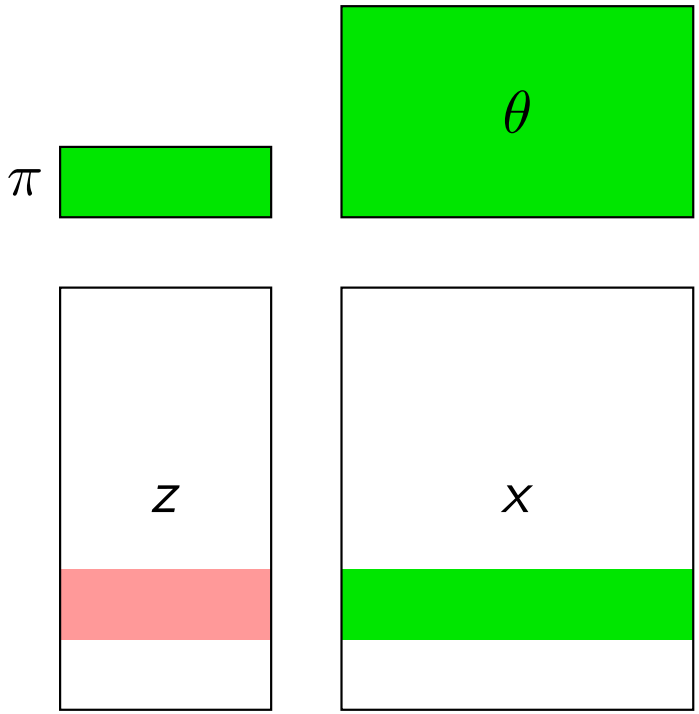
Access Pattern of Variables: $\tilde{\pi}$ update



Access Pattern of Variables: $\tilde{\theta}$ update



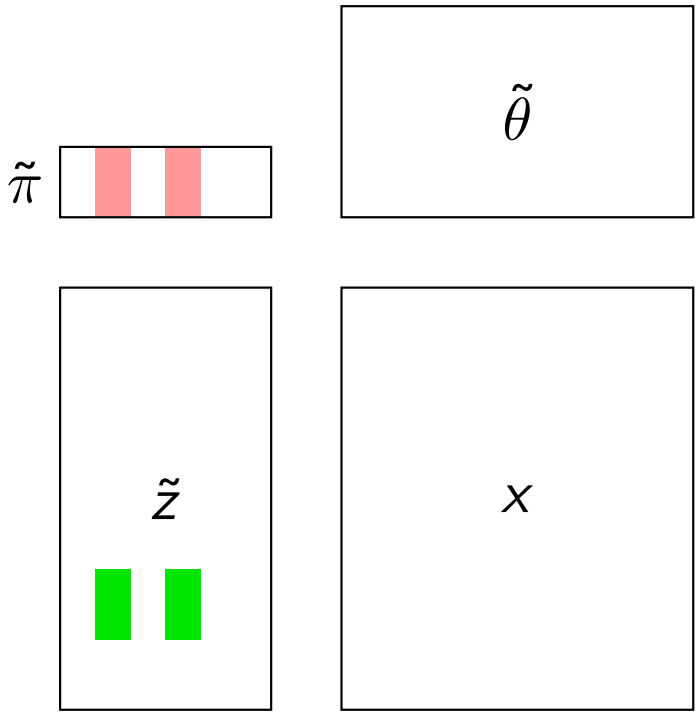
Access Pattern of Variables: $\tilde{z}$ update



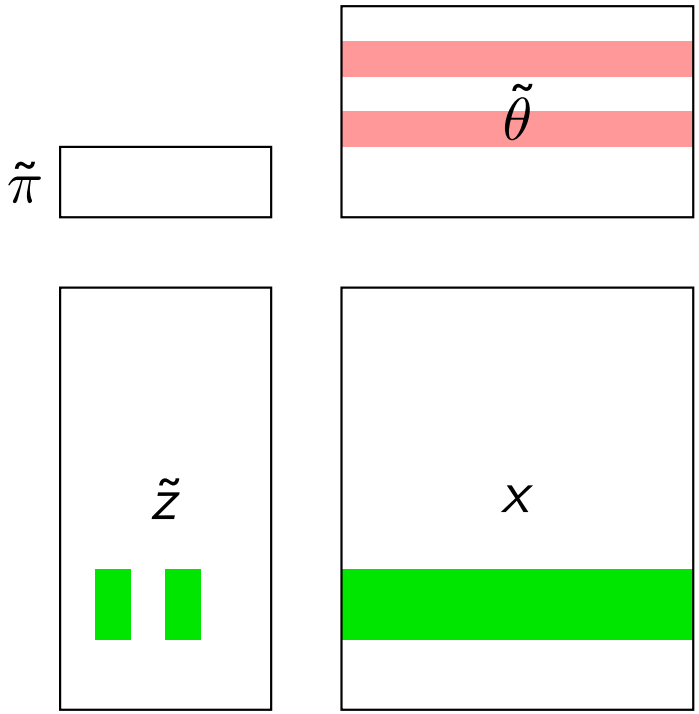
Extreme Stochastic Variational Inference (ESVI)

- M-step
  - $\max_{\tilde{\pi}_k, \tilde{\pi}_{k'}} \mathcal{L}\left([\tilde{\pi}_1, \ldots, \tilde{\pi}_k, \tilde{\pi}_{k+1}, \ldots, \tilde{\pi}_{k'}, \ldots, \tilde{\pi}_K], \tilde{\theta}, \tilde{z}\right)$
  - $\max_{\tilde{\theta}_k, \tilde{\theta}_{k'}} \mathcal{L}\left(\tilde{\pi}, \left[\tilde{\theta}_1, \ldots, \tilde{\theta}_k, \tilde{\theta}_{k+1}, \ldots, \tilde{\theta}_{k'}, \ldots, \tilde{\theta}_K\right], \tilde{z}\right)$
- E-step
  - $\max_{\tilde{z}_{i,k}, \tilde{z}_{i,k'}} \mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}_{\setminus i}, [\tilde{z}_{i,1}, \ldots, \tilde{z}_{i,k}, \tilde{z}_{i,k+1}, \ldots, \tilde{z}_{i,k'}, \ldots, \tilde{z}_{i,K}]\right)$

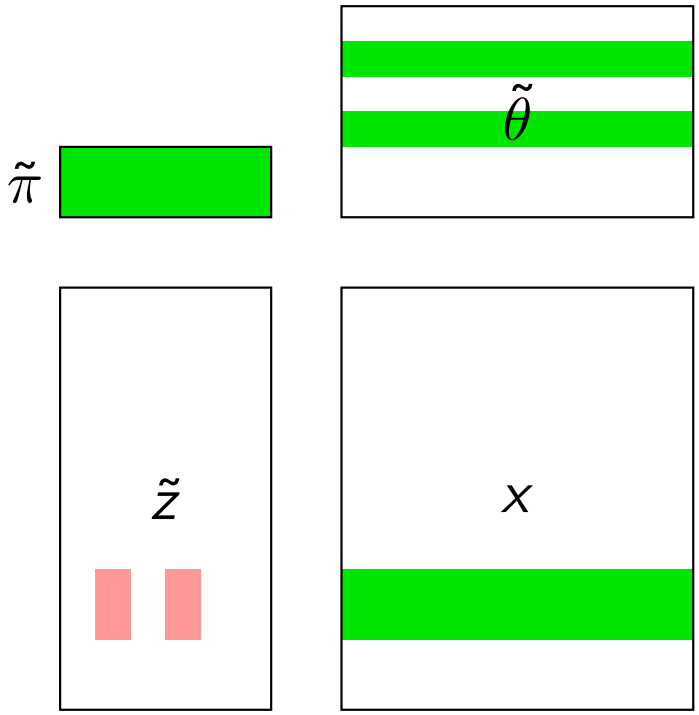
Access Pattern of Variables: $\tilde{\pi}$ update



Access Pattern of Variables: $\tilde{\theta}$ update



Access Pattern of Variables: $\tilde{z}$ update



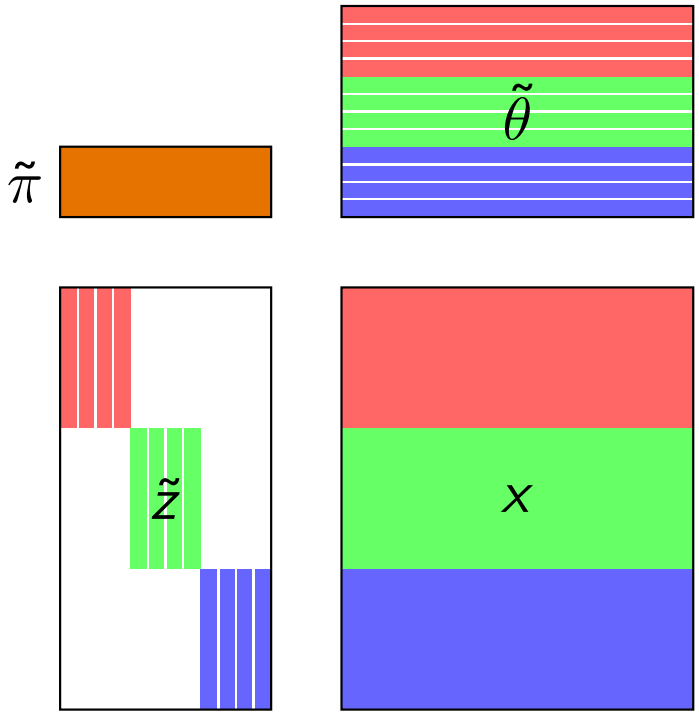
Key Observation

- M-step
  - $\max_{\tilde{\pi}_k, \tilde{\pi}_{k'}} \mathcal{L}\left([\tilde{\pi}_1, \ldots, \tilde{\pi}_k, \tilde{\pi}_{k+1}, \ldots, \tilde{\pi}_{k'}, \ldots, \tilde{\pi}_K], \tilde{\theta}, \tilde{z}\right)$
  - $\max_{\tilde{\theta}_k, \tilde{\theta}_{k'}} \mathcal{L}\left(\tilde{\pi}, \left[\tilde{\theta}_1, \ldots, \tilde{\theta}_k, \tilde{\theta}_{k+1}, \ldots, \tilde{\theta}_{k'}, \ldots, \tilde{\theta}_K\right], \tilde{z}\right)$
- E-step
  - $\max_{\tilde{z}_{i,k}, \tilde{z}_{i,k'}} \mathcal{L}\left(\tilde{\pi}, \tilde{\theta}, \tilde{z}_{\setminus i}, [\tilde{z}_{i,1}, \ldots, \tilde{z}_{i,k}, \tilde{z}_{i,k+1}, \ldots, \tilde{z}_{i,k'}, \ldots, \tilde{z}_{i,K}]\right)$

All the above updates can be done in parallel!

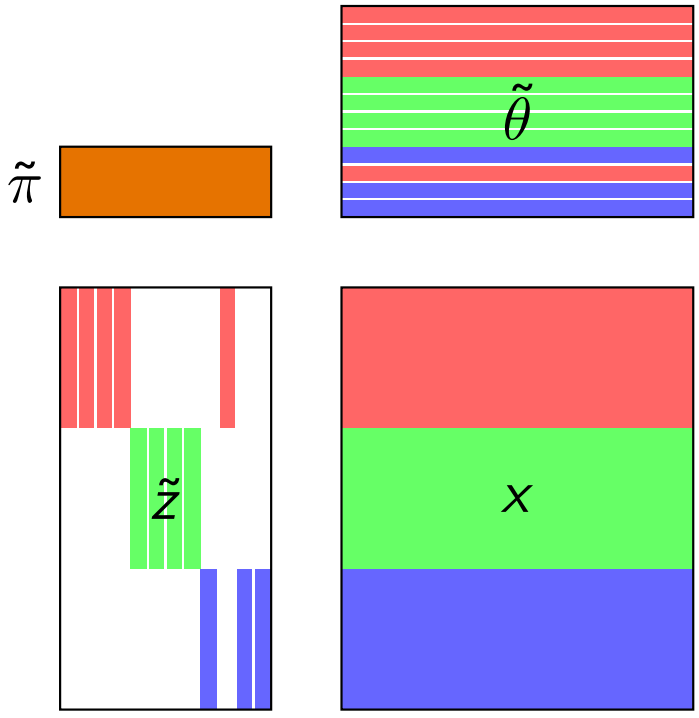
Distributing Computation

NOMAD [Yun et al., 2014]



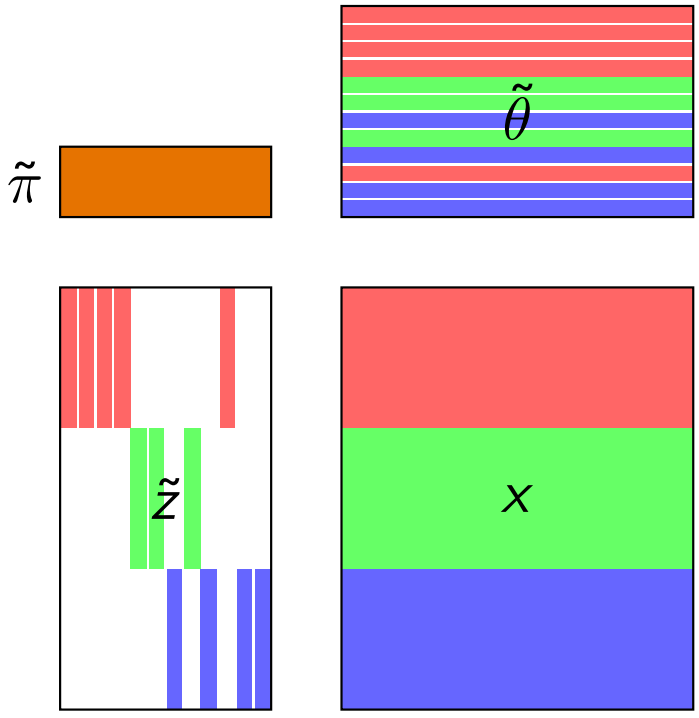
Distributing Computation

NOMAD [Yun et al., 2014]



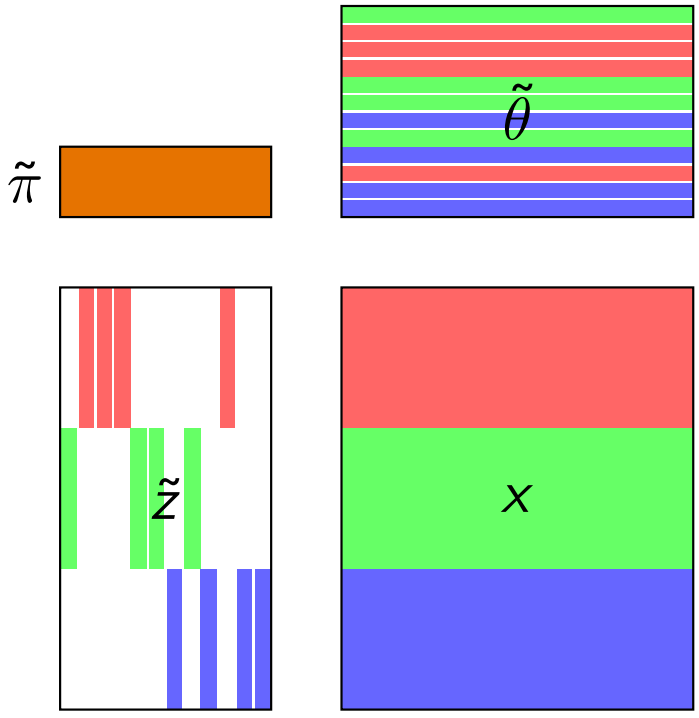
Distributing Computation

NOMAD [Yun et al., 2014]



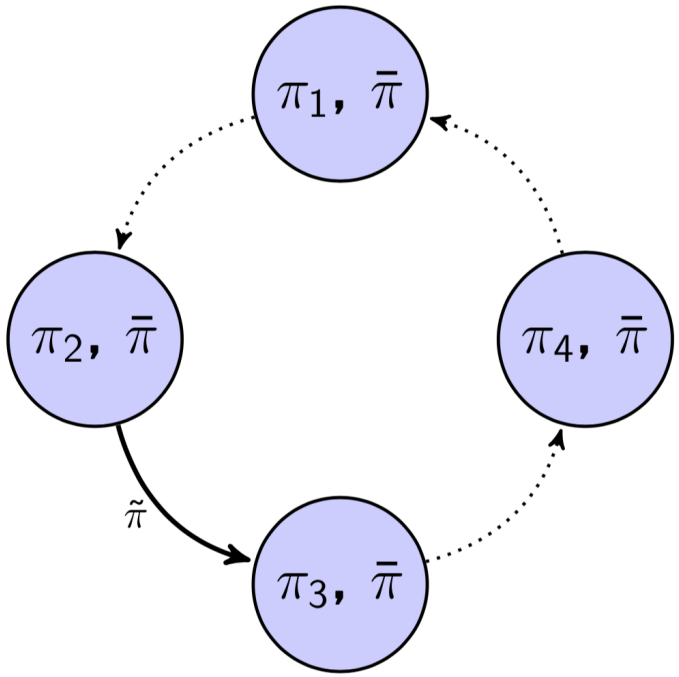
Distributing Computation

NOMAD [Yun et al., 2014]



Keeping $\tilde{\pi}$ in Sync

- Single global $\tilde{\pi}$
  - travels among machines
  - broadcasts local delta updates
- Every machine p: $(\pi_p, \bar{\pi})$
  - π_p: local working copy
  - $\bar{\pi}$: snapshot of global $\tilde{\pi}$



$\tilde{\pi} \leftarrow \tilde{\pi} + (\pi_3 - \bar{\pi})$
$\bar{\pi} \leftarrow \tilde{\pi}$
$\pi_3 \leftarrow \tilde{\pi}$

Experiments: Datasets

	# documents	# vocabulary	#words
NIPS	1,312	12,149	1,658,309
Enron	37,861	28,102	6,238,796
Ny Times	298,000	102,660	98,793,316
PubMed	8,200,000	141,043	737,869,083
UMBC-3B	40,599,164	3,431,260	3,013,004,127

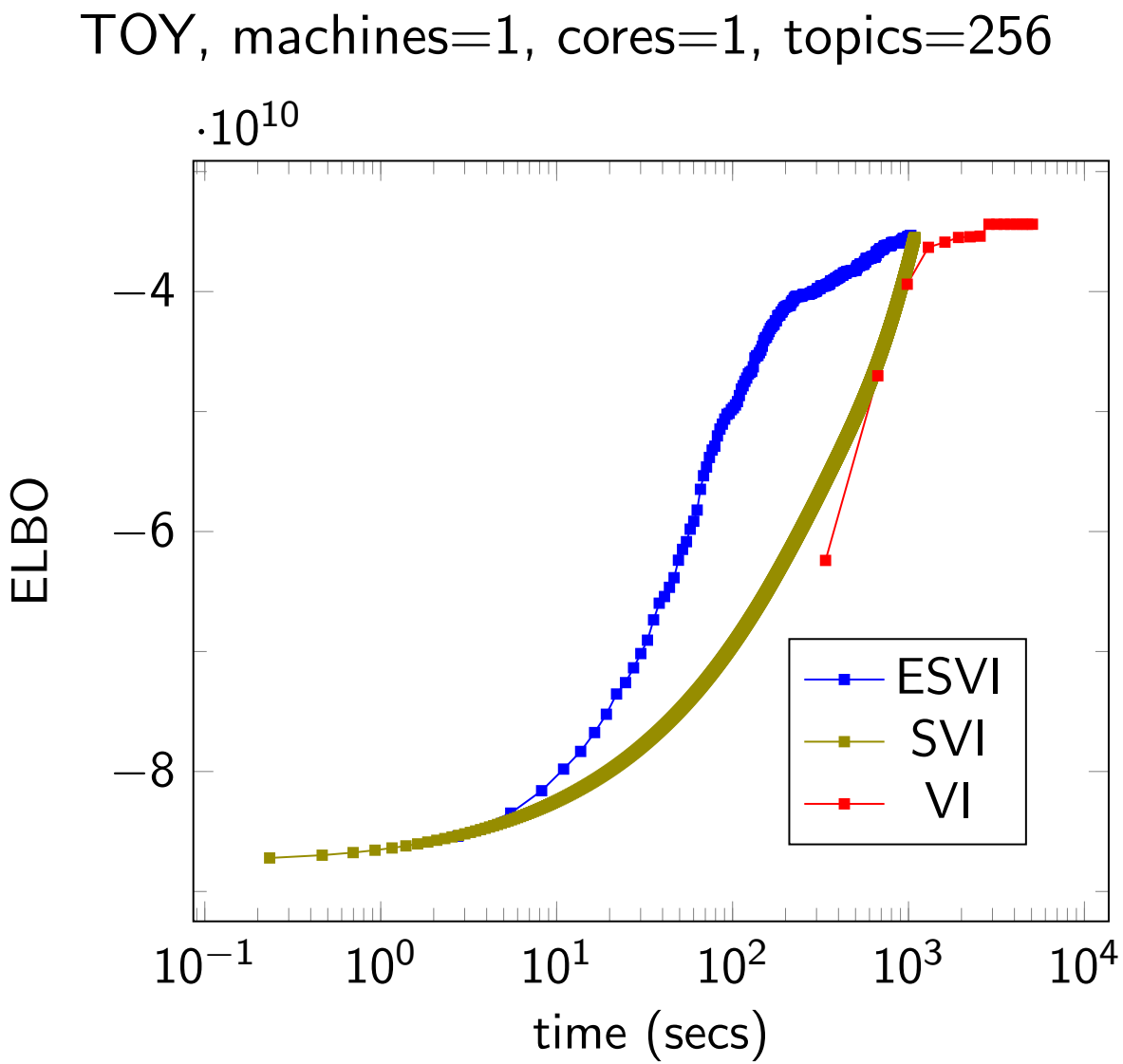
Mixture Models:

- Gaussian Mixture Model (GMM)
- Latent Dirichlet Allocation (LDA)

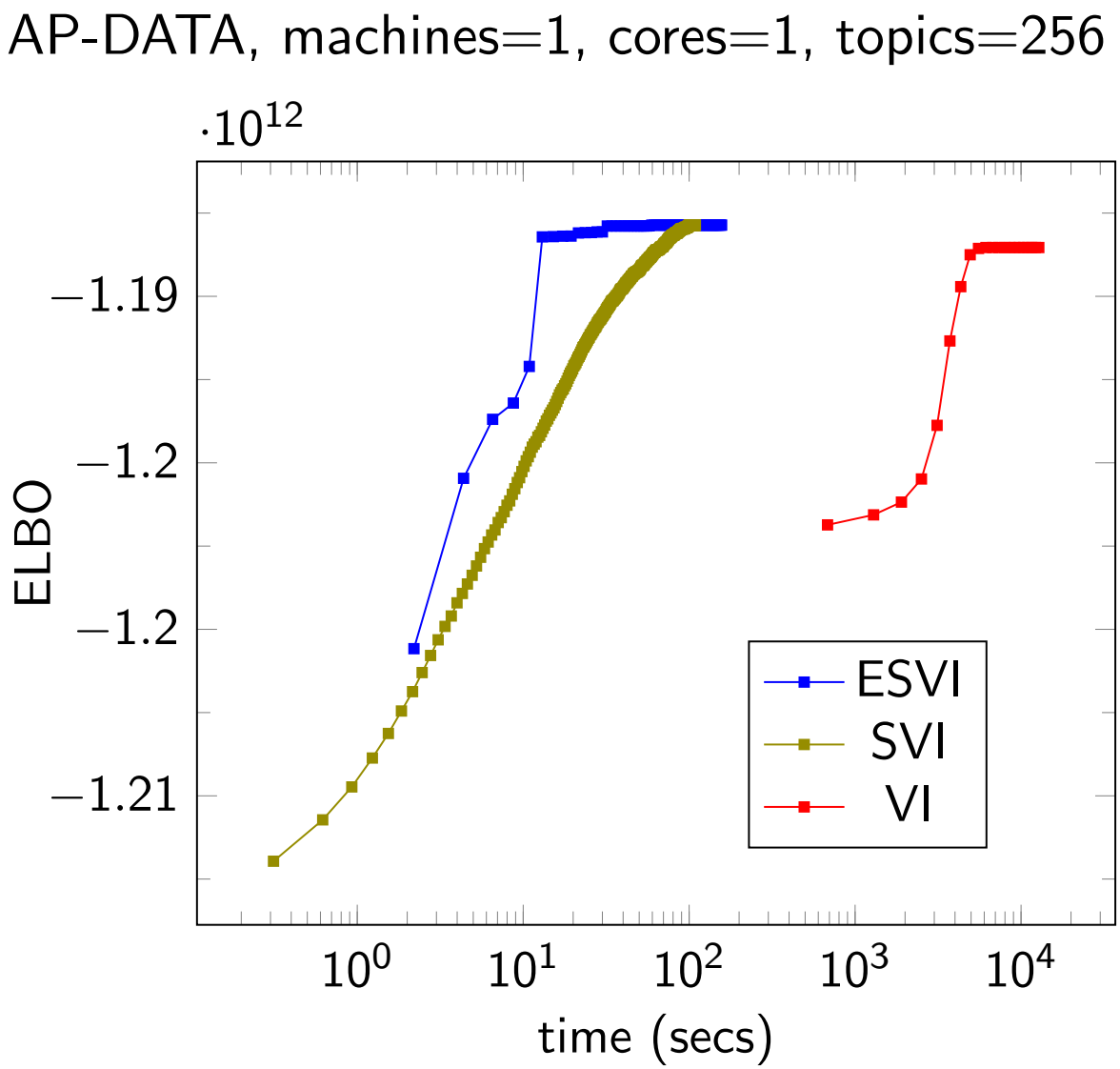
Methods:

- Variational Inference (VI)
- Stochastic Variational Inference (SVI)
- Extreme Stochastic Variational Inference (ESVI)

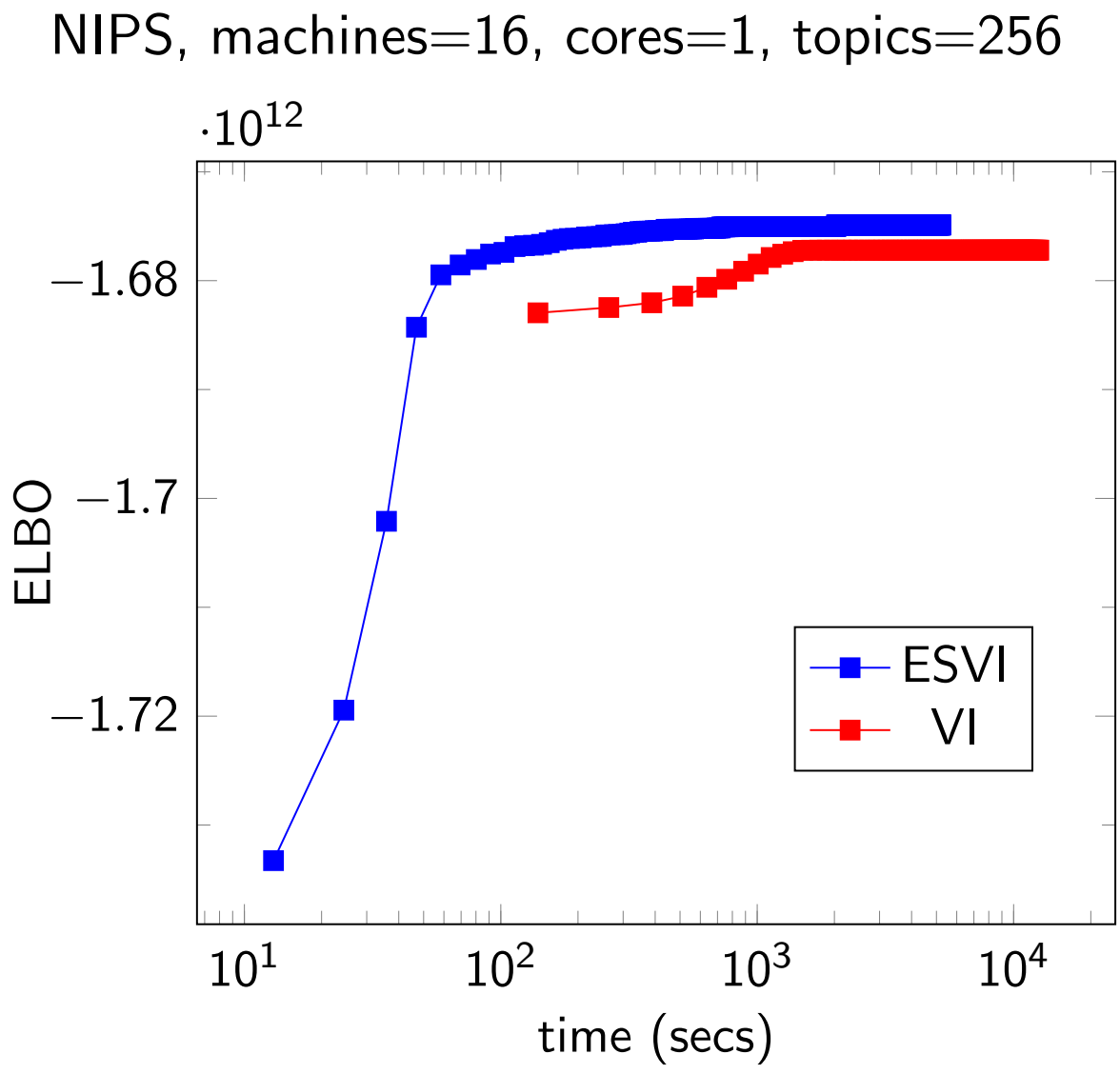
Experiments: Single Machine (GMM)



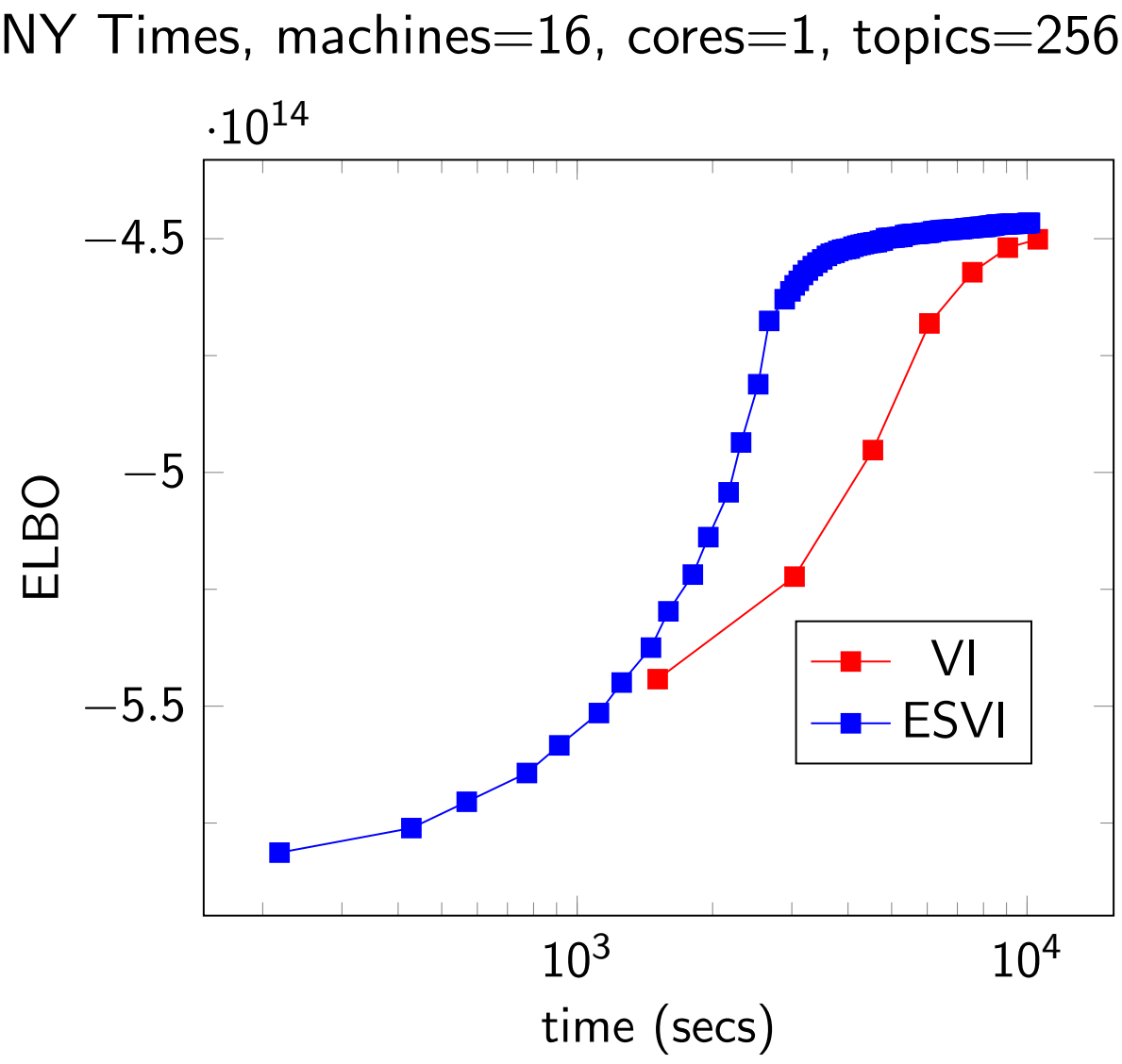
Experiments: Single Machine (GMM)



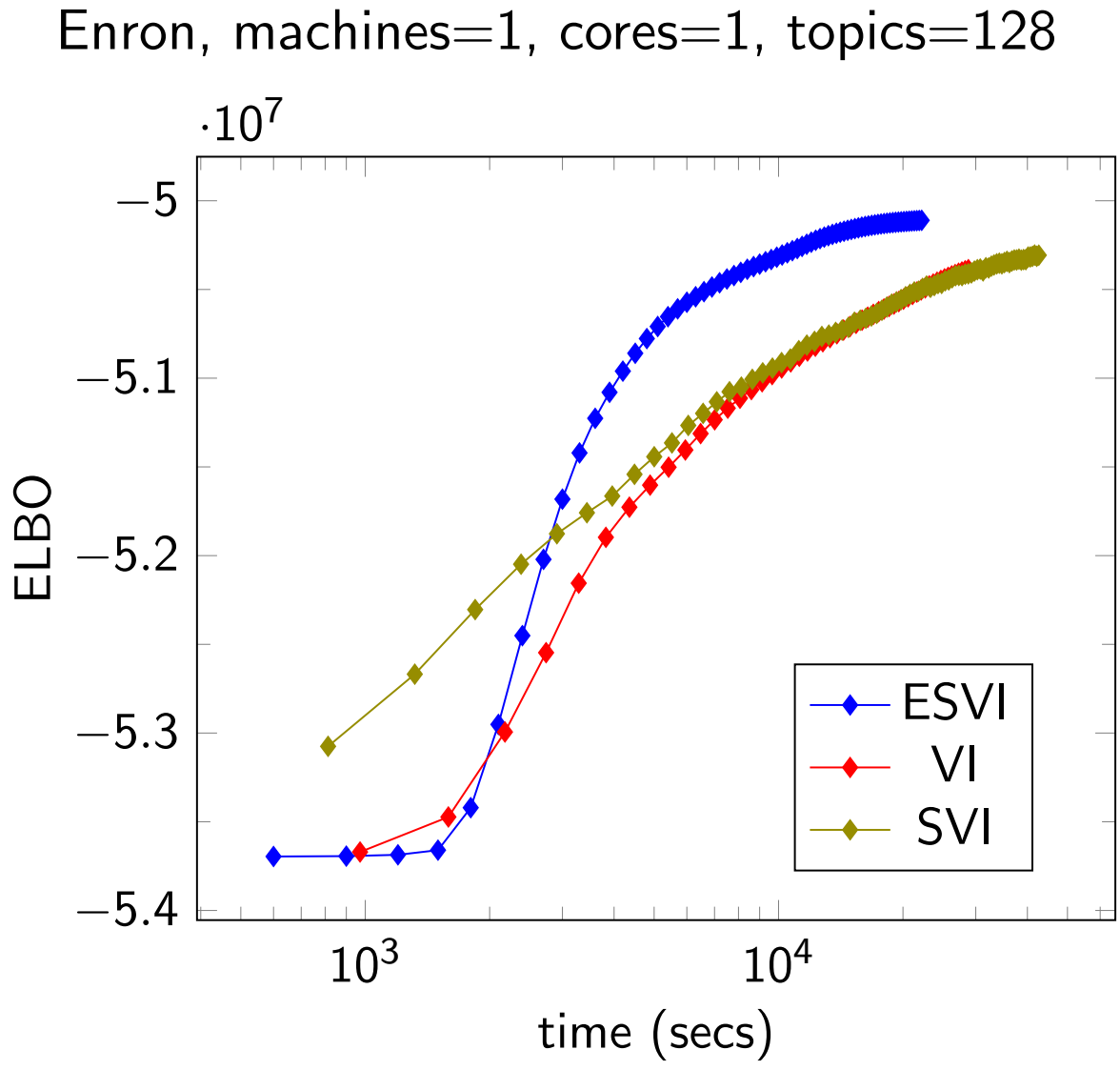
Experiments: Multi Machine (GMM)



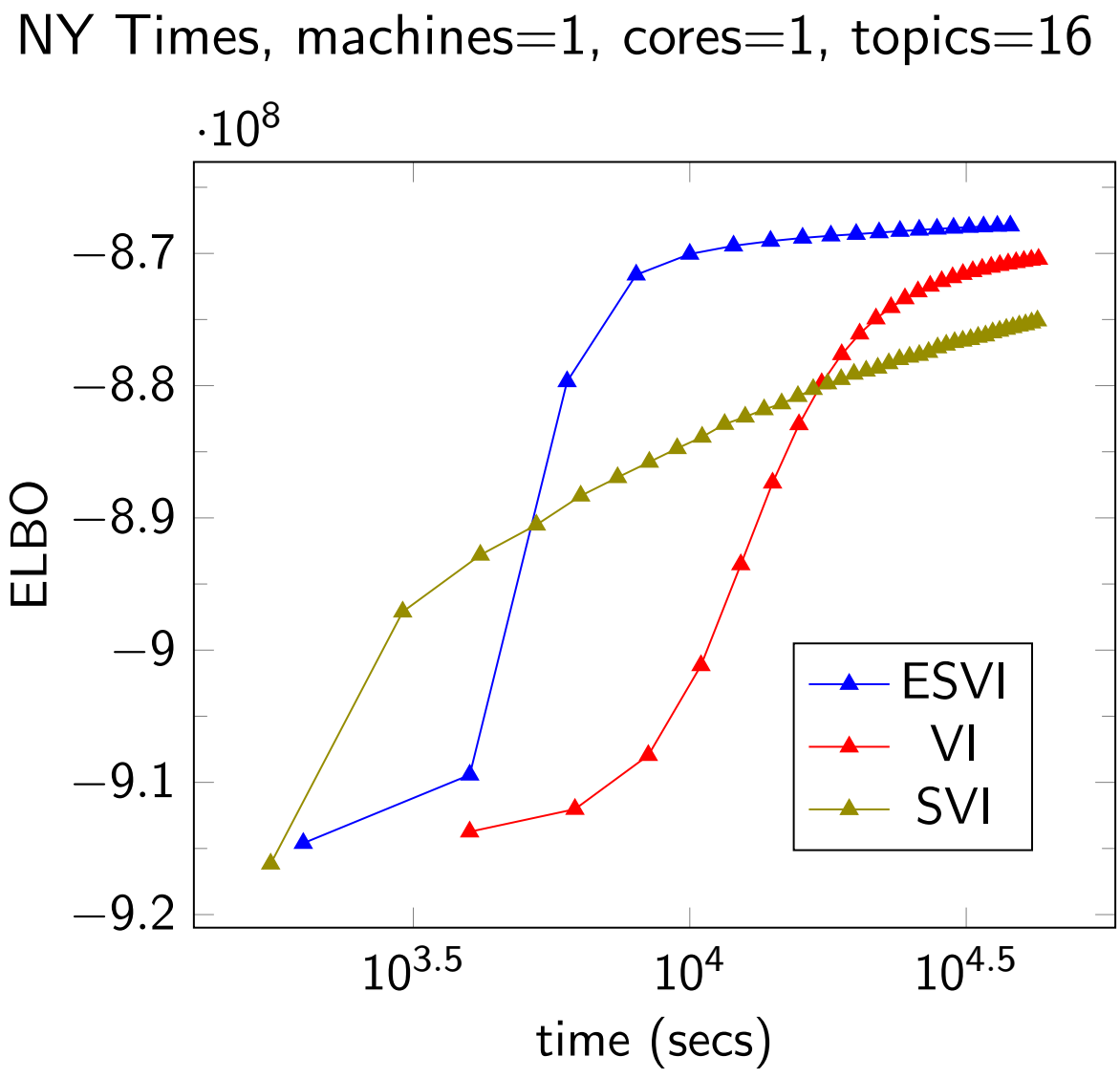
Experiments: Multi Machine (GMM)



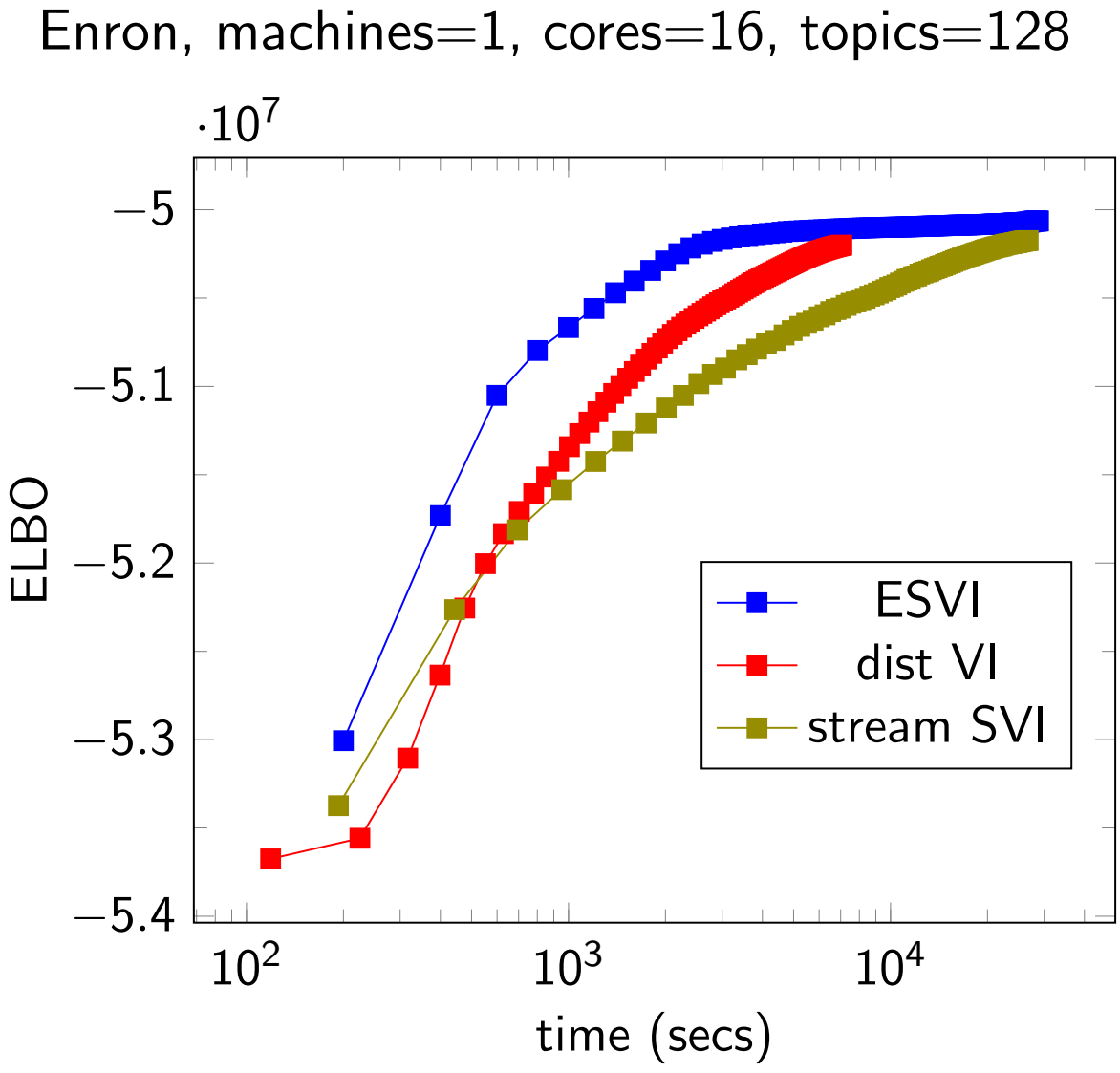
Experiments: Single Machine Single Core (LDA)



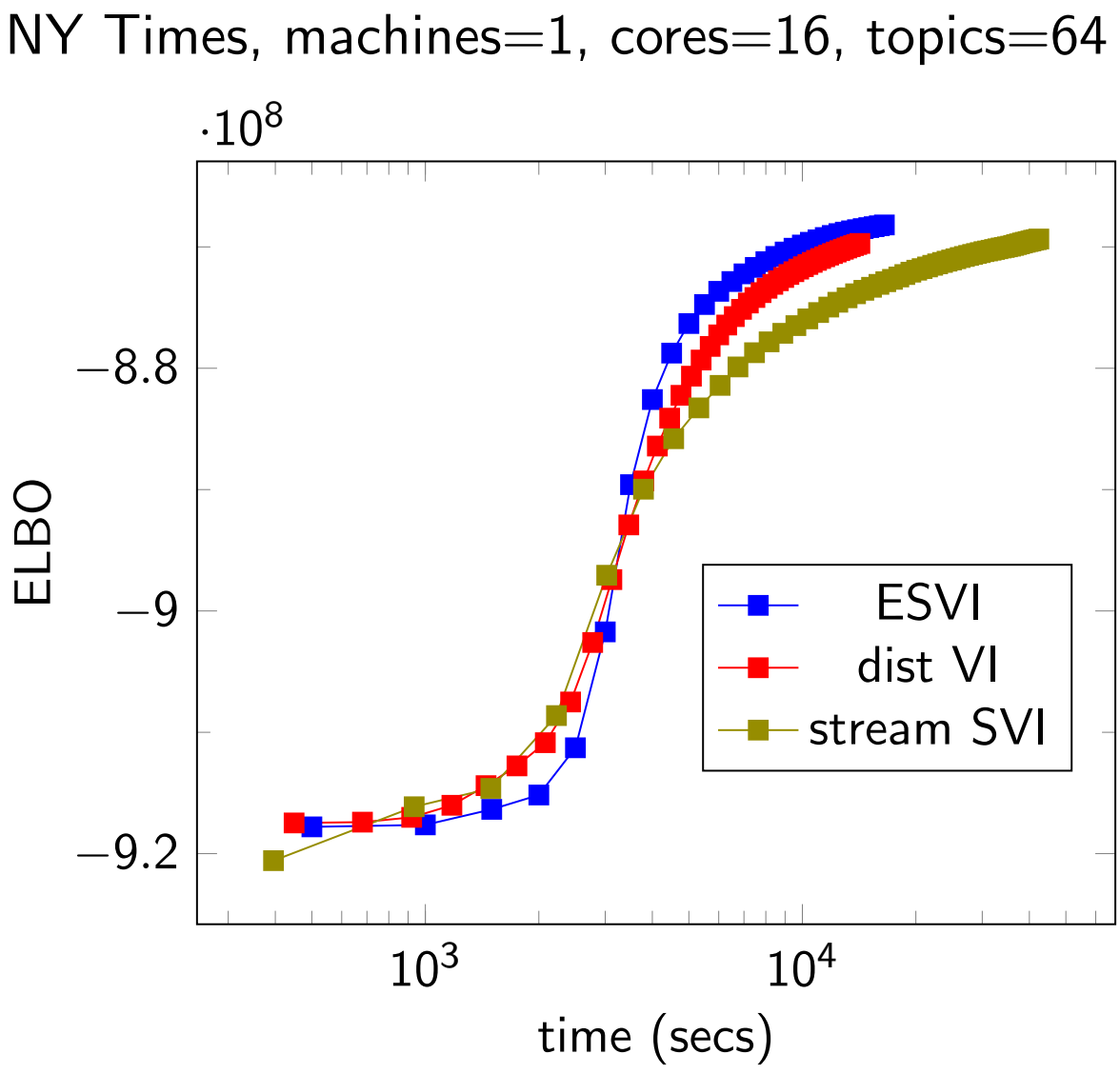
Experiments: Single Machine Single Core (LDA)



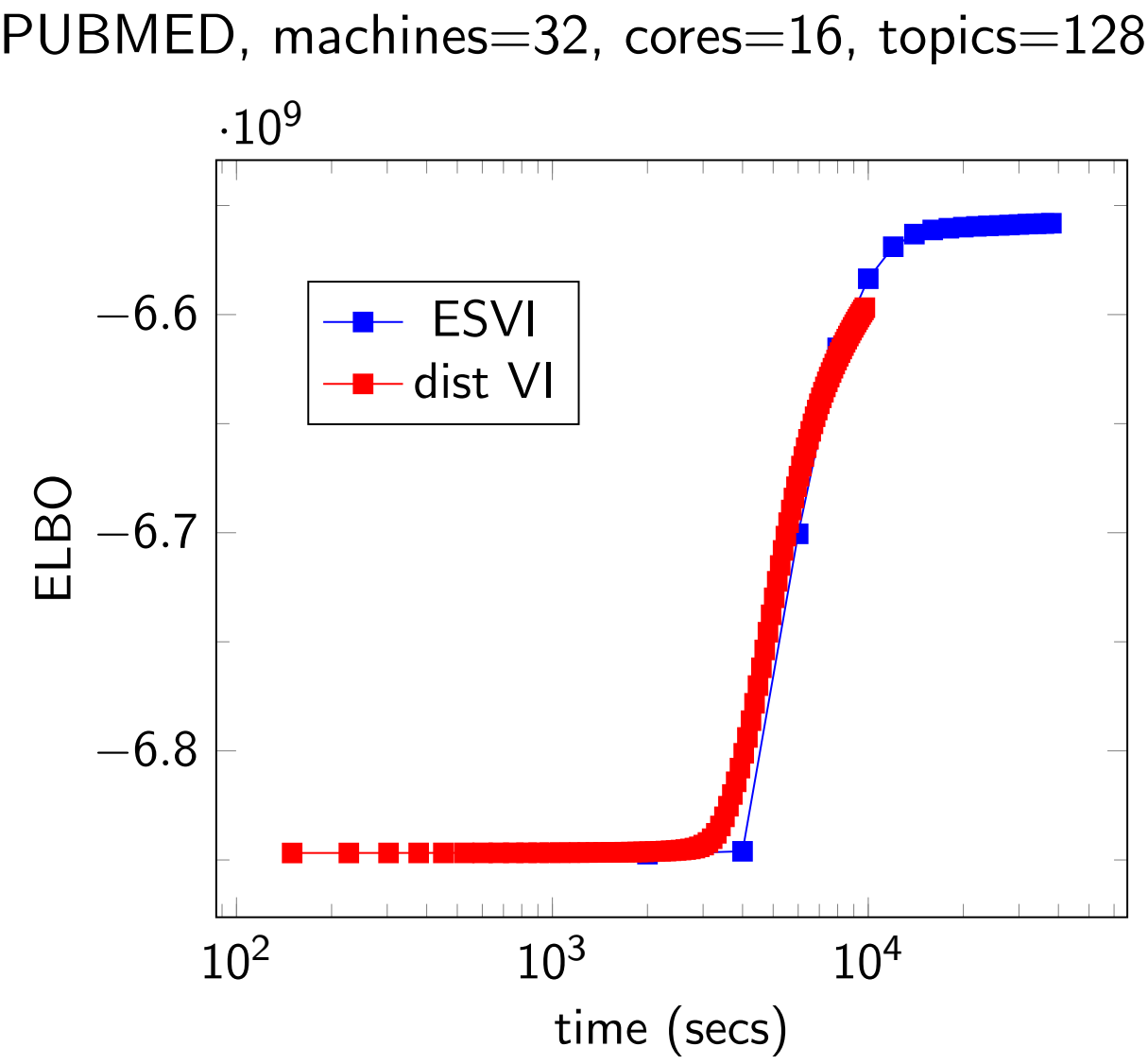
Experiments: Single Machine Multi Core (LDA)



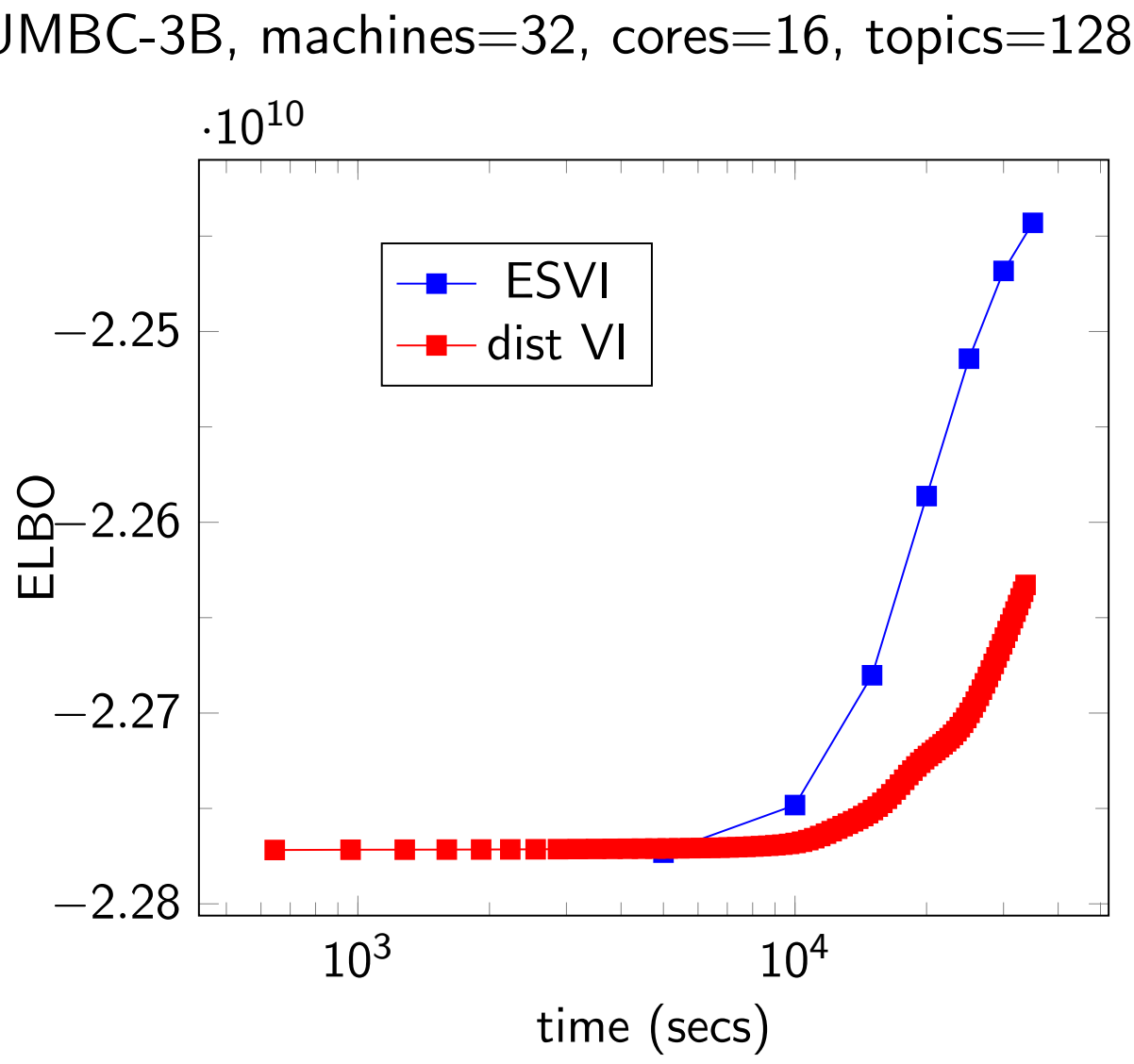
Experiments: Single Machine Multi Core (LDA)



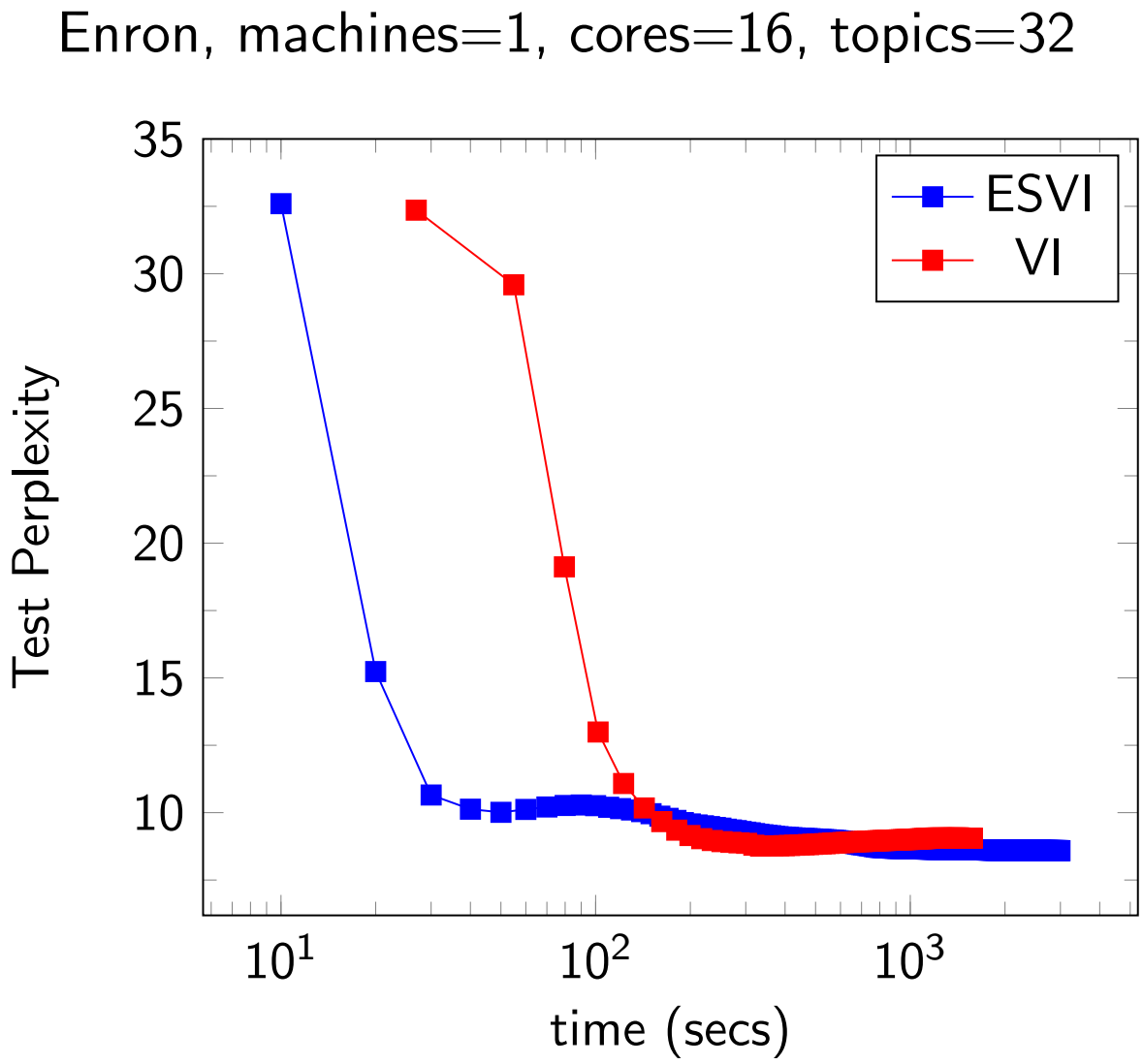
Experiments: Multi Machine Multi Core (LDA)



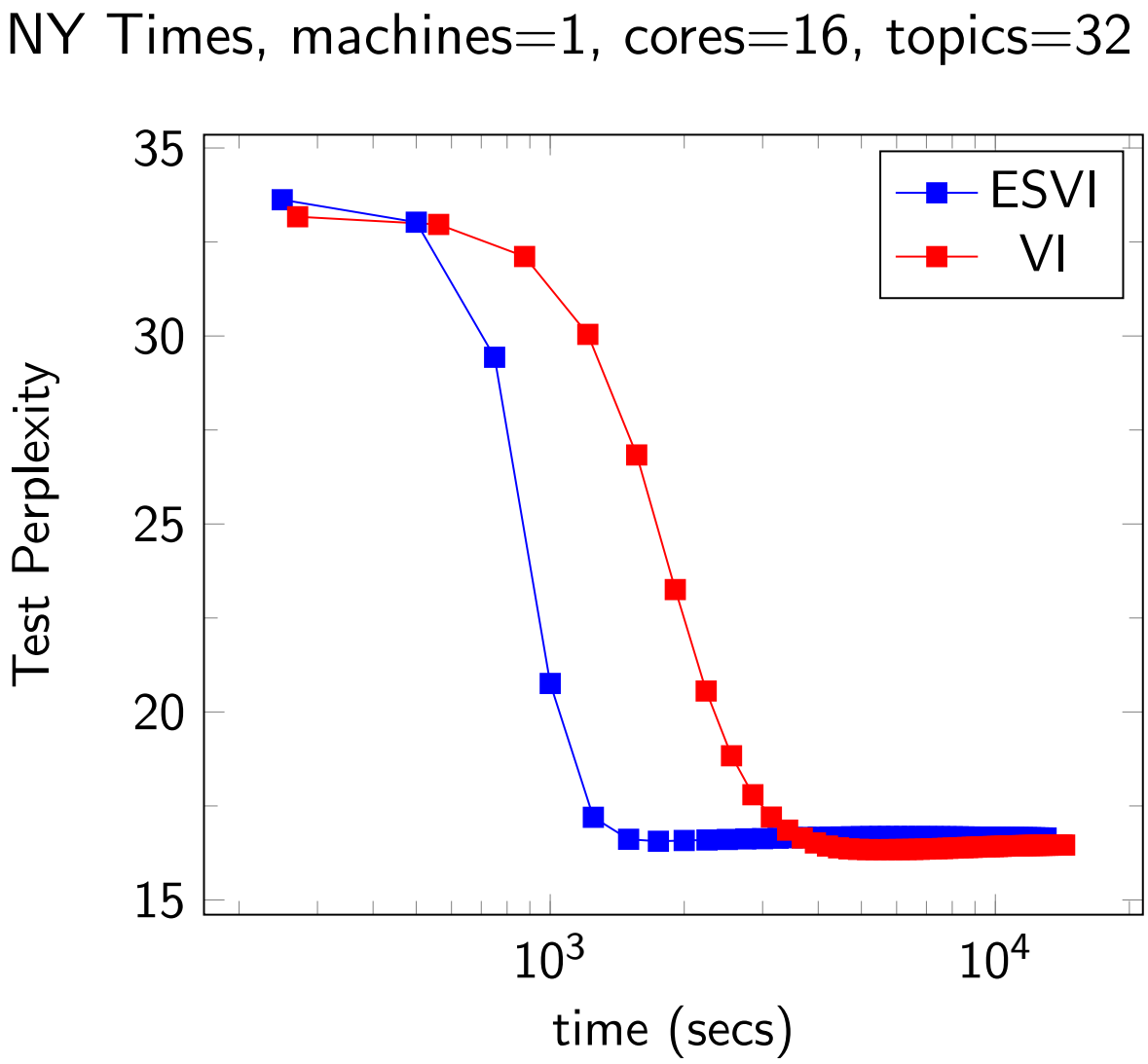
Experiments: Multi Machine Multi Core (LDA)



Experiments: Predictive Performance (LDA)



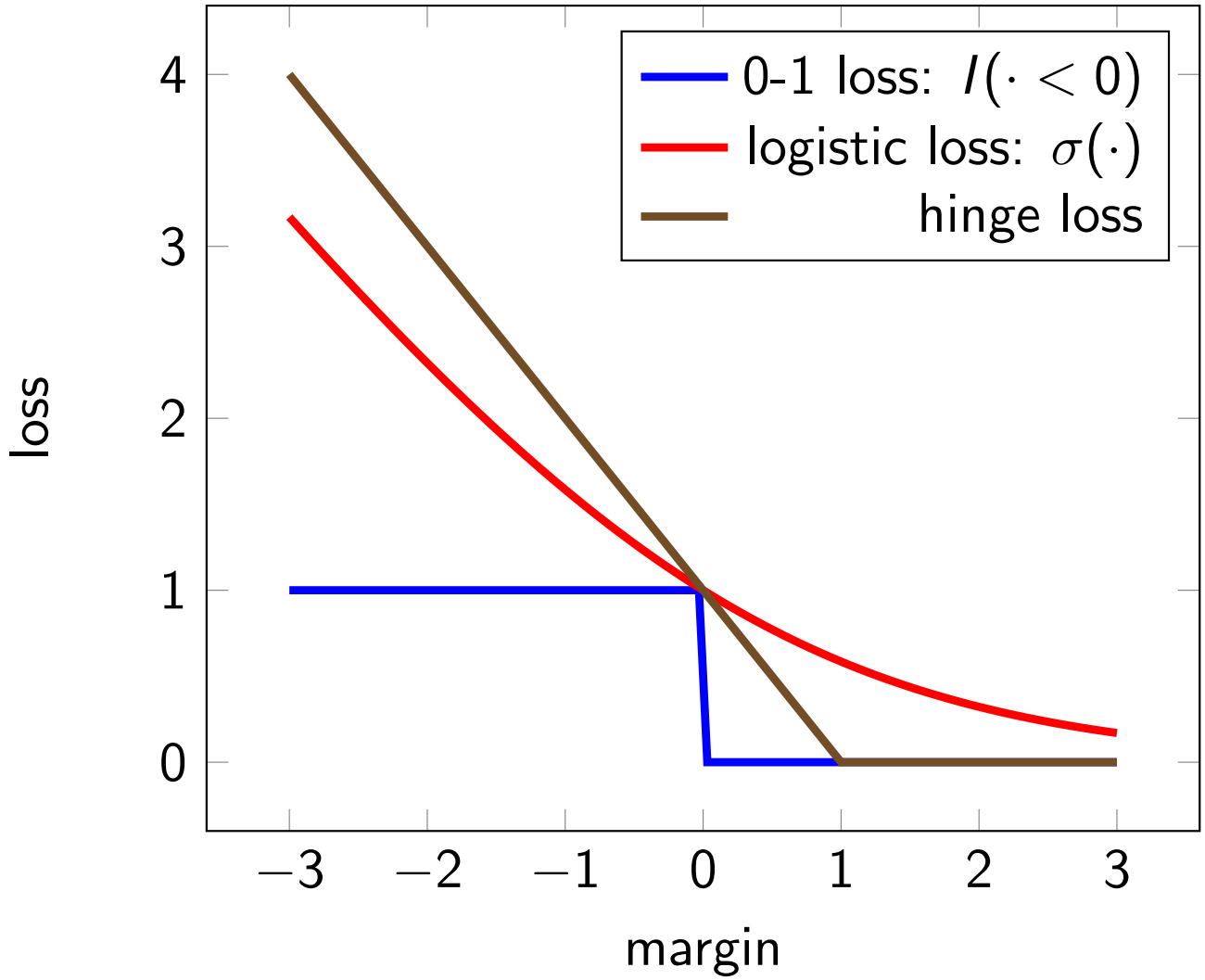
Experiments: Predictive Performance (LDA)



Outline

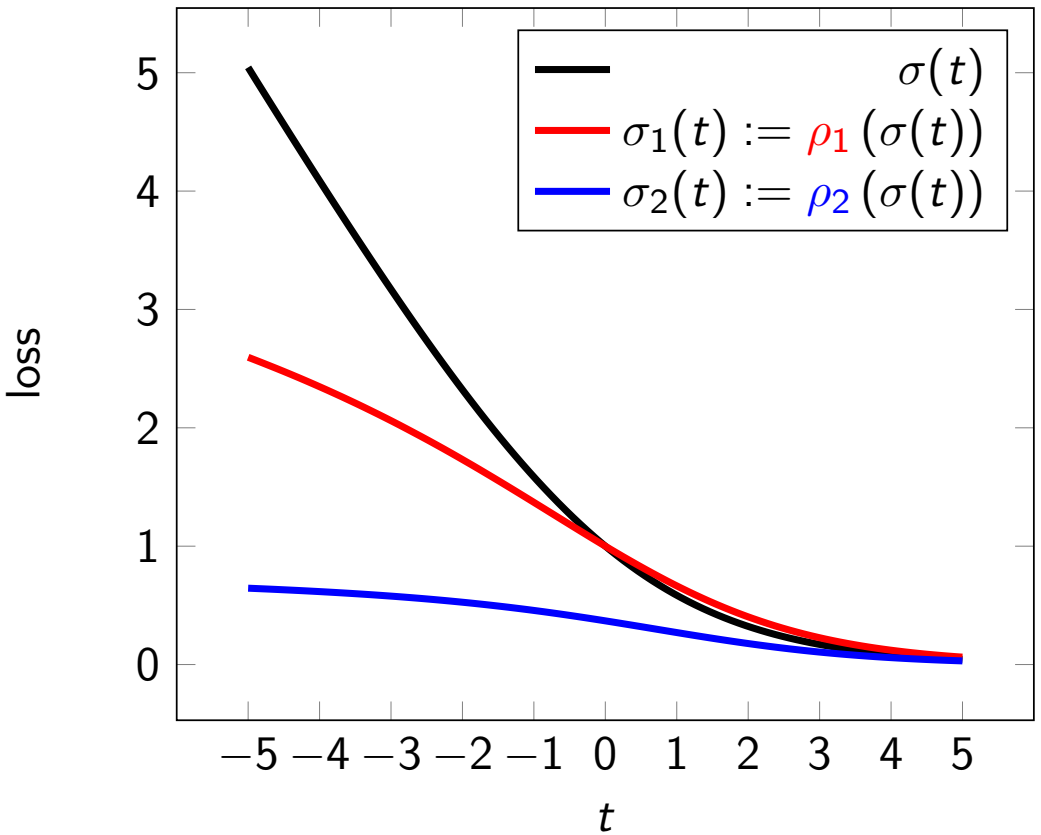
1. Introduction
2. Distributed Parameter Estimation
3. Thesis Roadmap
   - Multinomial Logistic Regression (KDD 2019)
   - Mixture Models (AISTATS 2019)
   - Latent Collaborative Retrieval (NIPS 2014)
4. Conclusion and Future Directions
5. Appendix

Binary Classification



Convex objective functions are sensitive to outliers

Robust Binary Classification



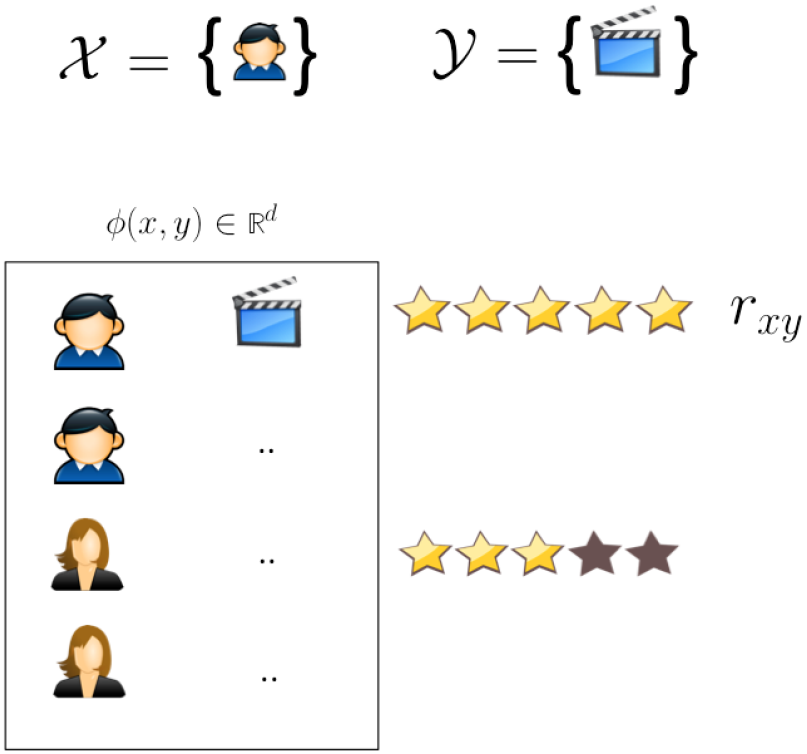
- $\rho_1(t) = \log_2(t+1)$
- $\rho_2(t) = 1 - \frac{1}{\log_2(t+2)}$

Bending the losses provides robustness

Objectives

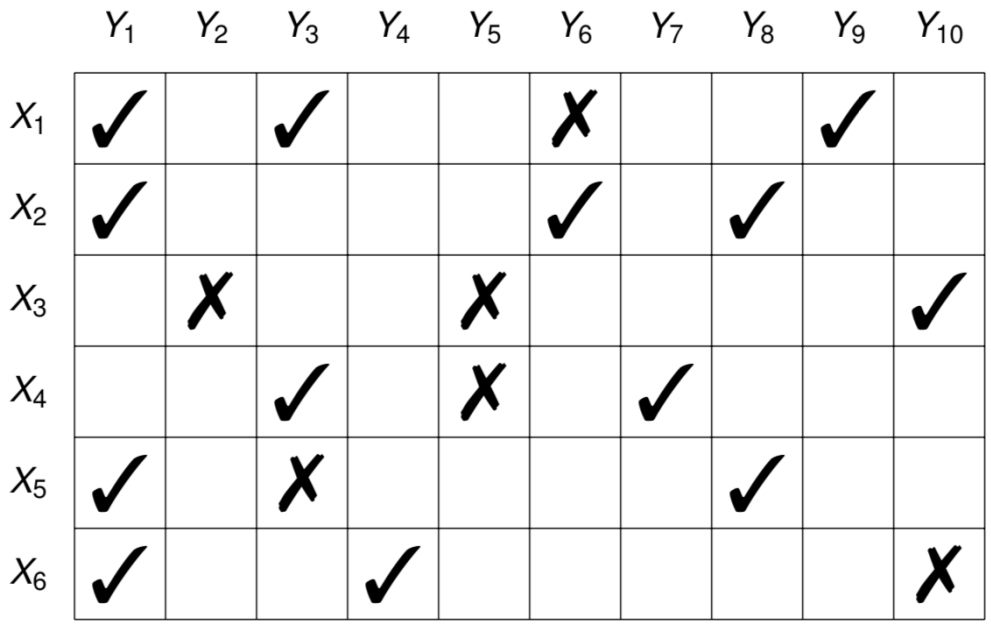
- Can we use this to design robust losses for ranking?
- Two scenarios:
  - **Explicit**: features, labels are observed (**Learning to Rank**)
  - **Implicit**: features, labels are latent (**Latent Collaborative Retrieval**)
- Distributed Optimization Algorithms

Explicit Setting - Learning to Rank



- Scoring function: $f(x, y) = \langle \phi(x, y), \mathbf{w} \rangle$

Implicit Setting - Latent Collaborative Retrieval



- Scoring function: $f(x, y) = \langle U_x, V_y \rangle$

Formulating a Robust Ranking Loss (RoBiRank)

Rank

$$\text{rank}_{\mathbf{w}}(x, y) = \sum_{y' \in \mathcal{Y}_x, y' \neq y} \mathbb{1}\left(f(x, y) - f(x, y') < 0\right)$$

Formulating a Robust Ranking Loss (RoBiRank)

Rank

$$\text{rank}_{\mathbf{w}}(x, y) = \sum_{y' \in \mathcal{Y}_x, y' \neq y} \mathbb{1}\left(f(x, y) - f(x, y') < 0\right)$$

Objective Function

$$L(\mathbf{w}) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}_x} r_{xy} \sum_{y' \in \mathcal{Y}_x, y' \neq y} \sigma\left(f(x, y) - f(x, y')\right)$$

where $\sigma =$ logistic loss.

Formulating a Robust Ranking Loss (RoBiRank)

Applying the robust transformation $\rho_2(t)$

Objective Function

$$L(\mathbf{w}) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}_x} r_{xy} \cdot \rho_2 \left(\sum_{y' \in \mathcal{Y}_x, y' \neq y} \sigma \left(f(x, y) - f(x, y') \right) \right)$$

where $\rho_2(t) = 1 - \frac{1}{\log_2(t+2)}$

Formulating a Robust Ranking Loss (RoBiRank)

Applying the robust transformation $\rho_2(t)$

Objective Function

$$L(\mathbf{w}) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}_x} r_{xy} \cdot \rho_2 \left(\sum_{y' \in \mathcal{Y}_x, y' \neq y} \sigma\left(f(x,y) - f(x,y')\right) \right)$$

where $\rho_2(t) = 1 - \frac{1}{\log_2(t+2)}$

Directly related to DCG (evaluation metric for ranking).

- minimizing $L(\mathbf{w})$ $\Longrightarrow$ maximizing DCG

Formulating a Robust Ranking Loss (RoBiRank)

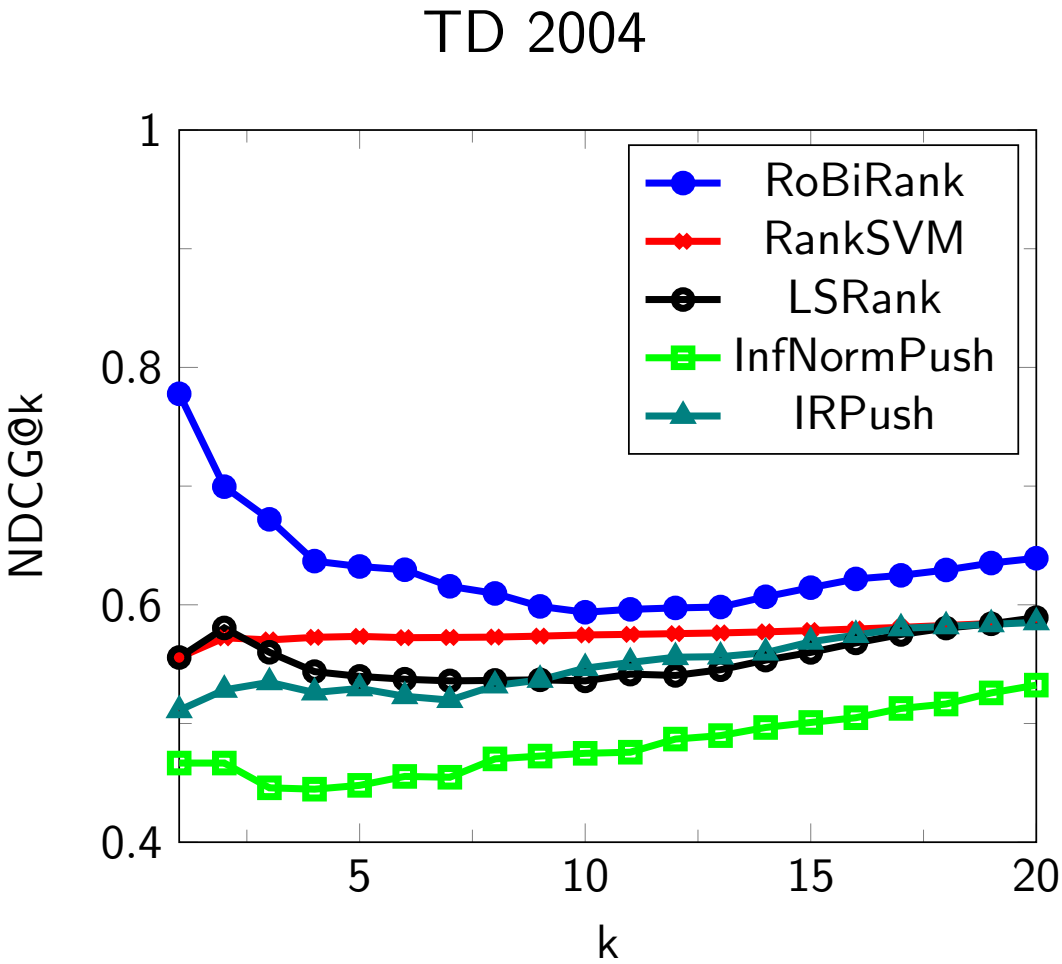
In practice, ρ_1 is easier to optimize,

Objective Function

$$L(\mathbf{w}) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}_x} r_{xy} \cdot \rho_1 \left(\sum_{y' \in \mathcal{Y}_x, y' \neq y} \sigma \left(f(x, y) - f(x, y') \right) \right)$$

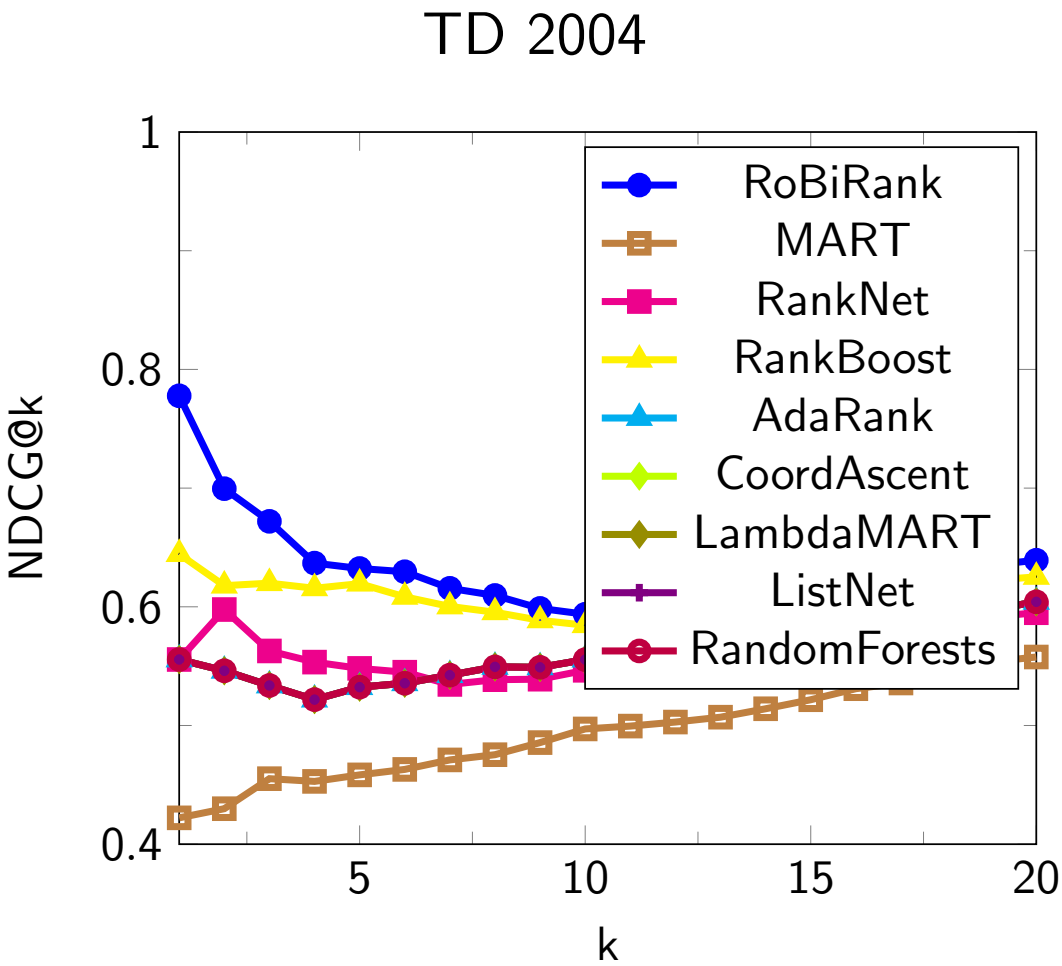
where, $\rho_1(t) = \log_2(t + 1)$

Experiments: Single Machine



RoBiRank shows good accuracy (NDCG) at the top of the ranking list

Experiments: Single Machine



RoBiRank shows good accuracy (NDCG) at the top of the ranking list

How to make RoBiRank Hybrid-Parallel?

Consider RoBiRank with the implicit case (no features, no explicit labels).

Implicit Case

$$\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}_x} r_{xy} \cdot \rho_1 \underbrace{\left(\sum_{y' \in \mathcal{Y}_x, y' \neq y} \sigma\left(\langle U_x, V_y \rangle - \langle U_x, V_{y'} \rangle \right) \right)}_{\text{expensive}}$$

How can we avoid this?

Linearization

Upper Bound for ρ_1 [Bouchard07]

$$\rho_1(t) = \log_2(t+1) \leq \log_2 \xi + \frac{\xi \cdot (t+1) - 1}{\log_2}$$

The bound holds for any $\xi > 0$ and is tight when $\xi = \frac{1}{t+1}$

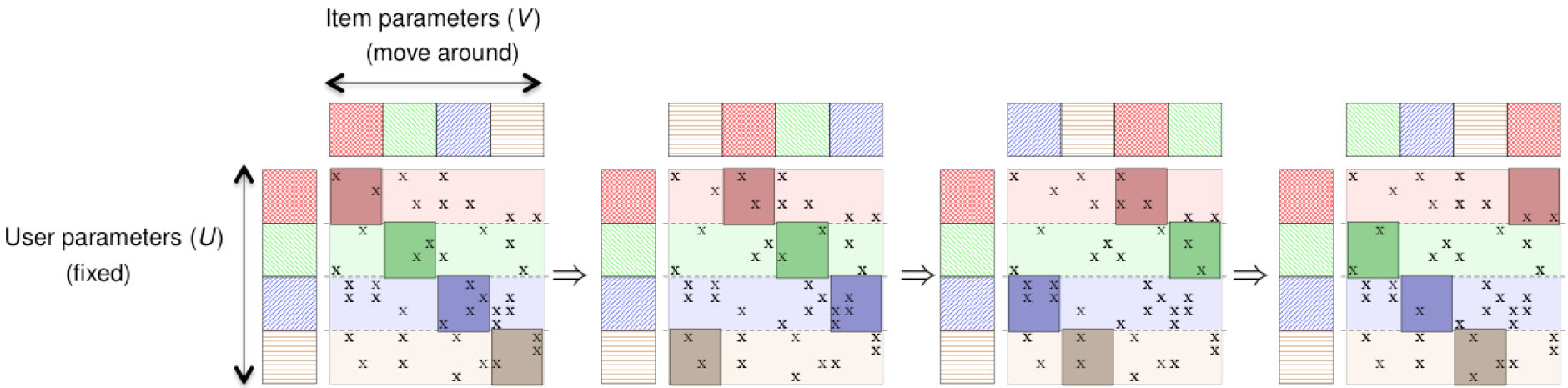
Linearization

We can linearize the objective function using this upper bound,

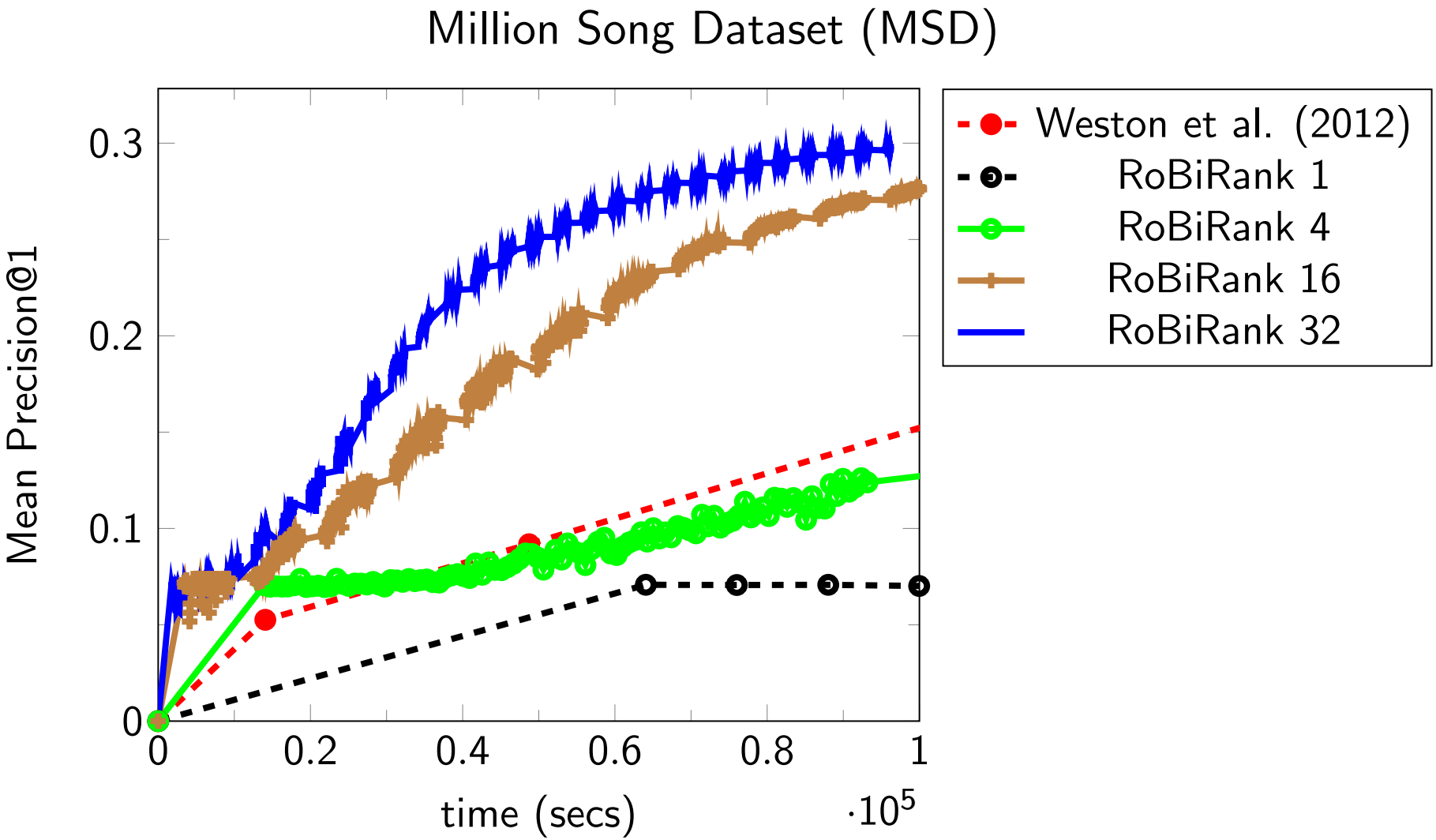
$$\sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}_x} r_{xy} \cdot \left[-\log_2 \xi_{xy} + \frac{\xi_{xy} \cdot \left(\sum_{y' \neq y} \sigma\left(\langle U_x, V_y \rangle - \langle U_x, V_{y'} \rangle \right) + 1 \right) - 1}{\log 2} \right]$$

- Can obtain stochastic gradient updates for U_x, V_y and ξ_{xy}
- ξ_{xy} capture the importance of xy-pair in optimization
- ξ_{xy} also has a closed form solution

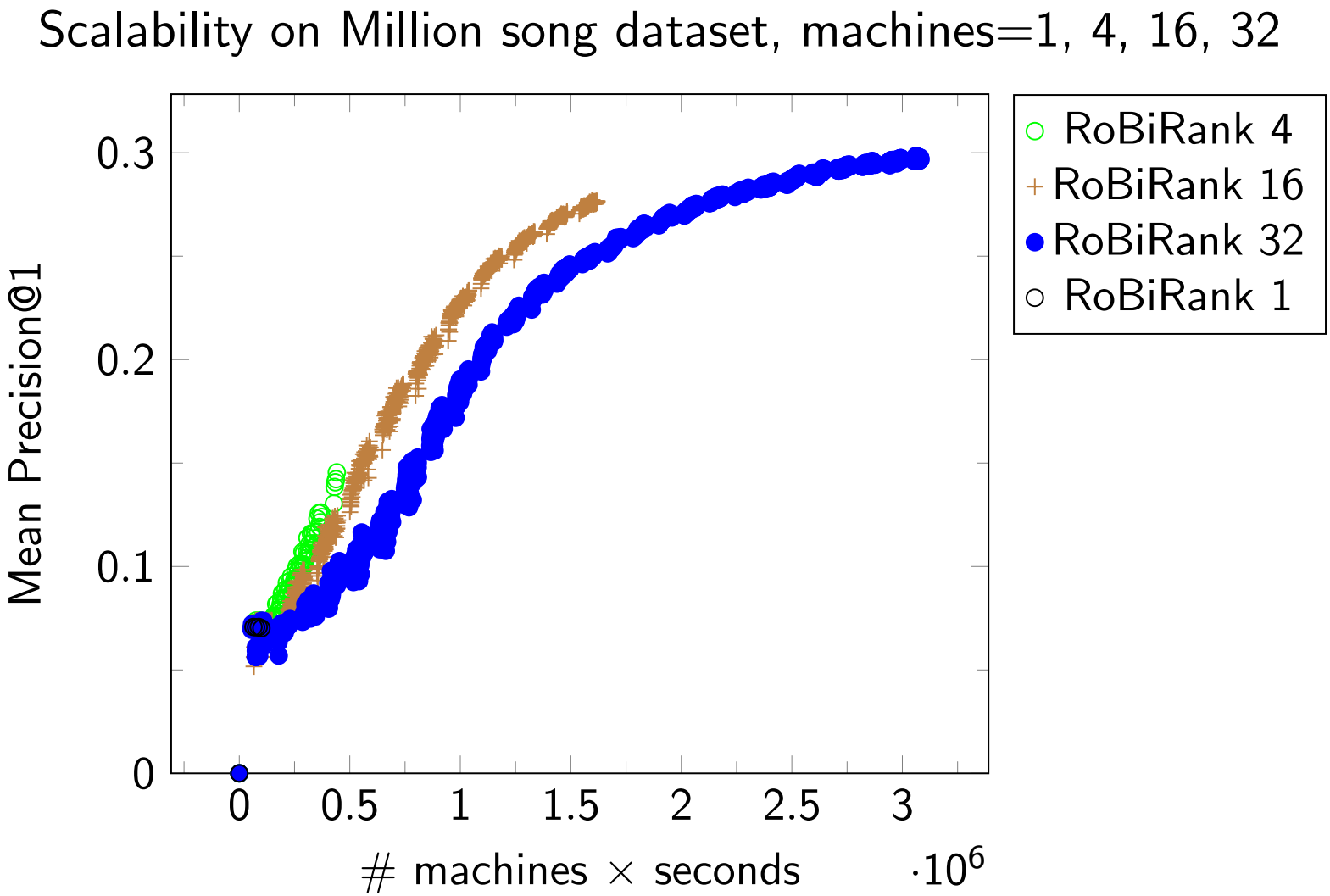
Distributing Computation



Experiments: Multi Machine



Experiments: Multi Machine



Outline

1. Introduction
2. Distributed Parameter Estimation
3. Thesis Roadmap
   - Multinomial Logistic Regression (KDD 2019)
   - Mixture Models (AISTATS 2019)
   - Latent Collaborative Retrieval (NIPS 2014)
4. Conclusion and Future Directions
5. Appendix

Conclusion and Key Takeaways

- Data and Model grow hand in hand.

Conclusion and Key Takeaways

- Data and Model grow hand in hand.
- Challenges in Parameter Estimation.

Conclusion and Key Takeaways

- Data and Model grow hand in hand.
- Challenges in Parameter Estimation.
- We proposed:

Conclusion and Key Takeaways

- Data and Model grow hand in hand.
- Challenges in Parameter Estimation.
- We proposed:
  - **Hybrid-Parallel** formulations

Conclusion and Key Takeaways

- Data and Model grow hand in hand.
- Challenges in Parameter Estimation.
- We proposed:
  - **Hybrid-Parallel** formulations
  - **Distributed, Asynchronous** Algorithms

Conclusion and Key Takeaways

- Data and Model grow hand in hand.
- Challenges in Parameter Estimation.
- We proposed:
  - **Hybrid-Parallel** formulations
  - **Distributed, Asynchronous** Algorithms
- Case-studies:
  - Multinomial Logistic Regression
  - Mixture Models
  - Latent Collaborative Retrieval

Future Work: Other Hybrid-Parallel models

- Infinite Mixture Models
  - DP Mixture Model [BleiJordan2006]
  - Pittman-Yor Mixture Model [Dubey et al., 2014]

Future Work: Other Hybrid-Parallel models

- Infinite Mixture Models
  - DP Mixture Model [BleiJordan2006]
  - Pittman-Yor Mixture Model [Dubey et al., 2014]
- Stochastic Mixed-Membership Block Models [Airoldi et al., 2008]

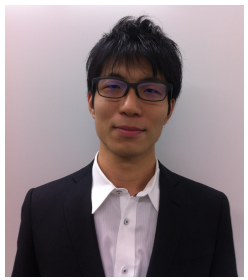
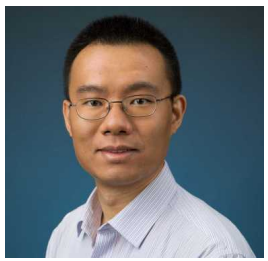
Future Work: Other Hybrid-Parallel models

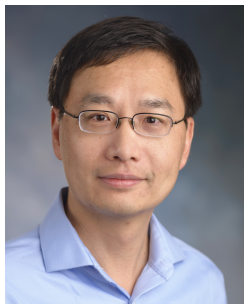
- Infinite Mixture Models
  - DP Mixture Model [BleiJordan2006]
  - Pittman-Yor Mixture Model [Dubey et al., 2014]
- Stochastic Mixed-Membership Block Models [Airoldi et al., 2008]
- Deep Neural Networks [Wang et al., 2014]

Acknowledgements

Thanks to all my collaborators.



References

- **Parameswaran Raman**, Sriram Srinivasan, Shin Matsushima, Xinhua Zhang, Hyokun Yun, S.V.N. Vishwanathan. *Scaling Multinomial Logistic Regression via Hybrid Parallelism*. KDD 2019.
- **Parameswaran Raman***, Jiong Zhang*, Shihao Ji, Hsiang-Fu Yu, S.V.N. Vishwanathan, Inderjit S. Dhillon. *Extreme Stochastic Variational Inference: Distributed and Asynchronous*. AISTATS 2019.
- Hyokun Yun, **Parameswaran Raman**, S.V.N. Vishwanathan. *Ranking via Robust Binary Classification*. NIPS 2014.
- Source Code:
  - DS-MLR: `https://bitbucket.org/params/dsmlr`
  - ESVI: `https://bitbucket.org/params/dmixmodels`
  - RoBiRank: `https://bitbucket.org/d_ijk_stra/robirank`

Outline

1. Introduction
2. Distributed Parameter Estimation
3. Thesis Roadmap
   - Multinomial Logistic Regression (KDD 2019)
   - Mixture Models (AISTATS 2019)
   - Latent Collaborative Retrieval (NIPS 2014)
4. Conclusion and Future Directions
5. Appendix

Polynomial Regression

$$f(\mathbf{x}_i) = w_0 + \sum_{j=1}^{D} w_j \, x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} w_{jj'} \, x_{ij} \, x_{ij'}$$

- $\mathbf{x}_i \in \mathbb{R}^D$ is an example from the dataset $\mathbf{X} \in \mathbb{R}^{N \times D}$
- $w_0 \in \mathbb{R}, \quad \mathbf{w} \in \mathbb{R}^D, \quad \mathbf{W} \in \mathbb{R}^{D \times D}$ are parameters of the model

Polynomial Regression

$$f\left(\mathbf{x}_i\right) = w_0 + \sum_{j=1}^{D} w_j \; x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} w_{jj'} \; x_{ij} \; x_{ij'}$$

- Dense parameterization is not suited for sparse data
- Computationally expensive - $\mathcal{O}\left(N \times D^2\right)$

Factorization Machines

$$f\left(\mathbf{x}_i\right) = w_0 + \sum_{j=1}^{D} w_j \; x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} \left\langle \mathbf{v}_j, \mathbf{v}_{j'} \right\rangle \; x_{ij} \; x_{ij'}$$

Factorization Machines

$$f\left(\mathbf{x}_i\right) = w_0 + \sum_{j=1}^{D} w_j \; x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} \left\langle \mathbf{v}_j, \mathbf{v}_{j'} \right\rangle \; x_{ij} \; x_{ij'}$$

- Factorized Parameterization using $\mathbf{V}\mathbf{V}^T$ instead of Dense $\mathbf{W}$

Factorization Machines

$$f\left(\mathbf{x}_i\right) = w_0 + \sum_{j=1}^{D} w_j \; x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} \left\langle \mathbf{v}_j, \mathbf{v}_{j'} \right\rangle \; x_{ij} \; x_{ij'}$$

- Factorized Parameterization using $\mathbf{V}\mathbf{V}^T$ instead of Dense $\mathbf{W}$
- Model parameters are $w_0 \in \mathbb{R}, \quad \mathbf{w} \in \mathbb{R}^D, \quad \mathbf{V} \in \mathbb{R}^{D \times K}$

Factorization Machines

$$f(\mathbf{x}_i) = w_0 + \sum_{j=1}^{D} w_j \, x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} \left\langle \mathbf{v}_j, \mathbf{v}_{j'} \right\rangle \, x_{ij} \, x_{ij'}$$

- Factorized Parameterization using $\mathbf{V}\mathbf{V}^T$ instead of Dense $\mathbf{W}$
- Model parameters are $w_0 \in \mathbb{R}, \quad \mathbf{w} \in \mathbb{R}^D, \quad \mathbf{V} \in \mathbb{R}^{D \times K}$
- $\mathbf{v}_j \in \mathbb{R}^K$ denotes latent embedding for the j-th feature

Factorization Machines

$$f(\mathbf{x}_i) = w_0 + \sum_{j=1}^{D} w_j \, x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} \left\langle \mathbf{v}_j, \mathbf{v}_{j'} \right\rangle \, x_{ij} \, x_{ij'}$$

- Factorized Parameterization using $\mathbf{V}\mathbf{V}^T$ instead of Dense $\mathbf{W}$
- Model parameters are $w_0 \in \mathbb{R}, \quad \mathbf{w} \in \mathbb{R}^D, \quad \mathbf{V} \in \mathbb{R}^{D \times K}$
- $\mathbf{v}_j \in \mathbb{R}^K$ denotes latent embedding for the j-th feature
- Computationally much cheaper - $\mathcal{O}(N \times D \times K)$

Factorization Machines

$$
\begin{aligned}
f\left(\mathbf{x}_i\right) &= w_0 + \sum_{j=1}^{D} w_j \; x_{ij} + \sum_{j=1}^{D} \sum_{j'=j+1}^{D} \left\langle \mathbf{v}_j, \mathbf{v}_{j'} \right\rangle \; x_{ij} \; x_{ij'} \\
&= w_0 + \sum_{j=1}^{D} w_j \; x_{ij} + \frac{1}{2} \sum_{k=1}^{K} \left\{ \left(\sum_{d=1}^{D} v_{dk} x_{id} \right)^2 - \sum_{j=1}^{D} v_{jk}^2 x_{ij}^2 \right\}
\end{aligned}
$$

Factorization Machines

$$\mathcal{L}(\mathbf{w}, \mathbf{V}) = \frac{1}{N} \sum_{i=1}^{N} l\left(f\left(\mathbf{x}_i\right), y_i\right) + \frac{\lambda_w}{2} \left(\|\mathbf{w}\|_2^2\right) + \frac{\lambda_v}{2} \left(\|\mathbf{V}\|_2^2\right)$$

- λ_w and λ_v are regularizers for $\mathbf{w}$ and $\mathbf{V}$
- $l(\cdot)$ is loss function depending on the task

Factorization Machines

Gradient Descent updates (First-Order Features)

$$
\begin{aligned}
w_j^{t+1} &\leftarrow w_j^t - \eta \sum_{i=1}^{N} \nabla_{w_j} l_i\left(\mathbf{w}, \mathbf{V}\right) + \lambda_w \; w_j^t \\
&= w_j^t - \eta \sum_{i=1}^{N} G_i^t \cdot \nabla_{w_j} f\left(\mathbf{x}_i\right) + \lambda_w \; w_j^t \\
&= w_j^t - \eta \sum_{i=1}^{N} G_i^t \cdot x_{ij} + \lambda_w \; w_j^t
\end{aligned} \tag{1}
$$

where, multiplier G_i^t is given by,

$$
G_i^t = \begin{cases} f\left(x_i\right) - y_i, & \text{if squared loss (regression)} \\ \frac{-y_i}{1+\exp(y_i \cdot f_i(x_i))}, & \text{if logistic loss (classification)} \end{cases} \tag{2}
$$

Factorization Machines

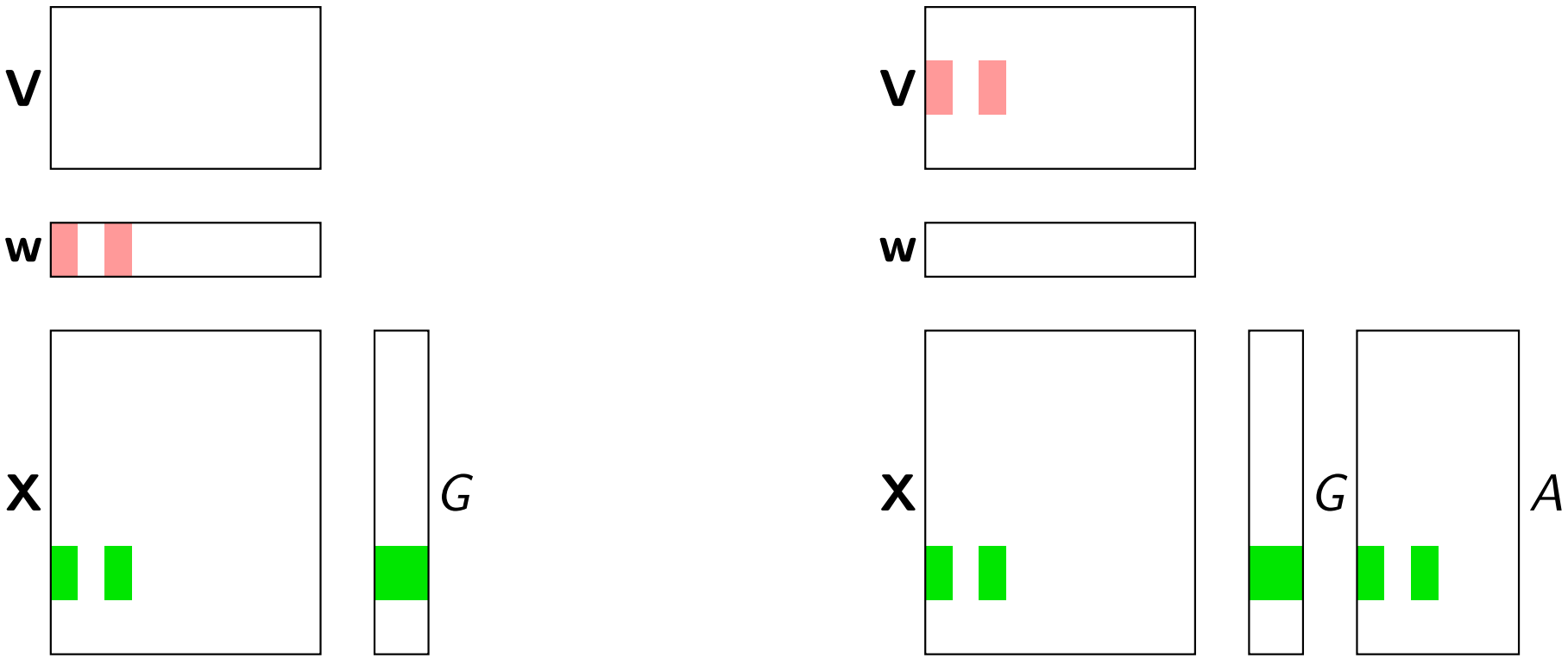
Gradient Descent updates (Second-Order Features)

$$
\begin{aligned}
v_{jk}^{t+1} &\leftarrow v_{jk}^{t} - \eta \sum_{i=1}^{N} \nabla_{v_{jk}} l_i \left(\mathbf{w}, \mathbf{V}\right) + \lambda_v \; v_{jk}^{t} \\
&= v_{jk}^{t} - \eta \sum_{i=1}^{N} G_i^t \cdot \nabla_{v_{jk}} f\left(\mathbf{x}_i\right) + \lambda_v \; v_{jk}^{t} \\
&= v_{jk}^{t} - \eta \sum_{i=1}^{N} G_i^t \cdot \left\{ x_{ij} \left(\sum_{d=1}^{D} v_{dk}^{t} \cdot x_{id} \right) - v_{jk}^{t} \; x_{ij}^2 \right\} + \lambda_v \; v_{jk}^{t} \qquad (3)
\end{aligned}
$$

where, multiplier G_i^t is same as before, synchronization term $a_{ik} = \sum_{d=1}^{D} v_{dk}^{t} \; x_{id}$.

Factorization Machines

Access pattern of parameter updates for w_j and v_{jk}



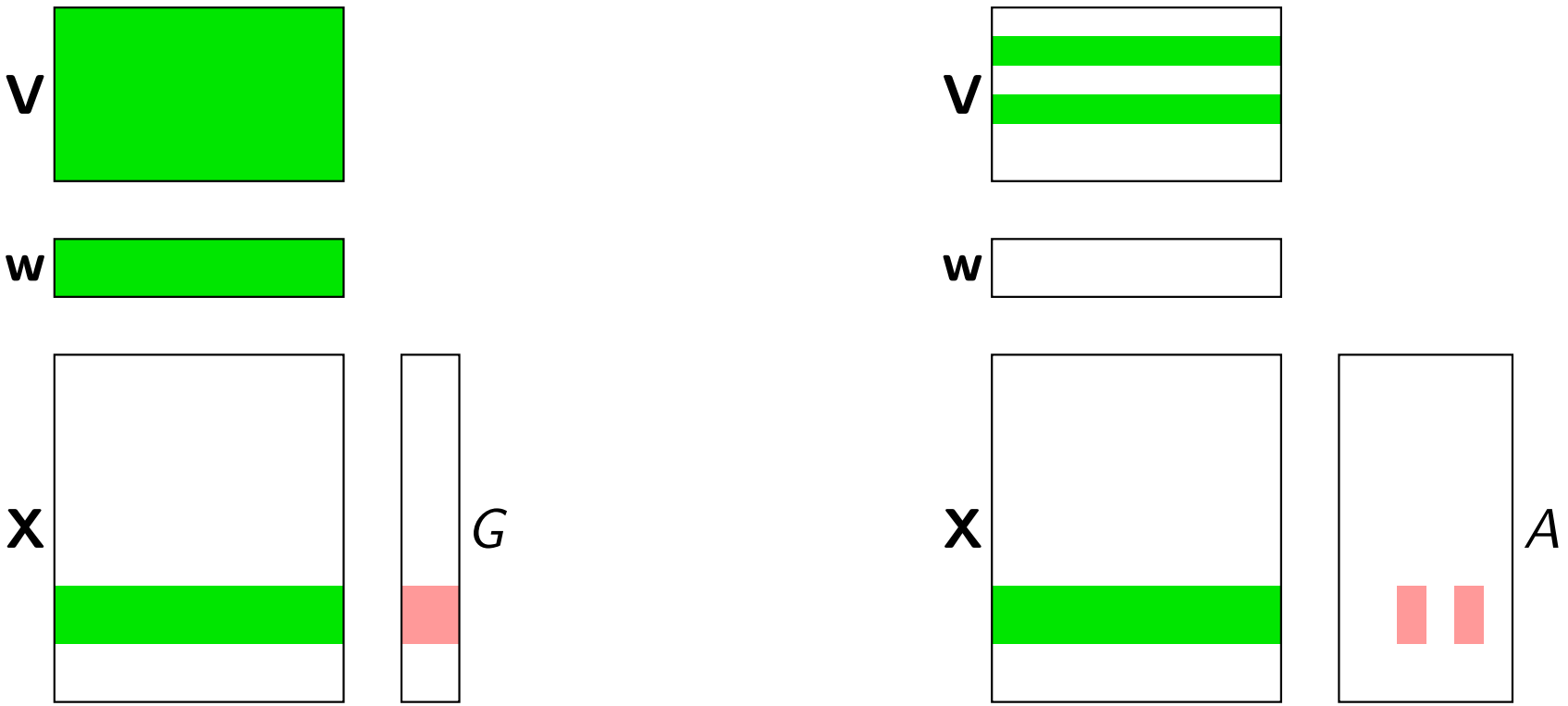
(a) w_j update

(b) v_{jk} update

Figure: Updating w_j requires computing G_i and likewise updating v_{jk} requires computing a_{ik}.

Factorization Machines

Access pattern of parameter updates for w_j and v_{jk}



(a) computing G_i (b) computing a_{ik}

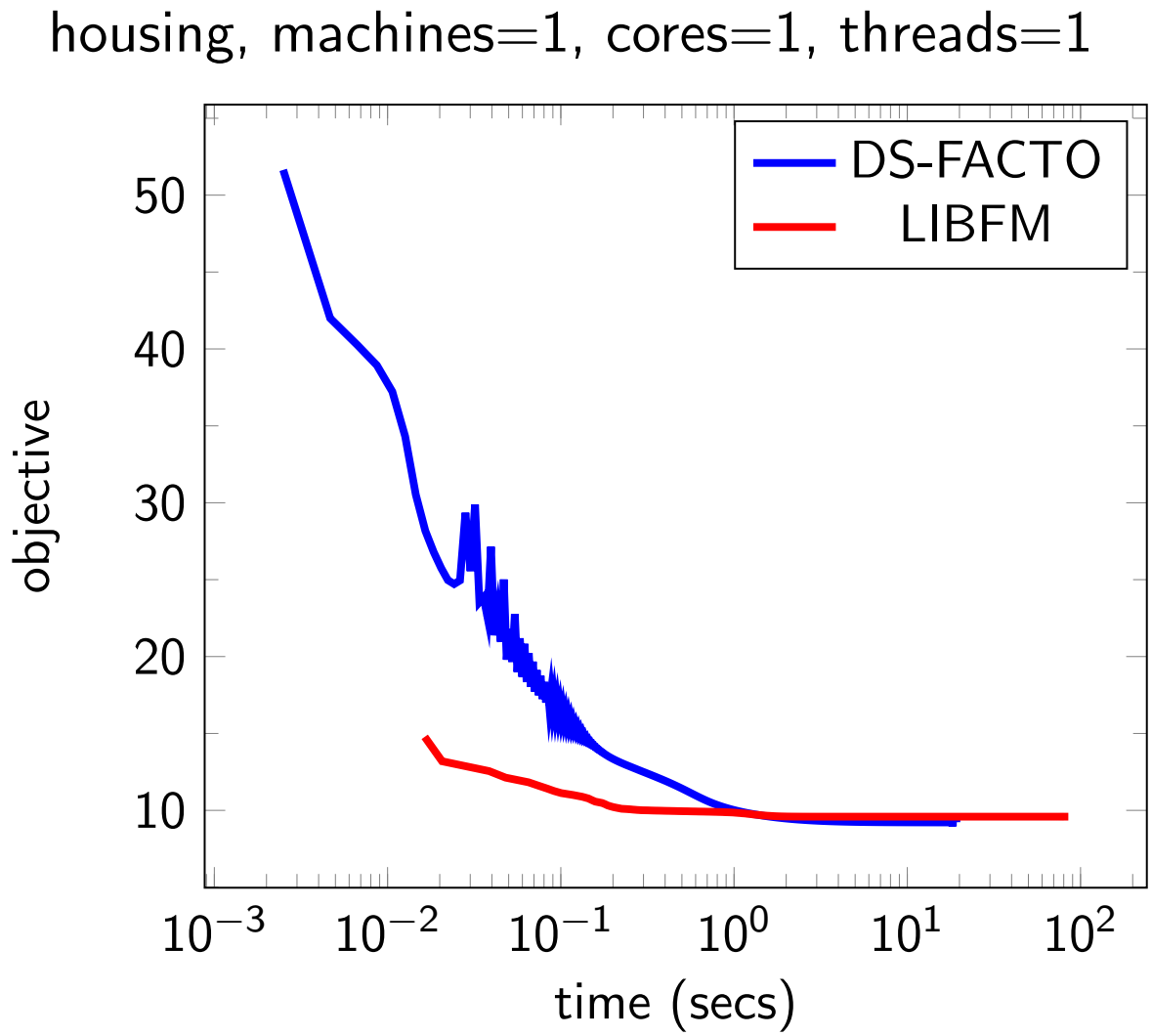
Figure: Computing both G and A requires accessing all the dimensions $j = 1, \ldots, D$. This is the main synchronization bottleneck.

Doubly-Separable Factorization Machines (DS-FACTO)

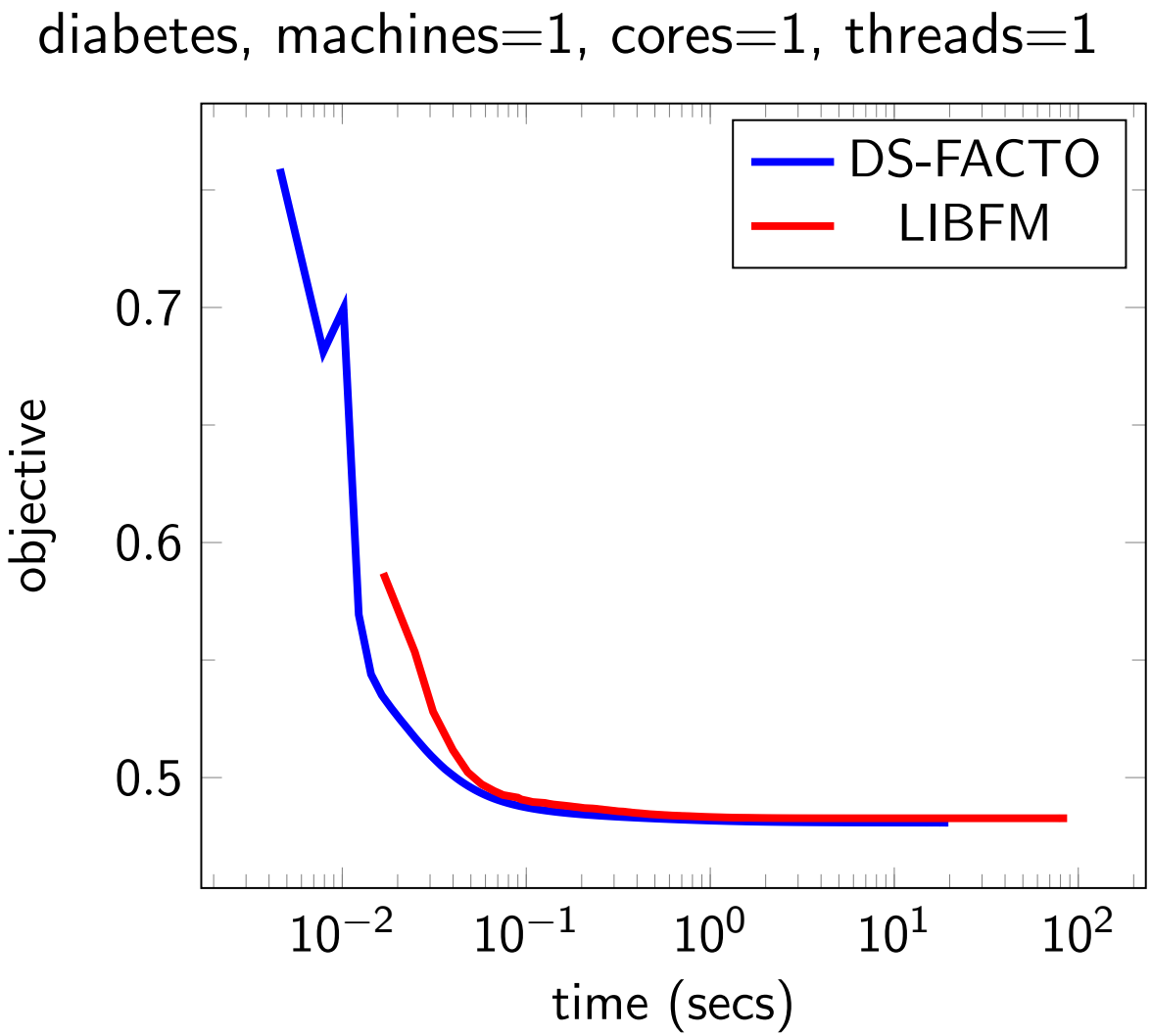
Avoiding bulk-synchronization

- Perform a round of Incremental synchronization after all D updates

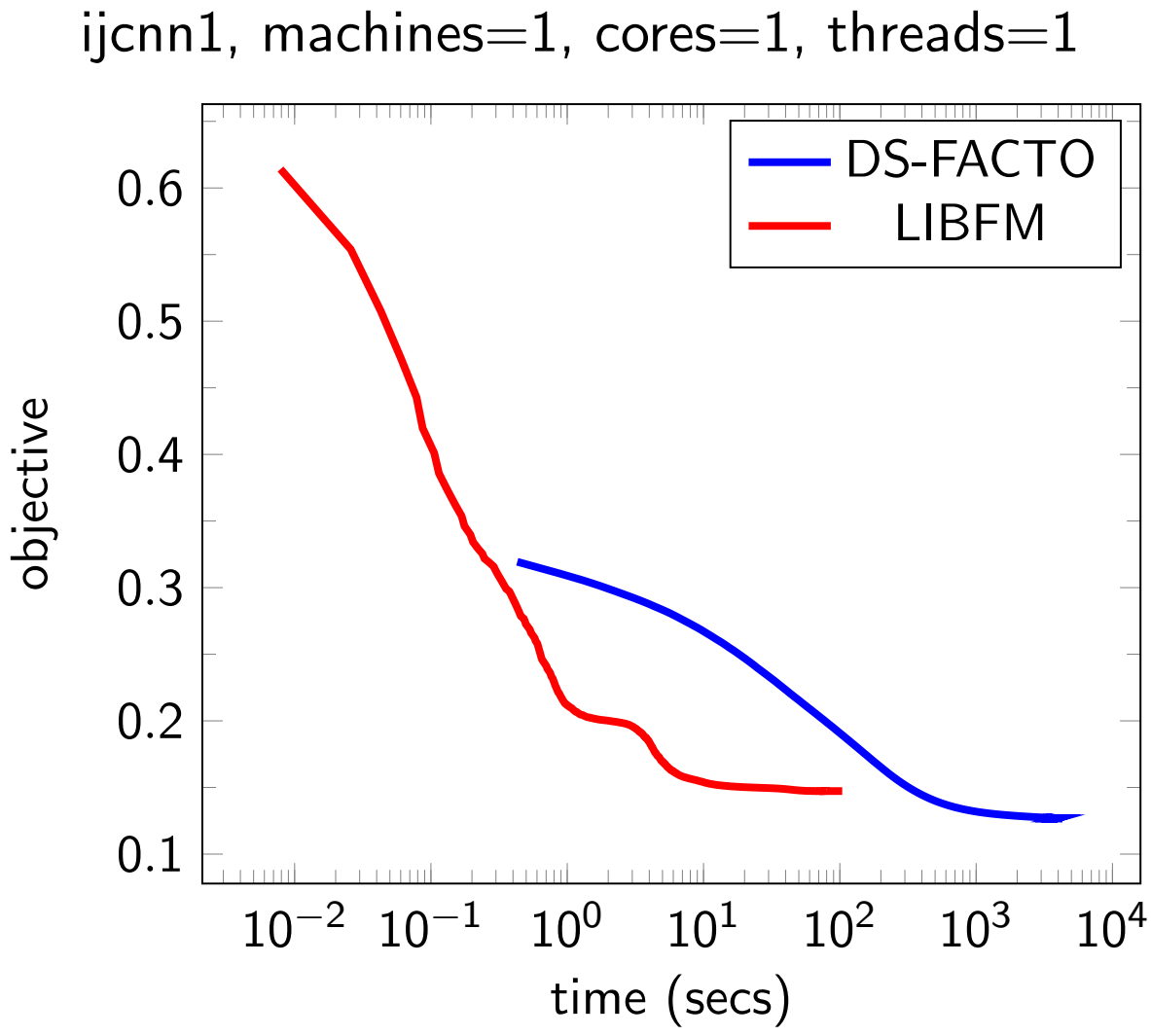
Experiments: Single Machine



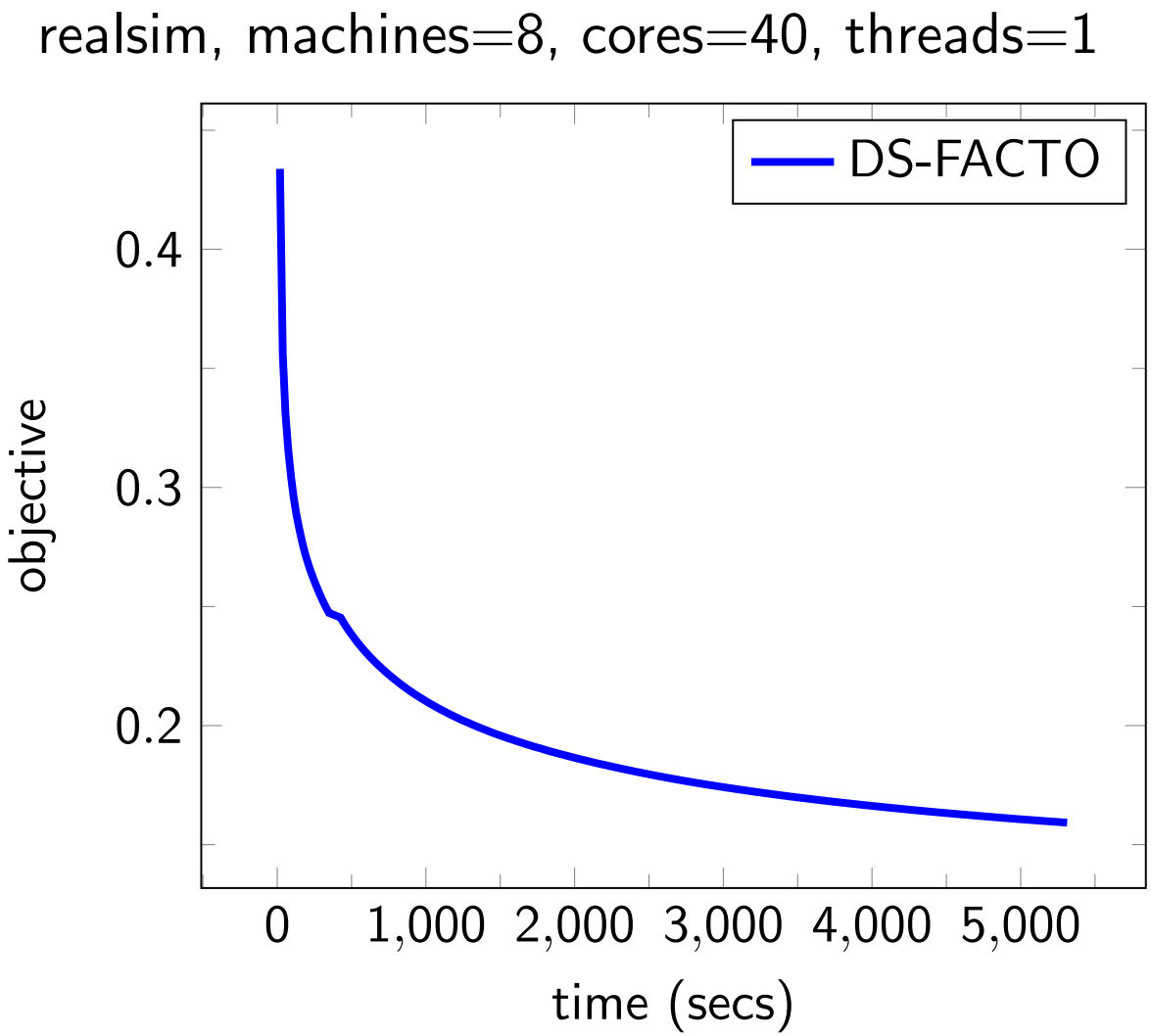
Experiments: Single Machine



Experiments: Single Machine

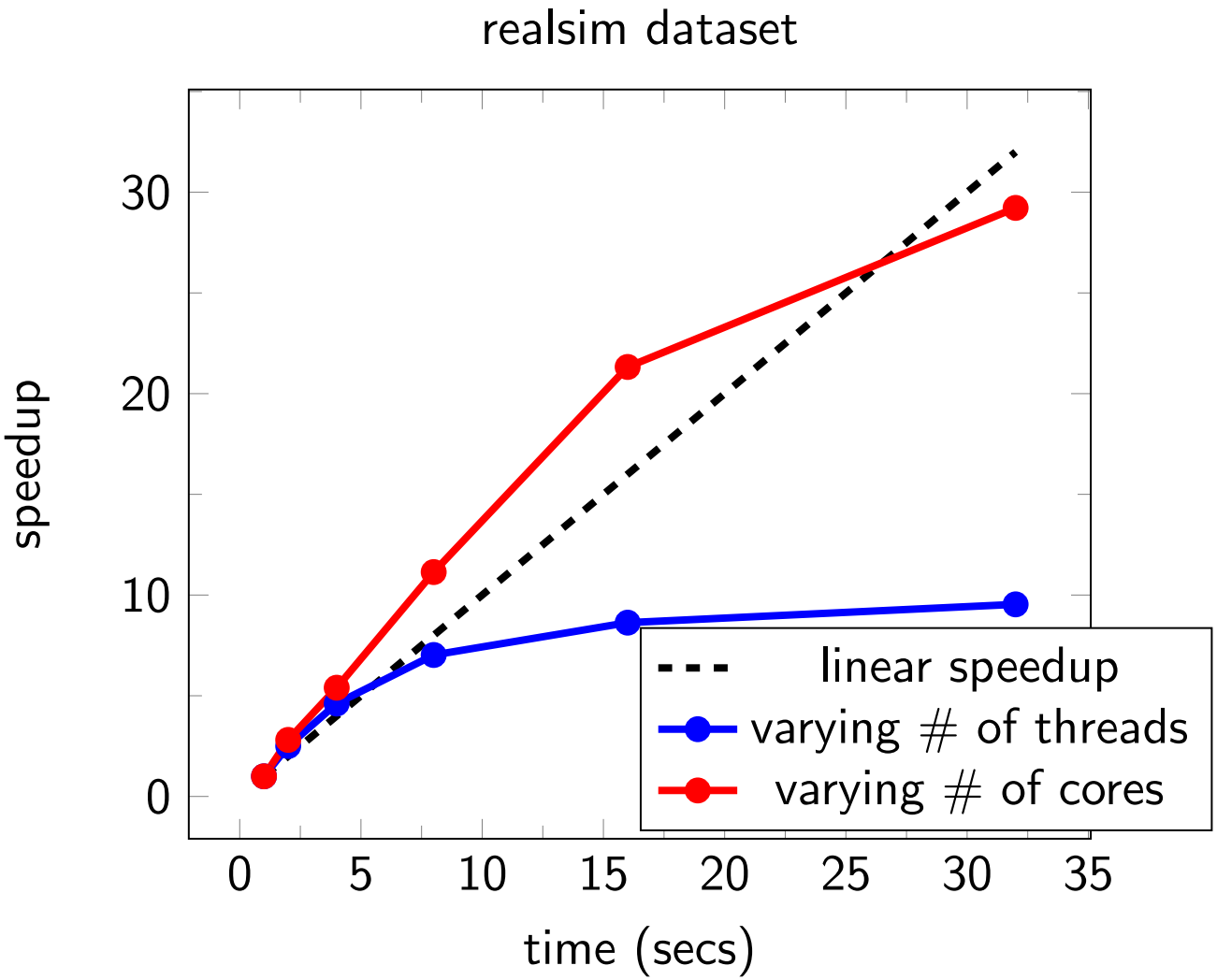


Experiments: Multi Machine



Experiments: Scaling in terms of threads

realsim, Varying cores and threads as 1, 2, 4, 8, 16, 32



Appendix: Alternative Reformulation for DS-MLR

Step 1: Introduce redundant constraints (new parameters A) into the original MLR problem

$$\min_{W,A} \quad L_1(W, A) = \frac{\lambda}{2} \sum_{k=1}^{K} \|\mathbf{w}_k\|^2 - \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \mathbf{w}_k{}^T \mathbf{x}_i - \frac{1}{N} \sum_{i=1}^{N} \log a_i$$

$$\text{s.t.} \quad a_i = \frac{1}{\sum_{k=1}^{K} \exp(\mathbf{w}_k{}^T \mathbf{x}_i)}$$

Appendix: Alternative Reformulation for DS-MLR

Step 2: Turn the problem to unconstrained min-max problem by introducing Lagrange multipliers $\beta_i, \forall i = 1, \ldots, N$

$$\min_{W,A} \max_{\beta} \quad L_2(W, A, \beta) = \frac{\lambda}{2} \sum_{k=1}^{K} \|\mathbf{w}_k\|^2 - \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \mathbf{w}_k{}^T \mathbf{x}_i - \frac{1}{N} \sum_{i=1}^{N} \log a_i + \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \beta_i \, a_i \exp(\mathbf{w}_k{}^T \mathbf{x}_i) - \frac{1}{N} \sum_{i=1}^{N} \beta_i$$

Appendix: Alternative Reformulation for DS-MLR

Step 2: Turn the problem to unconstrained min-max problem by introducing Lagrange multipliers $\beta_i, \forall i = 1, \ldots, N$

$$\min_{W,A} \max_{\beta} \quad L_2(W, A, \beta) = \frac{\lambda}{2} \sum_{k=1}^{K} \|\mathbf{w}_k\|^2 - \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \mathbf{w}_k{}^T \mathbf{x}_i - \frac{1}{N} \sum_{i=1}^{N} \log a_i + \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \beta_i \, a_i \exp(\mathbf{w}_k{}^T \mathbf{x}_i) - \frac{1}{N} \sum_{i=1}^{N} \beta_i$$

Primal Updates for W, A and Dual Update for β (similar in spirit to dual-decomp. methods).

Appendix: Alternative Reformulation for DS-MLR

Step 3: Stare at the updates long-enough!

- When a_i^{t+1} is solved to optimality, it admits an exact closed-form solution given by $a_i^* = \frac{1}{\beta_i \sum_{k=1}^{K} \exp(\mathbf{w}_k{}^T \mathbf{x}_i)}$.
- Dual-ascent update for β_i is no longer needed, since the penalty is always zero if β_i is set to a constant equal to 1.

$$\min_{W,A} \quad L_3(W, A) = \frac{\lambda}{2} \sum_{k=1}^{K} \|\mathbf{w}_k\|^2 - \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \mathbf{w}_k{}^T \mathbf{x}_i - \frac{1}{N} \sum_{i=1}^{N} \log a_i + \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} a_i \exp(\mathbf{w}_k{}^T \mathbf{x}_i) - \frac{1}{N}$$

Appendix: Alternative Reformulation for DS-MLR

Step 4: Simple re-write

Doubly-Separable form

$$\min_{W,A} \sum_{i=1}^{N}\sum_{k=1}^{K}\left(\frac{\lambda\|\mathbf{w}_k\|^2}{2N} - \frac{y_{ik}\mathbf{w}_k{}^T\mathbf{x}_i}{N} - \frac{\log a_i}{NK} + \frac{\exp(\mathbf{w}_k{}^T\mathbf{x}_i + \log a_i)}{N} - \frac{1}{NK}\right)$$