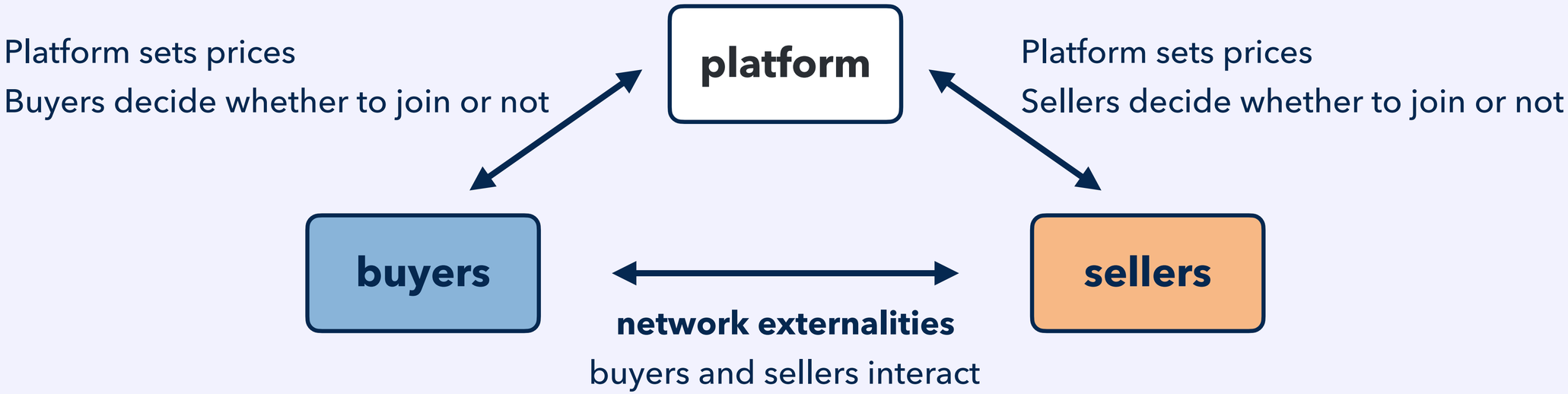
ESIF Economics and AI+ML Meeting

Two-Sided Markets and Restricted Boltzmann Machines

Tetsuya Hoshino Romans Pancs

ITAM

Two-Sided Markets



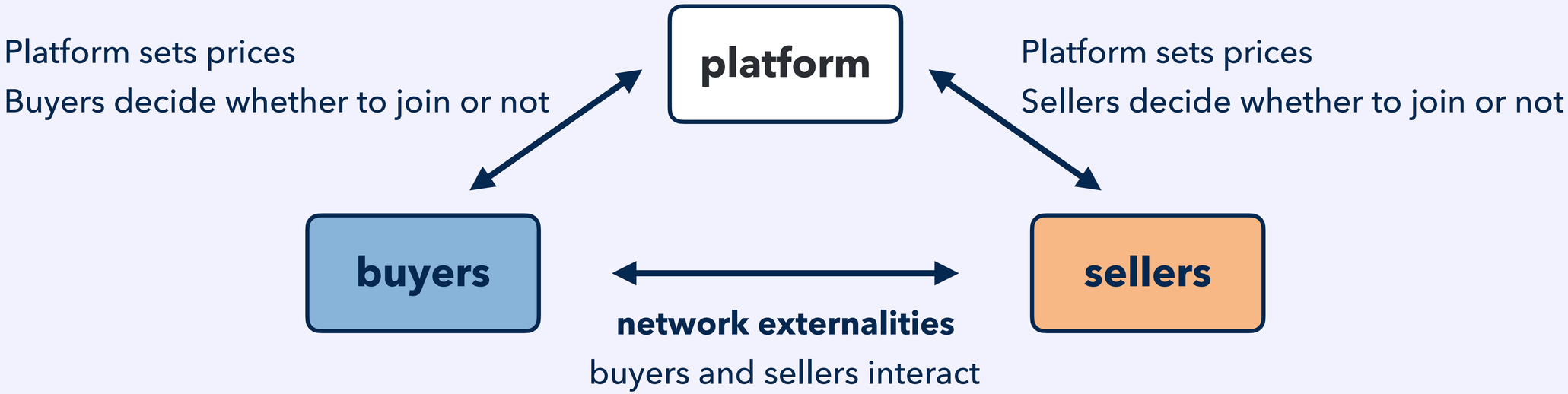
Examples

- credit cards: Amex is a platform on which cardholders and merchants interact
- ride-sharing: Uber is a platform on which riders and drivers interact
- freelancing: Upwork is a platform on which companies and freelancers interact

Literature

- pioneer: Caillaud–Jullien (2003) and Rochet–Tirole (2003)
- survey: Jullien–Pavan–Rysman (2021)

Two-Sided Markets



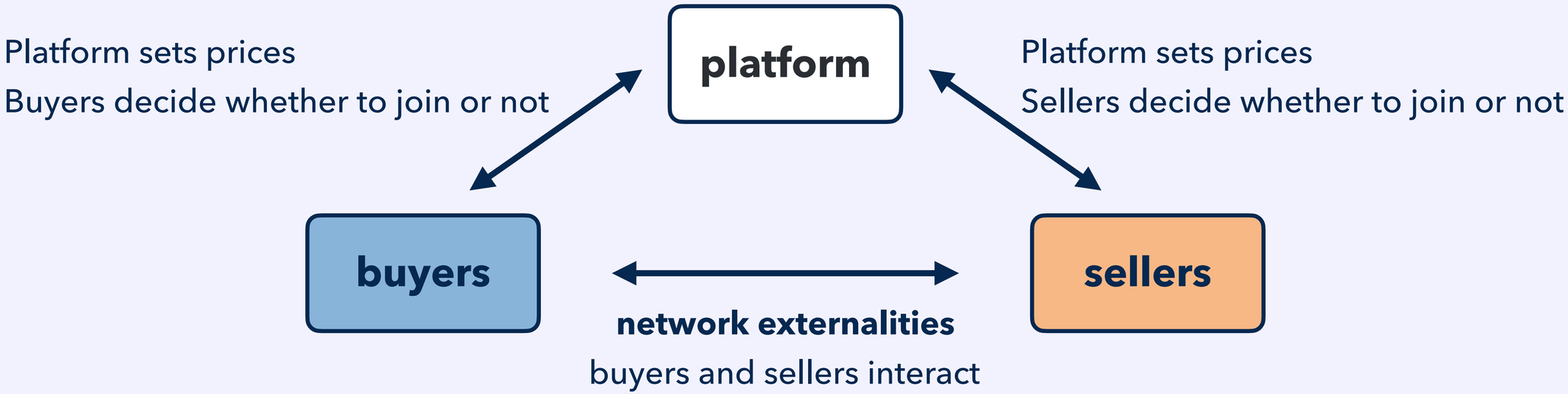
Examples

- credit cards: Amex is a platform on which cardholders and merchants interact
- ride-sharing: Uber is a platform on which riders and drivers interact
- freelancing: Upwork is a platform on which companies and freelancers interact

Literature

- pioneer: Caillaud–Jullien (2003) and Rochet–Tirole (2003)
- survey: Jullien–Pavan–Rysman (2021)

Two-Sided Markets



Examples

- credit cards: Amex is a platform on which cardholders and merchants interact
- ride-sharing: Uber is a platform on which riders and drivers interact
- freelancing: Upwork is a platform on which companies and freelancers interact

Literature

- pioneer: Caillaud–Jullien (2003) and Rochet–Tirole (2003)
- survey: Jullien–Pavan–Rysman (2021)

Two-Sided Markets

Agent i on side s decides whether to join the monopoly platform or not: $x_{si} = 1,0$

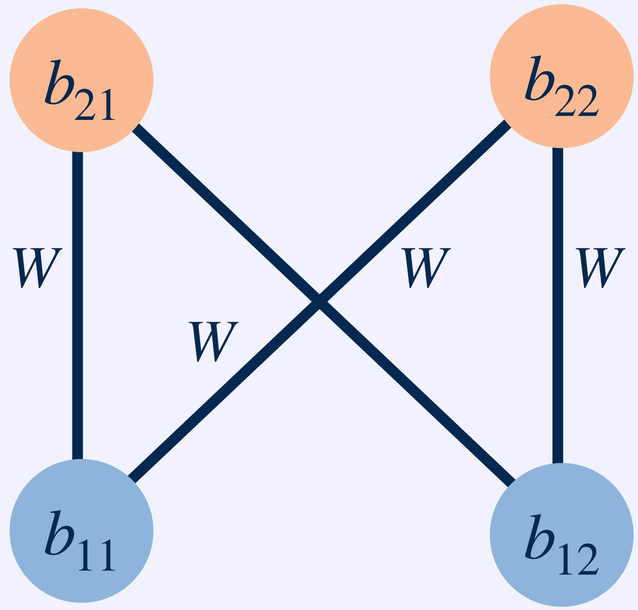
$$\text{side-1 agent's utility} = \left(W \sum_j x_{2j} + b_{1i}\right) x_{1i}$$

$$\text{side-2 agent's utility} = \left(W \sum_i x_{1i} + b_{2j}\right) x_{2j}$$

#participants

Network externalities are **homogeneous**:

- W is the interaction value
- $b_{si} = \beta_{si} - p_{si}$ is the participation benefit/cost
  - β_{si} = the standalone value from joining the platform
  - p_{si} = the price to join the platform



Every participant interacts with all the participants from the opposite side:
two-sided markets (Rochet–Tirole…) ≠ two-sided matching (Gale–Shapley, Roth…)

Two-Sided Markets

Agent i on side s decides whether to join the monopoly platform or not: $x_{si} = 1,0$

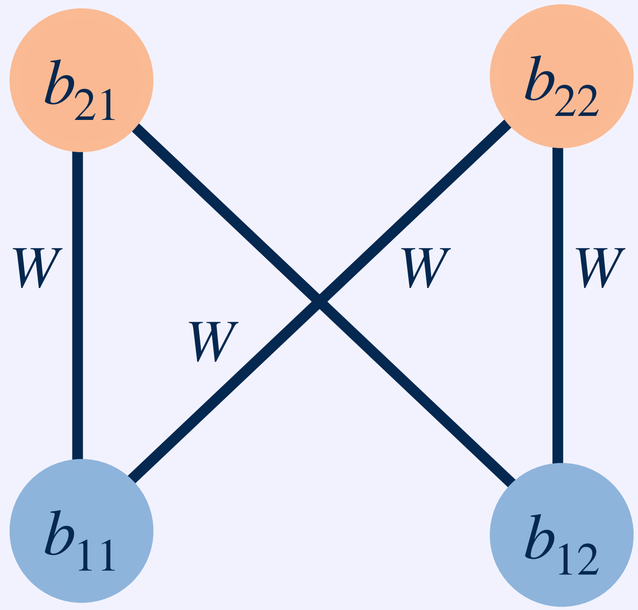
$$\text{side-1 agent's utility} = \left(W \sum_j x_{2j} + b_{1i}\right) x_{1i}$$

$$\text{side-2 agent's utility} = \left(W \sum_i x_{1i} + b_{2j}\right) x_{2j}$$

#participants

Network externalities are **homogeneous**:

- W is the interaction value
- $b_{si} = \beta_{si} - p_{si}$ is the participation benefit/cost
  - β_{si} = the standalone value from joining the platform
  - p_{si} = the price to join the platform



Every participant interacts with all the participants from the opposite side:
two-sided markets (Rochet–Tirole…) ≠ two-sided matching (Gale–Shapley, Roth…)

Two-Sided Markets

Agent i on side s decides whether to join the monopoly platform or not: $x_{si} = 1,0$

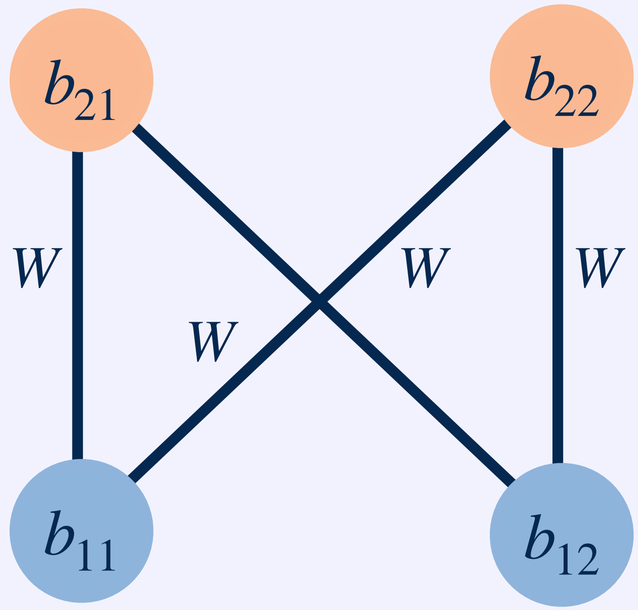
$$\text{side-1 agent's utility} = \left(W \sum_j x_{2j} + b_{1i}\right) x_{1i}$$

$$\text{side-2 agent's utility} = \left(W \sum_i x_{1i} + b_{2j}\right) x_{2j}$$

#participants

Network externalities are **homogeneous**:

- W is the interaction value
- $b_{si} = \beta_{si} - p_{si}$ is the participation benefit/cost
  - β_{si} = the standalone value from joining the platform
  - p_{si} = the price to join the platform



Every participant interacts with all the participants from the opposite side:
two-sided markets (Rochet–Tirole…) ≠ two-sided matching (Gale–Shapley, Roth…)

Two-Sided Markets with Heterogeneous Networks

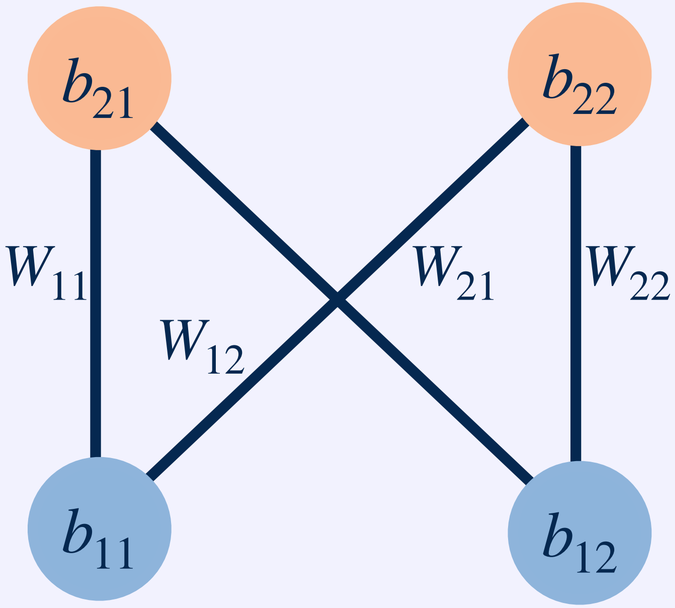
Agent i on side s decides whether to join the monopoly platform or not: $x_{si} = 1,0$

$$\text{side-1 agent's utility} = \left(\sum_j W_{ij}\, x_{2j} + b_{1i} \right) x_{1i}$$

$$\text{side-2 agent's utility} = \left(\sum_i W_{ij}\, x_{1i} + b_{2j} \right) x_{2j}$$

Network externalities are **heterogeneous**:

- W_{ij} is the interaction benefit between i and j
- $b_{si} = \beta_{si} - p_{si}$ is the participation benefit/cost



A **startup platform** faces two challenges:

- to learn or estimate parameters $(\mathbf{W}, \mathbf{b})$
- to set prices $\mathbf{p}$

Two-Sided Markets with Heterogeneous Networks

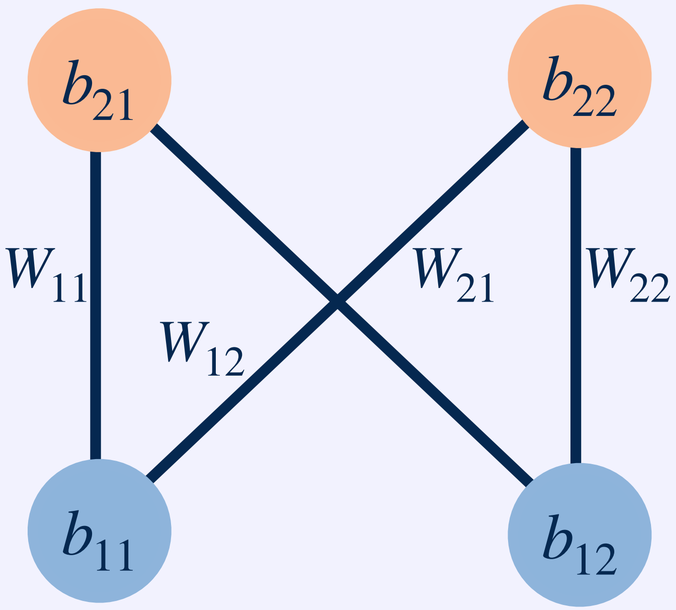
Agent i on side s decides whether to join the monopoly platform or not: $x_{si} = 1,0$

$$\text{side-1 agent's utility} = \left(\sum_j W_{ij}\, x_{2j} + b_{1i} \right) x_{1i}$$

$$\text{side-2 agent's utility} = \left(\sum_i W_{ij}\, x_{1i} + b_{2j} \right) x_{2j}$$

Network externalities are **heterogeneous**:

- W_{ij} is the interaction benefit between i and j
- $b_{si} = \beta_{si} - p_{si}$ is the participation benefit/cost



A **startup platform** faces two challenges:

- to learn or estimate parameters $(\mathbf{W}, \mathbf{b})$
- to set prices $\mathbf{p}$

Two-Sided Markets with Heterogeneous Networks

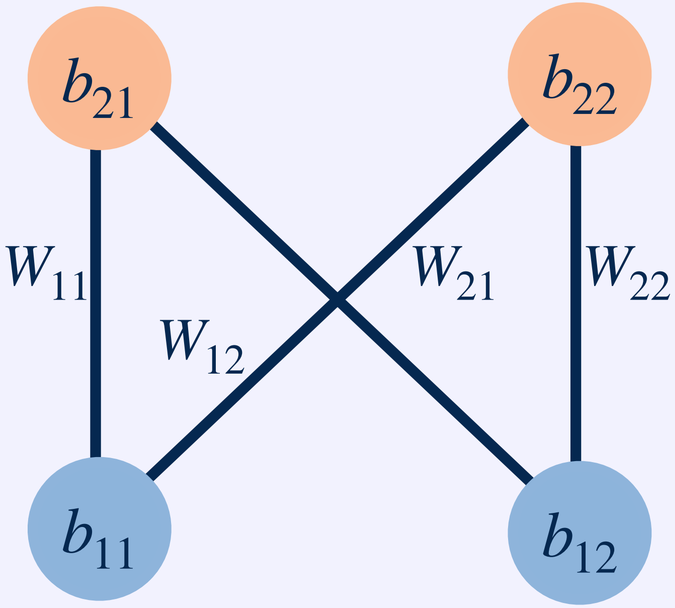
Agent i on side s decides whether to join the monopoly platform or not: $x_{si} = 1,0$

$$\text{side-1 agent's utility} = \left(\sum_j W_{ij}\, x_{2j} + b_{1i} \right) x_{1i}$$

$$\text{side-2 agent's utility} = \left(\sum_i W_{ij}\, x_{1i} + b_{2j} \right) x_{2j}$$

Network externalities are **heterogeneous**:

- W_{ij} is the interaction benefit between i and j
- $b_{si} = \beta_{si} - p_{si}$ is the participation benefit/cost



A **startup platform** faces two challenges:

- to learn or estimate parameters $(\mathbf{W}, \mathbf{b})$
- to set prices $\mathbf{p}$

today's focus

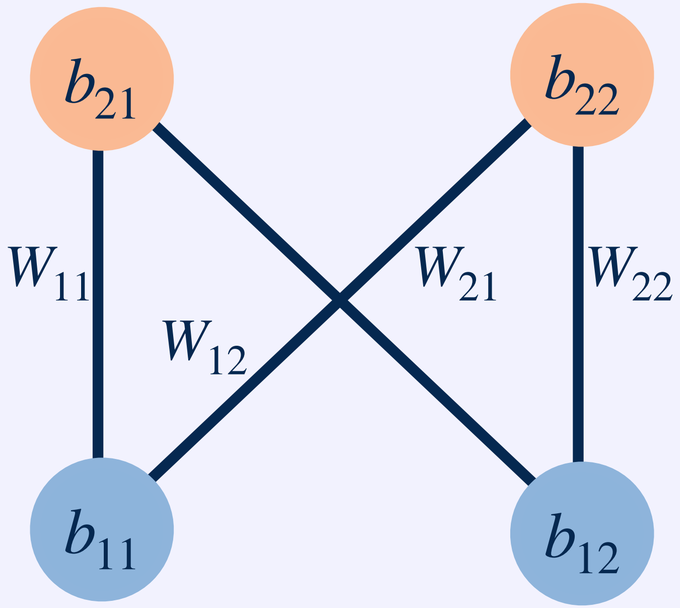
Scalability Problem

The startup's problem suffers from the **scalability** problem

- as #agents increases, estimating parameters $(\mathbf{W}, \mathbf{b})$ becomes exponentially harder

#action profiles $= 2^{\#\text{nodes}}$

Two-Sided Market (2SM)



$\Longleftrightarrow$

Restricted Boltzmann Machine (RBM)

- the RBM is a class of neural networks
- the RBM has a scalable training algorithm, called the **Contrastive Divergence** (**CD**)

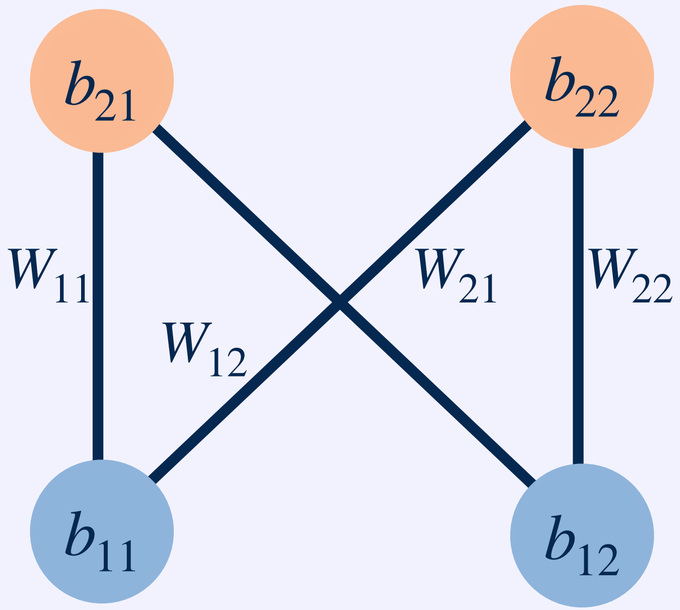
Scalability Problem

The startup's problem suffers from the **scalability** problem

- as #agents increases, estimating parameters $(\mathbf{W}, \mathbf{b})$ becomes exponentially harder

#action profiles $= 2^{\#\text{nodes}}$

Two-Sided Market (2SM)



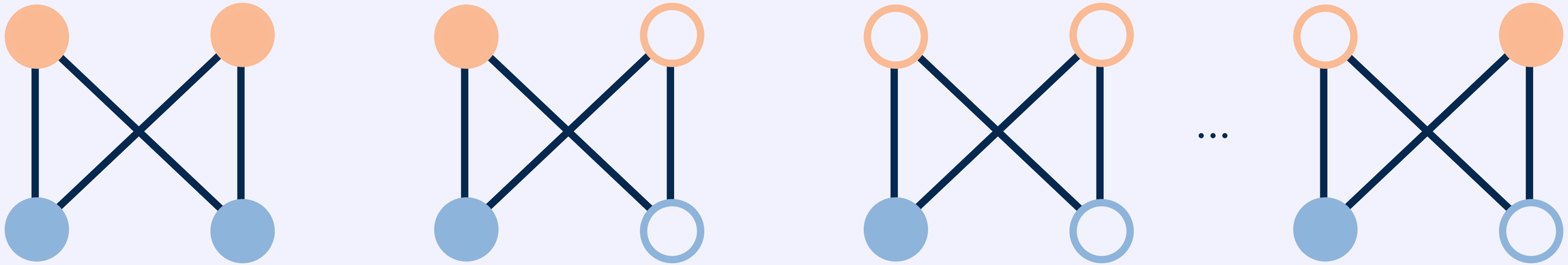
$\Longleftrightarrow$

Restricted Boltzmann Machine (RBM)

- the RBM is a class of neural networks
- the RBM has a scalable training algorithm, called the **Contrastive Divergence** (**CD**)

Participation Data

Assume that the participation data $\{\mathbf{x}(t)\}_{t=1}^{T}$ is generated according to some equilibrium



$\mathbf{x}_2(1) = (1,1)$
$\mathbf{x}_1(1) = (1,1)$

$\mathbf{x}_2(2) = (1,0)$
$\mathbf{x}_1(2) = (1,0)$

$\mathbf{x}_2(3) = (0,0)$
$\mathbf{x}_1(3) = (1,0)$

...

...

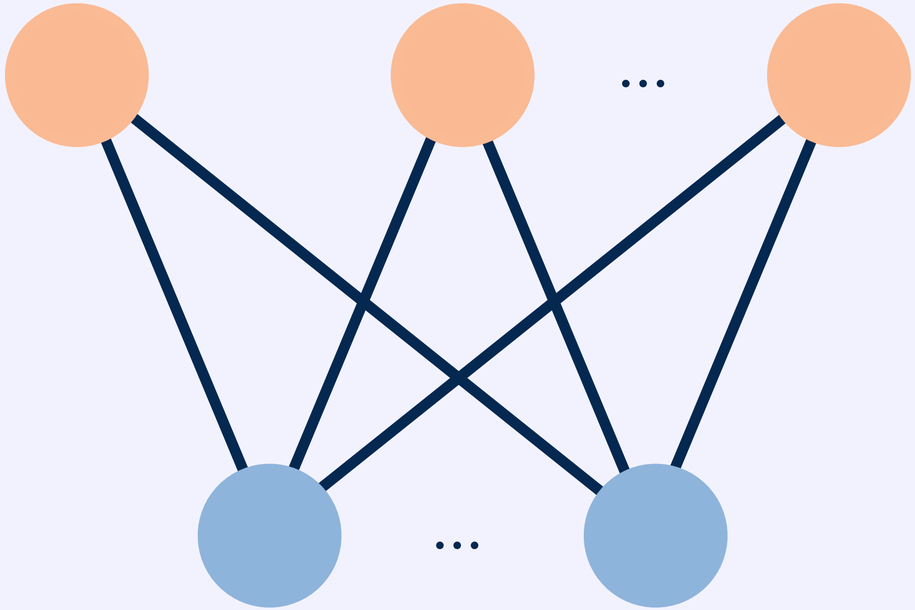
$\mathbf{x}_2(T) = (0,1)$
$\mathbf{x}_1(T) = (1,0)$

Notation

Participations

Parameters

$\mathbf{x}_2 \in \{0,1\}^{n_2}$



$\mathbf{b}_2 \in \mathbb{R}^{n_2}$

$\mathbf{W} \in \mathbb{R}^{n_1 \times n_2}$

$\mathbf{x}_1 \in \{0,1\}^{n_1}$

$\mathbf{b}_1 \in \mathbb{R}^{n_1}$

$\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2)$

$n = n_1 + n_2$ nodes

$\mathbf{b} = (\mathbf{b}_1, \mathbf{b}_2)$

Equilibrium

A **stochastic best-response equilibrium (SBRE)** is the limit of the Markov chain such that

for each step $\tau = 0,1,\ldots$

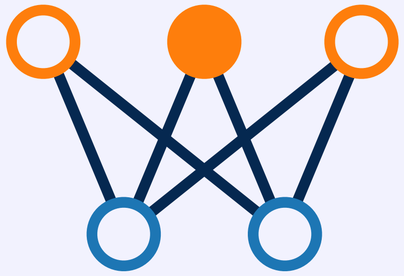
given $\mathbf{x}^{[0]} = (\mathbf{x}_1^{[0]}, \mathbf{x}_2^{[0]})$

- all side-1 agents i best-respond to side-2 actions $\mathbf{x}_2^{[\tau-1]}$ given an i.i.d. logistic noise ε_{1i}

$$x_{1i}^{[\tau]} = \begin{cases} 1 & \text{if } \sum_j W_{ij}\, x_{2j}^{[\tau-1]} + b_{1i} + \varepsilon_{1i} > 0 \\ 0 & \text{otherwise} \end{cases}$$

- all side-2 agents j best-respond to side-1 actions $\mathbf{x}_1^{[\tau]}$ given an i.i.d. logistic noise ε_{2j}

$$x_{2j}^{[\tau]} = \begin{cases} 1 & \text{if } \sum_i W_{ij}\, x_{1i}^{[\tau]} + b_{2j} + \varepsilon_{2j} > 0 \\ 0 & \text{otherwise} \end{cases}$$



Remark: We do not need to update side by side. We can update a randomly chosen agent's action for each step.

Equilibrium

A **stochastic best-response equilibrium (SBRE)** is the limit of the Markov chain such that

for each step $\tau = 0,1,\ldots$

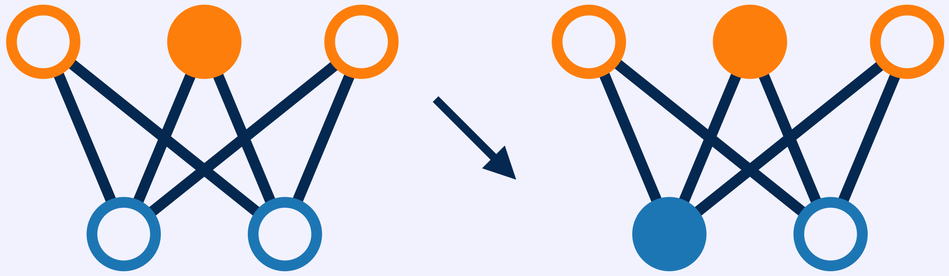
given $\mathbf{x}^{[0]} = (\mathbf{x}_1^{[0]}, \mathbf{x}_2^{[0]})$

- all side-1 agents i best-respond to side-2 actions $\mathbf{x}_2^{[\tau-1]}$ given an i.i.d. logistic noise ε_{1i}

$$x_{1i}^{[\tau]} = \begin{cases} 1 & \text{if } \sum_j W_{ij}\, x_{2j}^{[\tau-1]} + b_{1i} + \varepsilon_{1i} > 0 \\ 0 & \text{otherwise} \end{cases}$$

- all side-2 agents j best-respond to side-1 actions $\mathbf{x}_1^{[\tau]}$ given an i.i.d. logistic noise ε_{2j}

$$x_{2j}^{[\tau]} = \begin{cases} 1 & \text{if } \sum_i W_{ij}\, x_{1i}^{[\tau]} + b_{2j} + \varepsilon_{2j} > 0 \\ 0 & \text{otherwise} \end{cases}$$



Remark: We do not need to update side by side. We can update a randomly chosen agent's action for each step.

Equilibrium

A **stochastic best-response equilibrium (SBRE)** is the limit of the Markov chain such that for each step $\tau = 0,1,\ldots$

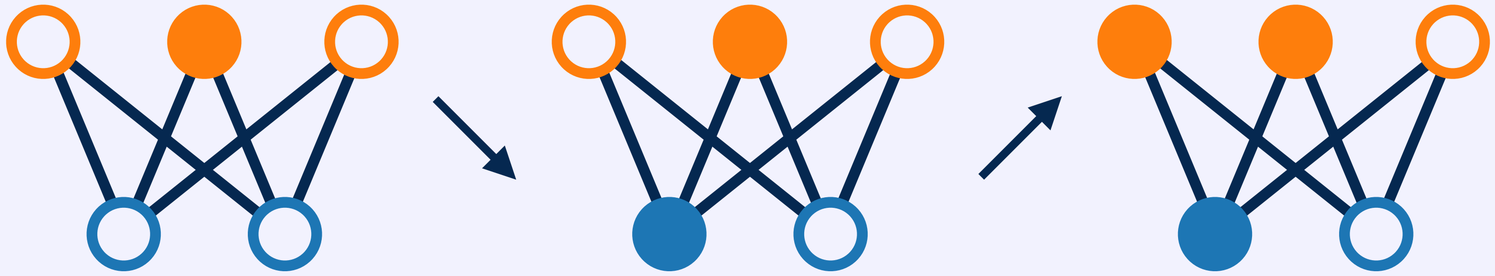
given $\mathbf{x}^{[0]} = (\mathbf{x}_1^{[0]}, \mathbf{x}_2^{[0]})$

- all side-1 agents i best-respond to side-2 actions $\mathbf{x}_2^{[\tau-1]}$ given an i.i.d. logistic noise ε_{1i}

$$x_{1i}^{[\tau]} = \begin{cases} 1 & \text{if } \sum_j W_{ij}\, x_{2j}^{[\tau-1]} + b_{1i} + \varepsilon_{1i} > 0 \\ 0 & \text{otherwise} \end{cases}$$

- all side-2 agents j best-respond to side-1 actions $\mathbf{x}_1^{[\tau]}$ given an i.i.d. logistic noise ε_{2j}

$$x_{2j}^{[\tau]} = \begin{cases} 1 & \text{if } \sum_i W_{ij}\, x_{1i}^{[\tau]} + b_{2j} + \varepsilon_{2j} > 0 \\ 0 & \text{otherwise} \end{cases}$$



Remark: We do not need to update side by side. We can update a randomly chosen agent's action for each step.

Equilibrium

A **stochastic best-response equilibrium (SBRE)** is the limit of the Markov chain such that for each step $\tau = 0,1,\ldots$

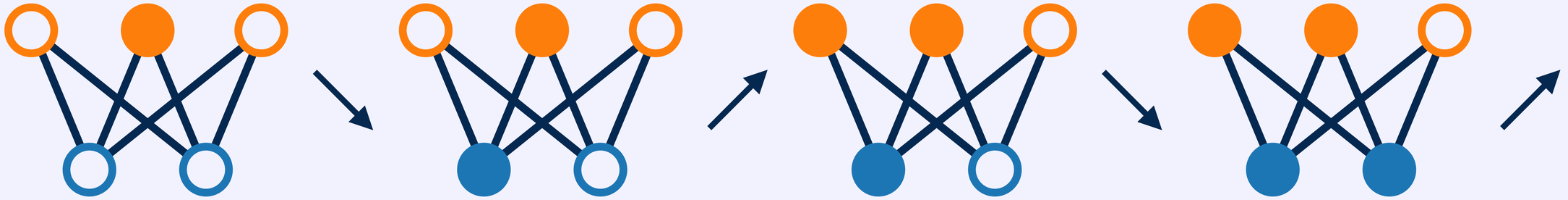
given $\mathbf{x}^{[0]} = (\mathbf{x}_1^{[0]}, \mathbf{x}_2^{[0]})$

- all side-1 agents i best-respond to side-2 actions $\mathbf{x}_2^{[\tau-1]}$ given an i.i.d. logistic noise ε_{1i}

$$x_{1i}^{[\tau]} = \begin{cases} 1 & \text{if } \sum_j W_{ij}\, x_{2j}^{[\tau-1]} + b_{1i} + \varepsilon_{1i} > 0 \\ 0 & \text{otherwise} \end{cases}$$

- all side-2 agents j best-respond to side-1 actions $\mathbf{x}_1^{[\tau]}$ given an i.i.d. logistic noise ε_{2j}

$$x_{2j}^{[\tau]} = \begin{cases} 1 & \text{if } \sum_i W_{ij}\, x_{1i}^{[\tau]} + b_{2j} + \varepsilon_{2j} > 0 \\ 0 & \text{otherwise} \end{cases}$$



Remark: We do not need to update side by side. We can update a randomly chosen agent's action for each step.

Equilibrium

The 2SM model has a unique SBRE:

$$\mathbb{P}\left(\mathbf{x} \mid \mathbf{W}, \mathbf{b}\right) = \frac{\exp\left(\mathbf{b}^\top \mathbf{x} + \mathbf{x}_1^\top \mathbf{W} \mathbf{x}_2\right)}{\sum_{\mathbf{y} \in \{0,1\}^n} \exp\left(\mathbf{b}^\top \mathbf{y} + \mathbf{y}_1^\top \mathbf{W} \mathbf{y}_2\right)}$$

The startup's challenge:

How to estimate $(\mathbf{W}, \mathbf{b})$ from i.i.d data $\{\mathbf{x}(t)\}_{t=1}^{T}$ drawn from the SBRE?

Equilibrium

The 2SM model has a unique SBRE:

$$\mathbb{P}\left(\mathbf{x} \mid \mathbf{W}, \mathbf{b}\right) = \frac{\exp\left(\mathbf{b}^\top \mathbf{x} + \mathbf{x}_1^\top \mathbf{W} \mathbf{x}_2\right)}{\sum_{\mathbf{y} \in \{0,1\}^n} \exp\left(\mathbf{b}^\top \mathbf{y} + \mathbf{y}_1^\top \mathbf{W} \mathbf{y}_2\right)}$$

The startup's challenge:

How to estimate $(\mathbf{W}, \mathbf{b})$ from i.i.d data $\{\mathbf{x}(t)\}_{t=1}^{T}$ drawn from the SBRE?

The maximum likelihood estimator $(\mathbf{W}^{\mathsf{ML}}, \mathbf{b}^{\mathsf{ML}})$ is consistent:

$$\{(\mathbf{W}^{\mathsf{ML}}, \mathbf{b}^{\mathsf{ML}})\} = \arg\max_{(\mathbf{W},\mathbf{b})} \underbrace{\frac{1}{T}\sum_{t=1}^{T} \ln \mathbb{P}\left(\mathbf{x}(t) \mid \mathbf{W}, \mathbf{b}\right)}_{\equiv L(\mathbf{W}, \mathbf{b})}$$

But **computationally intractable** when #nodes n is large

$$\nabla_{\mathbf{W}} L = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}_1(t)\mathbf{x}_2^{\top}(t) - \sum_{\mathbf{y}\in\{0,1\}^n} \mathbb{P}\left(\mathbf{y} \mid \mathbf{W}, \mathbf{b}\right) \mathbf{y}_1\mathbf{y}_2^{\top}$$

$$\nabla_{\mathbf{b}} L = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}(t) - \sum_{\mathbf{y}\in\{0,1\}^n} \mathbb{P}\left(\mathbf{y} \mid \mathbf{W}, \mathbf{b}\right) \mathbf{y}$$

the scalability problem
(combinatorial explosion)

The maximum likelihood estimator $(\mathbf{W}^{\mathsf{ML}}, \mathbf{b}^{\mathsf{ML}})$ is consistent:

$$\{(\mathbf{W}^{\mathsf{ML}}, \mathbf{b}^{\mathsf{ML}})\} = \arg\max_{(\mathbf{W},\mathbf{b})} \underbrace{\frac{1}{T}\sum_{t=1}^{T} \ln \mathbb{P}\left(\mathbf{x}(t) \mid \mathbf{W}, \mathbf{b}\right)}_{\equiv L(\mathbf{W},\mathbf{b})}$$

But **computationally intractable** when #nodes n is large

$$\nabla_{\mathbf{W}} L = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}_1(t)\mathbf{x}_2^{\mathsf{T}}(t) - \sum_{\mathbf{y}\in\{0,1\}^n} \mathbb{P}\left(\mathbf{y} \mid \mathbf{W}, \mathbf{b}\right) \mathbf{y}_1\mathbf{y}_2^{\mathsf{T}}$$

$$\nabla_{\mathbf{b}} L = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}(t) - \sum_{\mathbf{y}\in\{0,1\}^n} \mathbb{P}\left(\mathbf{y} \mid \mathbf{W}, \mathbf{b}\right) \mathbf{y}$$

2^n terms — the scalability problem (combinatorial explosion)

The maximum likelihood estimator $(\mathbf{W}^{\mathsf{ML}}, \mathbf{b}^{\mathsf{ML}})$ is consistent:

$$\{(\mathbf{W}^{\mathsf{ML}}, \mathbf{b}^{\mathsf{ML}})\} = \arg\max_{(\mathbf{W},\mathbf{b})} \underbrace{\frac{1}{T}\sum_{t=1}^{T} \ln \mathbb{P}\left(\mathbf{x}(t) \mid \mathbf{W}, \mathbf{b}\right)}_{\equiv L(\mathbf{W},\mathbf{b})}$$

But **computationally intractable** when #nodes n is large

$$\nabla_{\mathbf{W}} L = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}_1(t)\mathbf{x}_2^{\mathsf{T}}(t) - \sum_{\mathbf{y}\in\{0,1\}^n} \mathbb{P}\left(\mathbf{y} \mid \mathbf{W}, \mathbf{b}\right) \mathbf{y}_1\mathbf{y}_2^{\mathsf{T}}$$

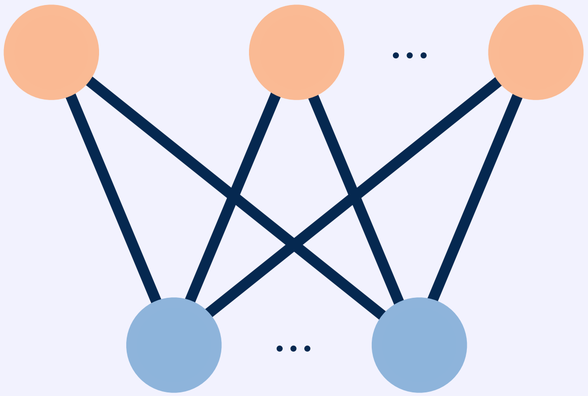
$$\nabla_{\mathbf{b}} L = \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}(t) - \sum_{\mathbf{y}\in\{0,1\}^n} \mathbb{P}\left(\mathbf{y} \mid \mathbf{W}, \mathbf{b}\right) \mathbf{y}$$

2^n terms ← the scalability problem (combinatorial explosion)

Restricted Boltzmann Machines

A RBM is a two-layer neutral network with layer 1 visible and layer 2 hidden

$$\mathbb{P}\left(\mathbf{x} \mid \mathbf{W}, \mathbf{b}\right) = \frac{\exp\left(\mathbf{b}^\top \mathbf{x} + \mathbf{x}_1^\top \mathbf{W} \mathbf{x}_2\right)}{\sum_{\mathbf{y} \in \{0,1\}^n} \exp\left(\mathbf{b}^\top \mathbf{y} + \mathbf{y}_1^\top \mathbf{W} \mathbf{y}_2\right)}$$



Are 2SMs and RBMs just similar?

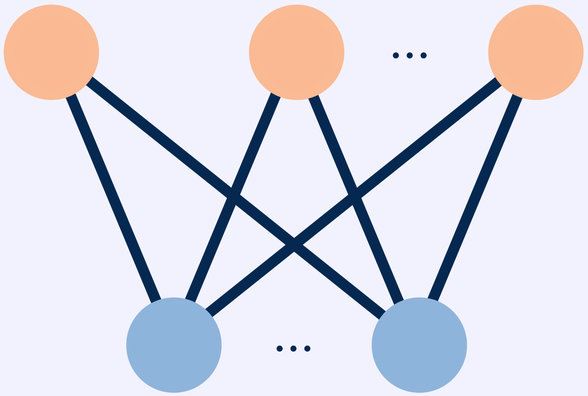
2SM models = **fully visible RBMs** (with both layers visible)

- the 2SM utilities determine the RBM energy $\mathbf{b}^\top \mathbf{x} + \mathbf{x}_1^\top \mathbf{W} \mathbf{x}_2$, and vice versa
- the RBM energy is the potential function of the 2SM model

Restricted Boltzmann Machines

A RBM is a two-layer neutral network with layer 1 visible and layer 2 hidden

$$\mathbb{P}\left(\mathbf{x} \mid \mathbf{W}, \mathbf{b}\right) = \frac{\exp\left(\mathbf{b}^\top \mathbf{x} + \mathbf{x}_1^\top \mathbf{W} \mathbf{x}_2\right)}{\sum_{\mathbf{y} \in \{0,1\}^n} \exp\left(\mathbf{b}^\top \mathbf{y} + \mathbf{y}_1^\top \mathbf{W} \mathbf{y}_2\right)}$$



Are 2SMs and RBMs just similar?

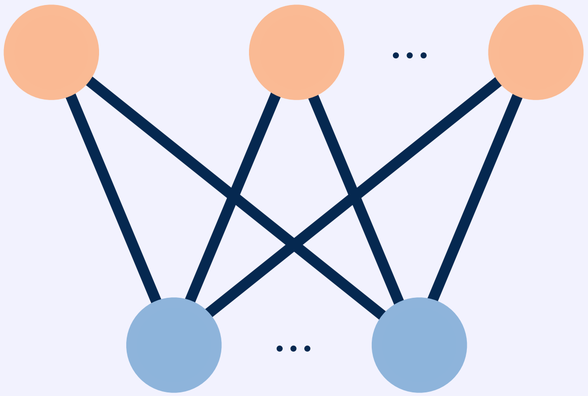
2SM models = **fully visible RBMs** (with both layers visible)

- the 2SM utilities determine the RBM energy $\mathbf{b}^\top \mathbf{x} + \mathbf{x}_1^\top \mathbf{W} \mathbf{x}_2$, and vice versa
- the RBM energy is the potential function of the 2SM model

Restricted Boltzmann Machines

A RBM is a two-layer neutral network with layer 1 visible and layer 2 hidden

$$\mathbb{P}\left(\mathbf{x} \mid \mathbf{W}, \mathbf{b}\right) = \frac{\exp\left(\mathbf{b}^\top \mathbf{x} + \mathbf{x}_1^\top \mathbf{W} \mathbf{x}_2\right)}{\sum_{\mathbf{y} \in \{0,1\}^n} \exp\left(\mathbf{b}^\top \mathbf{y} + \mathbf{y}_1^\top \mathbf{W} \mathbf{y}_2\right)}$$



Are 2SMs and RBMs just similar?

Fully visible RBMs are not used in the machine learning contexts

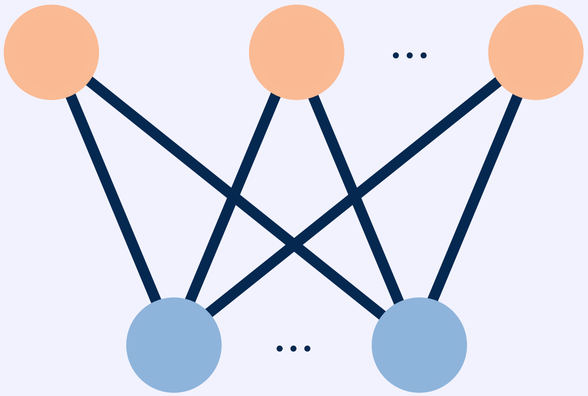
2SM models = **fully visible RBMs** (with both layers visible)

- the 2SM utilities determine the RBM energy $\mathbf{b}^\top \mathbf{x} + \mathbf{x}_1^\top \mathbf{W} \mathbf{x}_2$, and vice versa
- the RBM energy is the potential function of the 2SM model

Restricted Boltzmann Machines

A RBM is a two-layer neutral network with layer 1 visible and layer 2 hidden

$$\mathbb{P}\left(\mathbf{x} \mid \mathbf{W}, \mathbf{b}\right) = \frac{\exp\left(\mathbf{b}^\top \mathbf{x} + \mathbf{x}_1^\top \mathbf{W} \mathbf{x}_2\right)}{\sum_{\mathbf{y} \in \{0,1\}^n} \exp\left(\mathbf{b}^\top \mathbf{y} + \mathbf{y}_1^\top \mathbf{W} \mathbf{y}_2\right)}$$



Are 2SMs and RBMs just similar?

Fully visible RBMs are not used in the machine learning contexts

2SM models = **fully visible RBMs** (with both layers visible)

- the 2SM utilities determine the RBM energy $\mathbf{b}^\top \mathbf{x} + \mathbf{x}_1^\top \mathbf{W} \mathbf{x}_2$, and vice versa
- the RBM energy is the potential function of the 2SM model

Contrastive Divergence

How about approximating the sum using MCMC (stochastic best-response dynamics)?

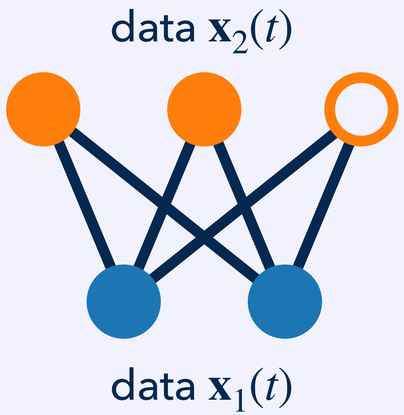
$$\nabla_{\mathbf{b}} L \quad = \quad \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}(t) \; - \sum_{\mathbf{y}\in\{0,1\}^n} \mathbb{P}\left(\mathbf{y} \mid \mathbf{W}, \mathbf{b}\right) \mathbf{y}$$

$\approx$ expectation $\sum_{t=1}^{T} \mathbf{x}^{[k]}(t) \big/ T$

- for all data t

$$\text{data } \mathbf{x}(t) \quad \rightsquigarrow \quad \underbrace{\mathbf{x}_1^{[1]}(t) \rightsquigarrow \mathbf{x}_2^{[1]}(t)}_{\text{step } 1} \quad \rightsquigarrow \cdots \rightsquigarrow \quad \underbrace{\mathbf{x}_1^{[k]}(t) \rightsquigarrow \mathbf{x}_2^{[k]}(t)}_{\text{step } k} \quad \rightsquigarrow$$

MCMC is slow



Contrastive Divergence

How about approximating the sum using MCMC (stochastic best-response dynamics)?

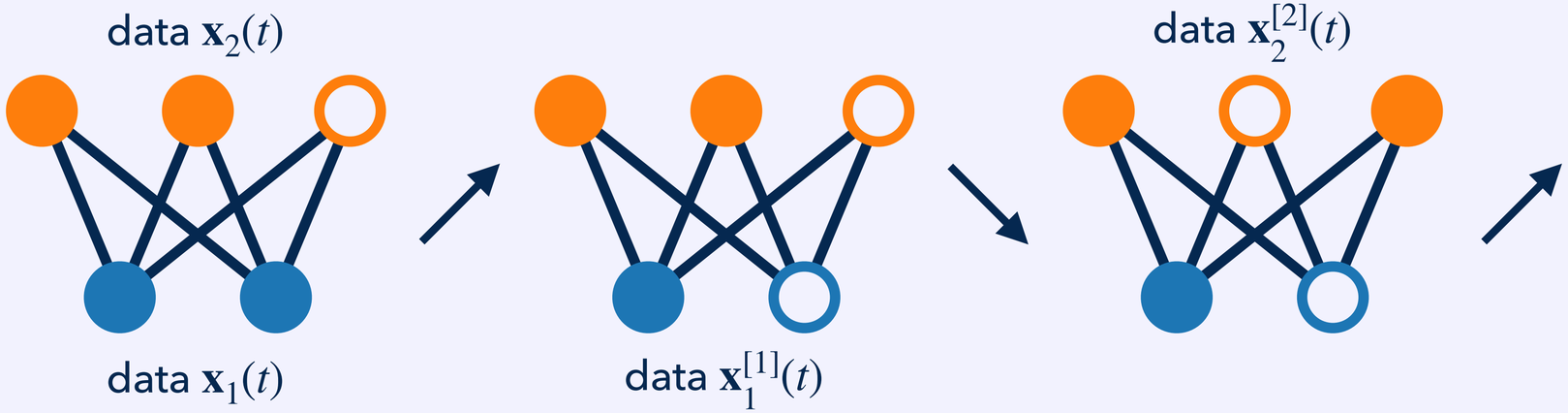
$$\nabla_{\mathbf{b}} L \quad = \quad \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}(t) \; - \sum_{\mathbf{y}\in\{0,1\}^n} \mathbb{P}\left(\mathbf{y} \mid \mathbf{W}, \mathbf{b}\right) \mathbf{y}$$

$\approx$ expectation $\sum_{t=1}^{T} \mathbf{x}^{[k]}(t) \big/ T$

- for all data t

$$\text{data } \mathbf{x}(t) \quad \rightsquigarrow \quad \underbrace{\mathbf{x}_1^{[1]}(t) \rightsquigarrow \mathbf{x}_2^{[1]}(t)}_{\text{step } 1} \quad \rightsquigarrow \cdots \rightsquigarrow \quad \underbrace{\mathbf{x}_1^{[k]}(t) \rightsquigarrow \mathbf{x}_2^{[k]}(t)}_{\text{step } k} \rightsquigarrow$$

MCMC is slow



Contrastive Divergence

How about approximating the sum using MCMC (stochastic best-response dynamics)?

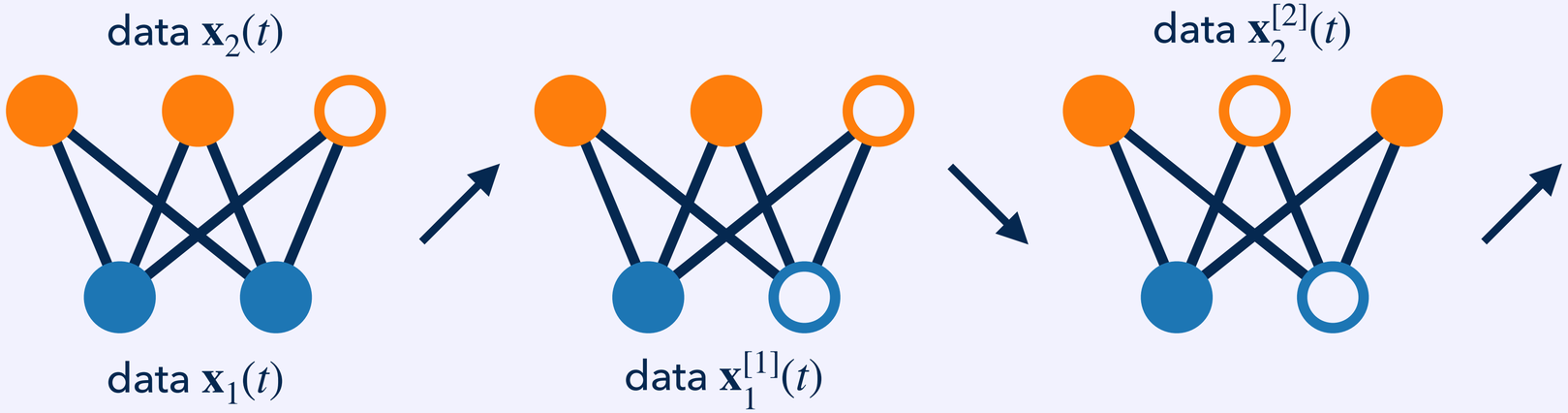
$$\nabla_{\mathbf{b}} L \quad = \quad \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}(t) \; - \sum_{\mathbf{y}\in\{0,1\}^n} \mathbb{P}\left(\mathbf{y} \mid \mathbf{W}, \mathbf{b}\right) \mathbf{y}$$

$\approx$ expectation $\sum_{t=1}^{T} \mathbf{x}^{[k]}(t) \big/ T$

- for all data t

$$\text{data } \mathbf{x}(t) \quad \rightsquigarrow \quad \underbrace{\mathbf{x}_1^{[1]}(t) \rightsquigarrow \mathbf{x}_2^{[1]}(t)}_{\text{step } 1} \quad \rightsquigarrow \cdots \rightsquigarrow \quad \underbrace{\mathbf{x}_1^{[k]}(t) \rightsquigarrow \mathbf{x}_2^{[k]}(t)}_{\text{step } k} \rightsquigarrow$$

MCMC is slow



Contrastive Divergence

How about approximating the sum using MCMC (stochastic best-response dynamics)?

$$\nabla_{\mathbf{b}} L \quad = \quad \frac{1}{T}\sum_{t=1}^{T} \mathbf{x}(t) \; - \sum_{\mathbf{y}\in\{0,1\}^n} \mathbb{P}\left(\mathbf{y} \mid \mathbf{W}, \mathbf{b}\right) \mathbf{y}$$

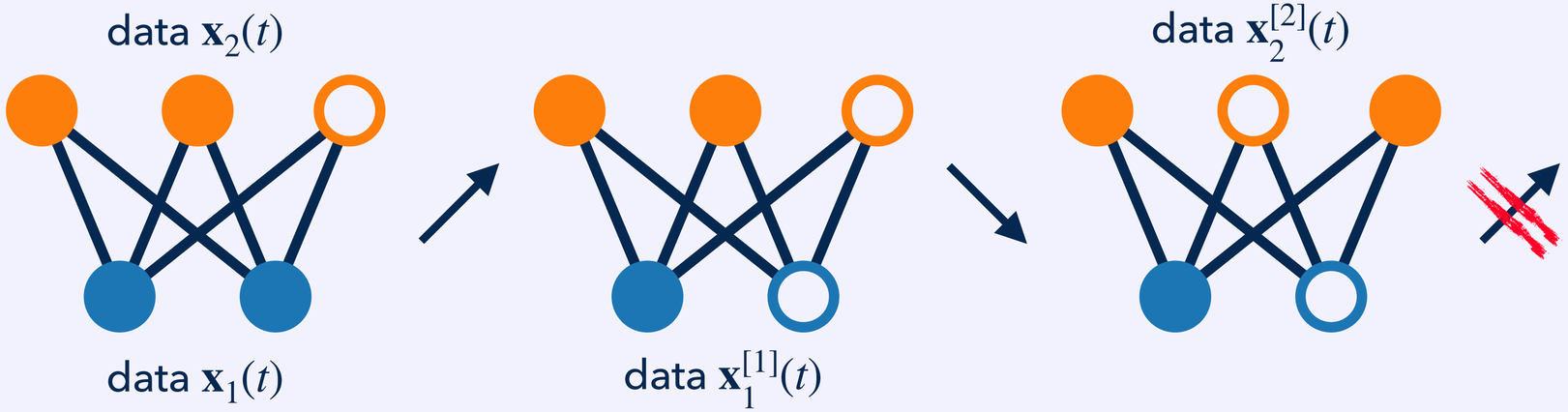
$\approx$ expectation $\sum_{t=1}^{T} \mathbf{x}^{[k]}(t) \big/ T$

- for all data t

$$\text{data } \mathbf{x}(t) \quad \rightsquigarrow \quad \underbrace{\mathbf{x}_1^{[1]}(t) \rightsquigarrow \mathbf{x}_2^{[1]}(t)}_{\text{step 1}}$$

The **Contrastive Divergence** (**CD-*k***) method runs MCMC transitions for k times (e.g. $k = 1$)

- the approximated gradient is biased
- but somehow it works well



Contrastive Divergence

The CD has achieved an empirical success

- empirical evidence that the CD estimator is biased

But the CD estimator is consistent under certain conditions (Jiang–Wu–Jin–Wong 2018)

- in exponential families, there exists k such that the CD-k estimator is consistent:

$$\lim_{T\to\infty} \operatorname*{plim}_{M\to\infty} \frac{1}{M} \sum_{m=1}^{M} \left(\mathbf{W}_{(m)}^{[k]}, \mathbf{b}_{(m)}^{[k]} \right) = \left(\mathbf{W}^{\text{true}}, \mathbf{b}^{\text{true}} \right)$$

Their condition is **not** satisfied in machine learning settings

- but it is satisfied in our 2SM-RBM applications

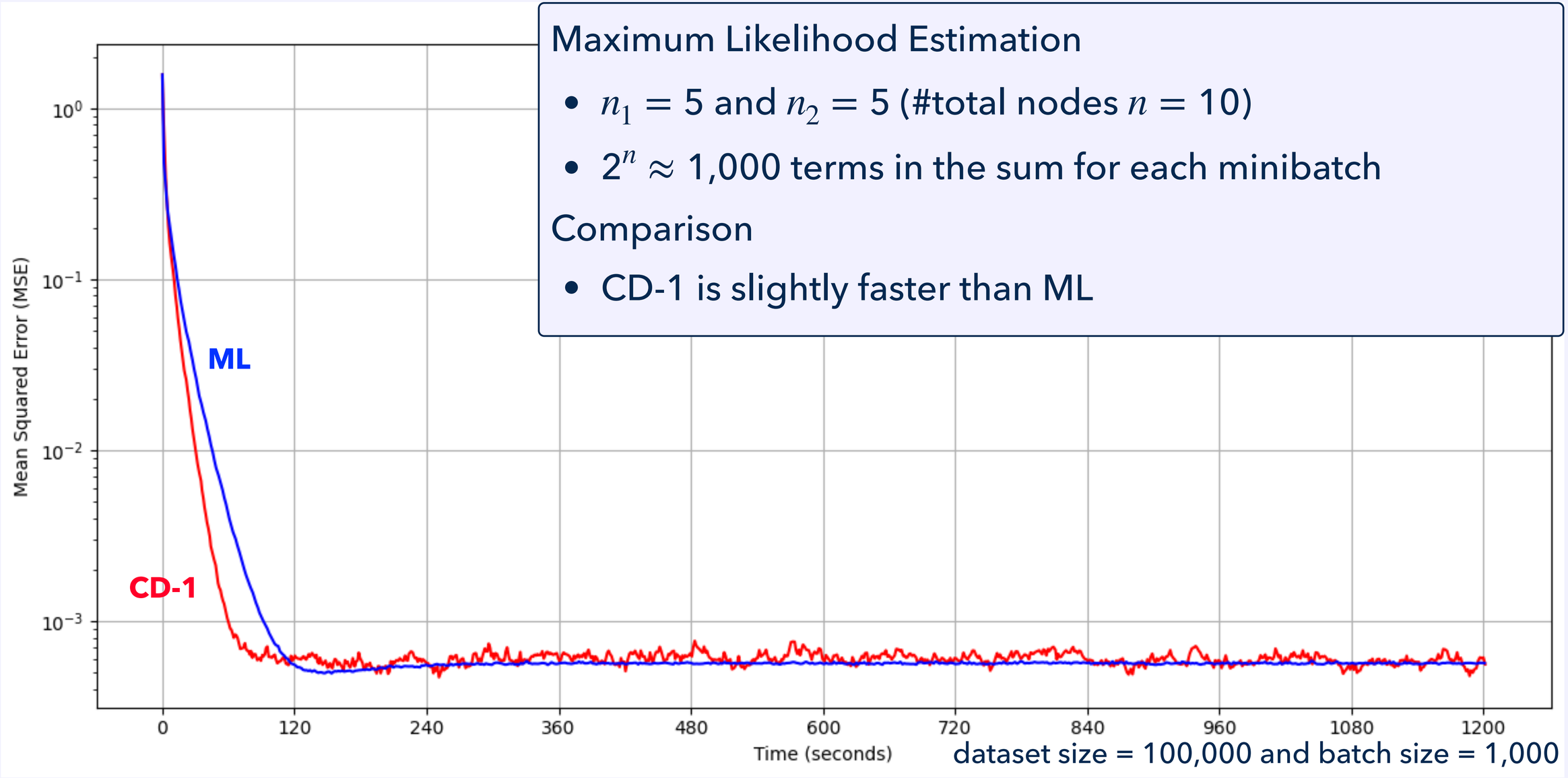
Simulation

Maximum Likelihood Estimation

- $n_1 = 5$ and $n_2 = 5$ (#total nodes $n = 10$)
- $2^n \approx 1{,}000$ terms in the sum for each minibatch

Comparison

- CD-1 is slightly faster than ML



dataset size = 100,000 and batch size = 1,000

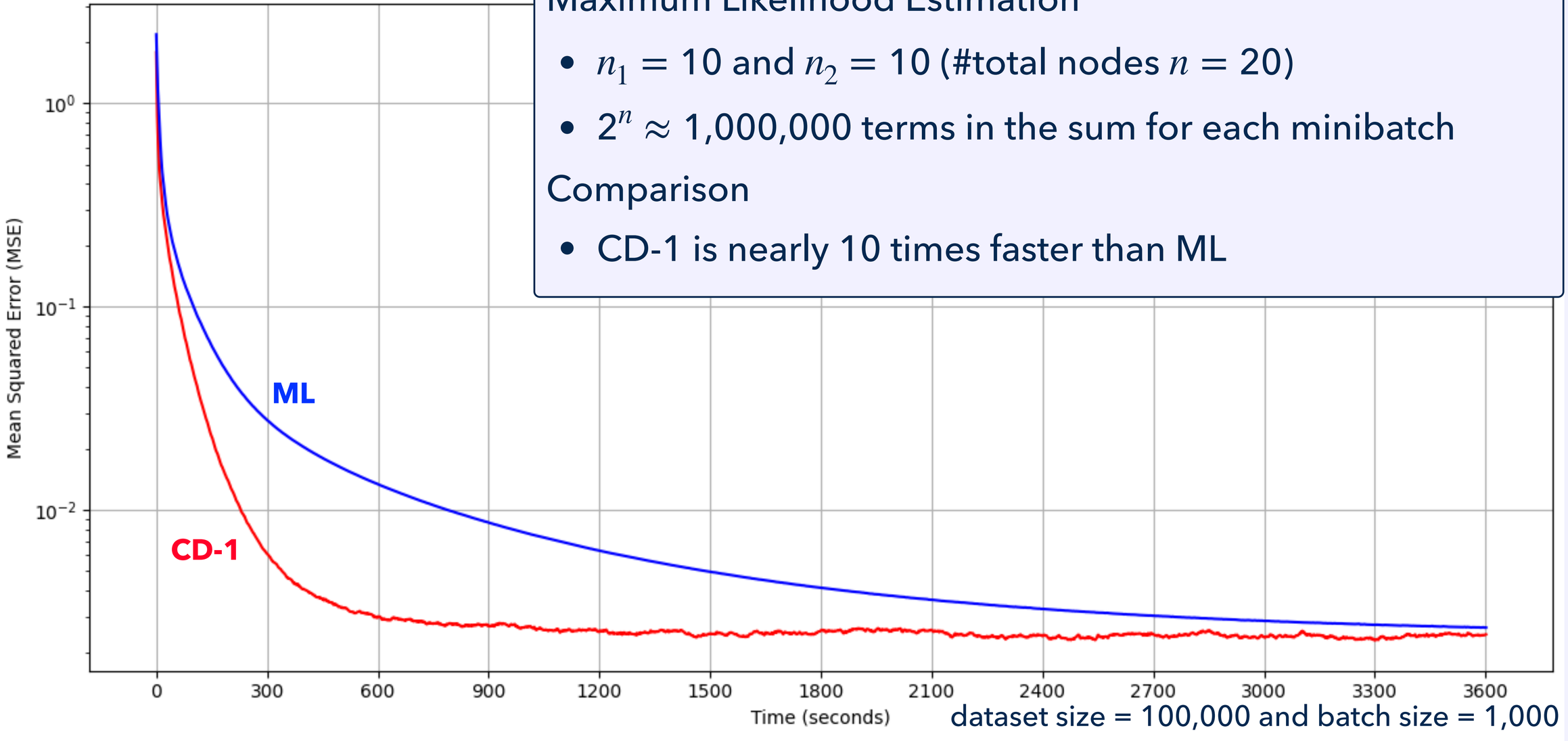
Simulation

Maximum Likelihood Estimation

- $n_1 = 10$ and $n_2 = 10$ (#total nodes $n = 20$)
- $2^n \approx 1{,}000{,}000$ terms in the sum for each minibatch

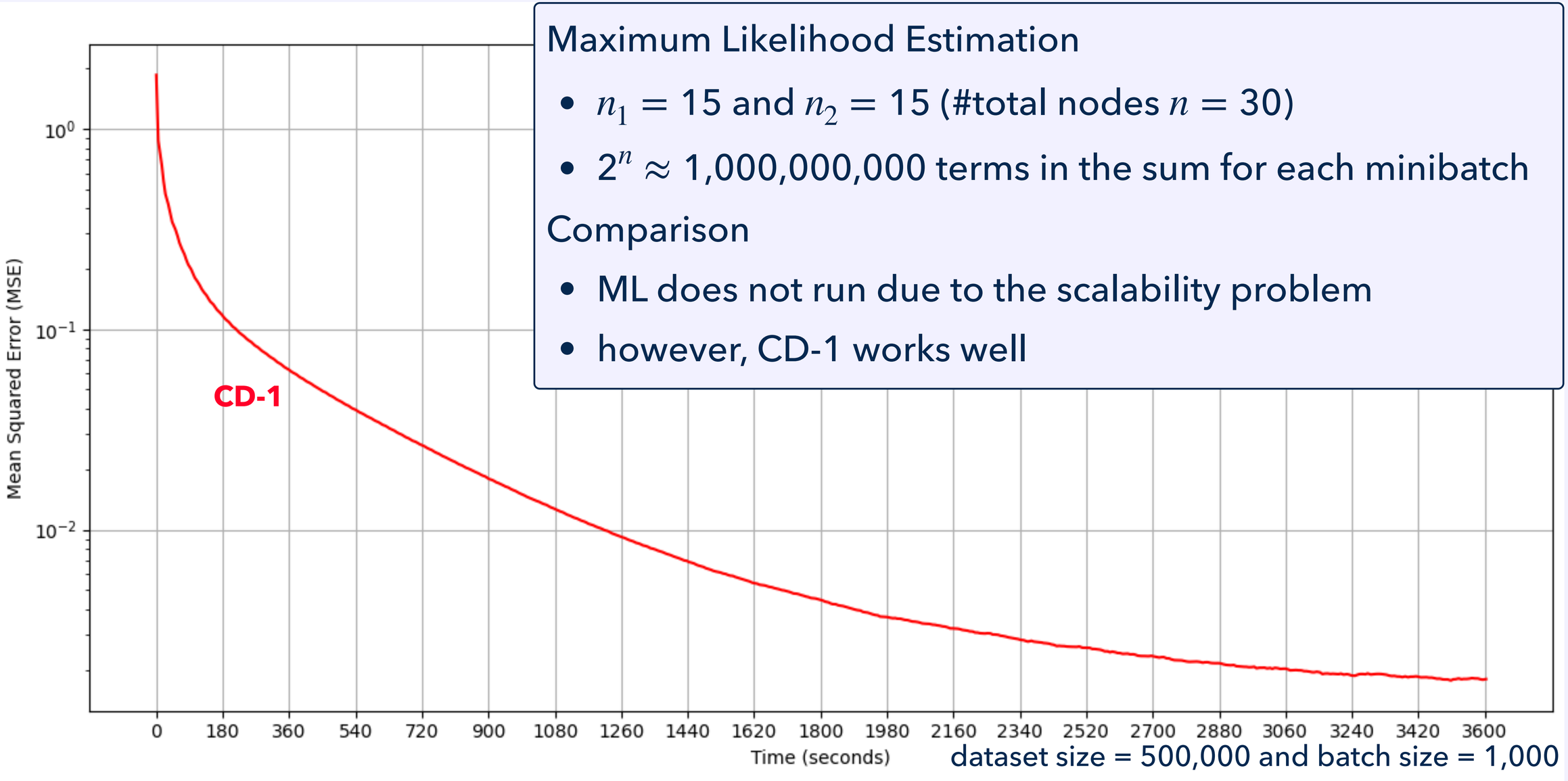
Comparison

- CD-1 is nearly 10 times faster than ML



dataset size = 100,000 and batch size = 1,000

Simulation



Maximum Likelihood Estimation

- $n_1 = 15$ and $n_2 = 15$ (#total nodes $n = 30$)
- $2^n \approx 1{,}000{,}000{,}000$ terms in the sum for each minibatch

Comparison

- ML does not run due to the scalability problem
- however, CD-1 works well

dataset size = 500,000 and batch size = 1,000

Summary

A tractable model of two-sided markets with heterogeneous networks

Two-sided markets are equivalent to (fully visible) RBMs

- CD is a scalable algorithm for RBMs
- CD is a consistent estimator in our two-sided market applications

A startup will set prices $\mathbf{p}$

- if the startup assumes network homogeneity by misspecification, it loses revenue