Decentralized Estimation in Open Multi-Agent Systems

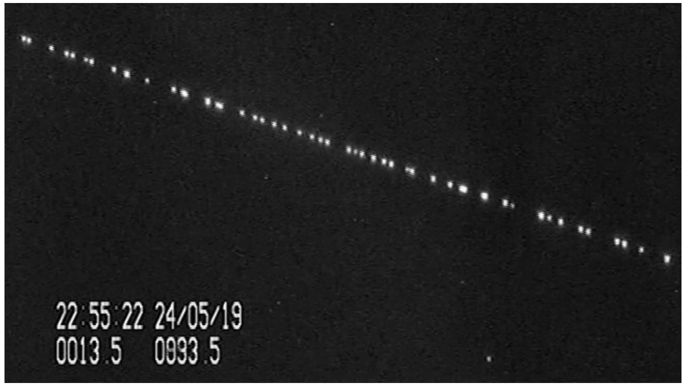
Sébastien Colla, Charles Monnoyer de Galland and Julien M. Hendrickx

European Control Conference

June 30, 2021

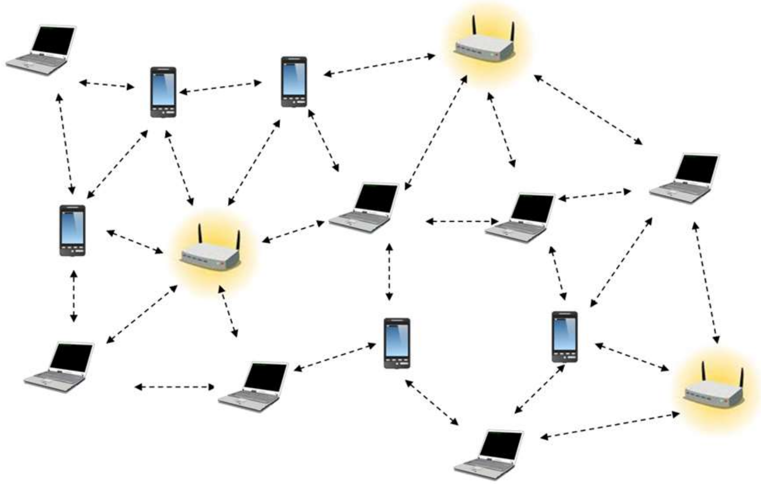
Multi-Agent systems

Starlink

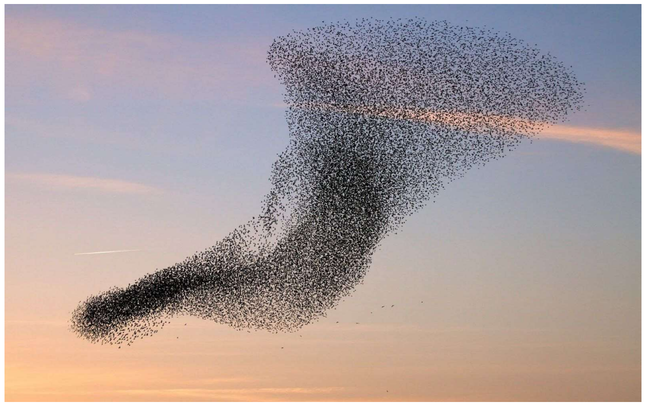


Multi-robot system

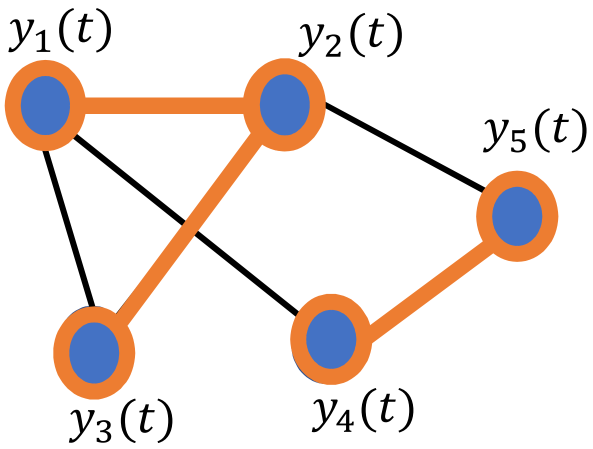
Ad-hoc and sensor networks



Flock of birds



Multi-Agent systems



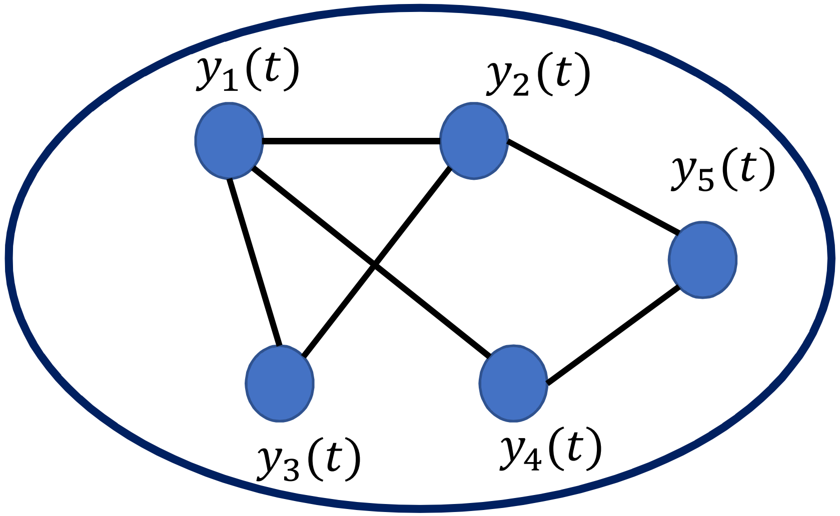
Under some assumption on the network,

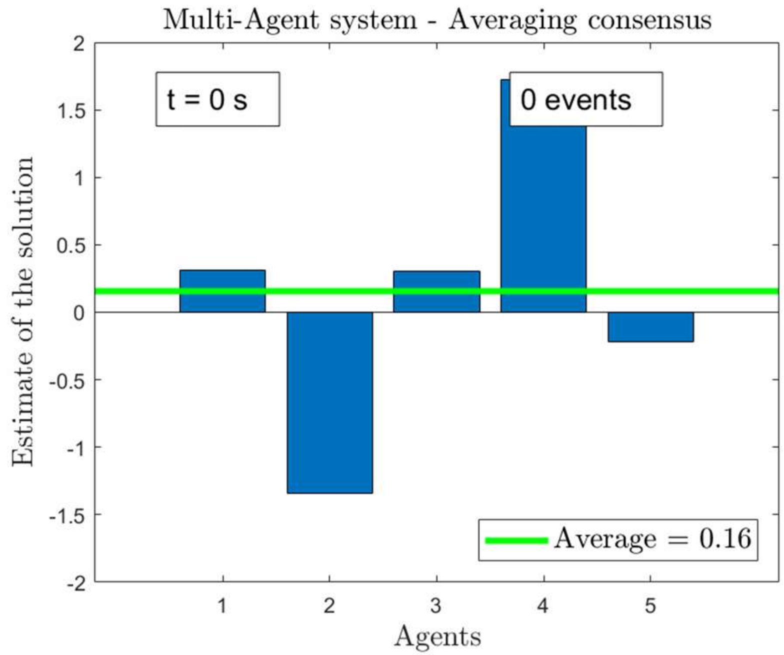
$$\lim_{t \to \infty} y(t) = y^* \qquad \text{For some desirable } y^*$$

e.g. for consensus: $y^* \in \mathrm{span}\{1\}$

Multi-Agent systems

Closed system





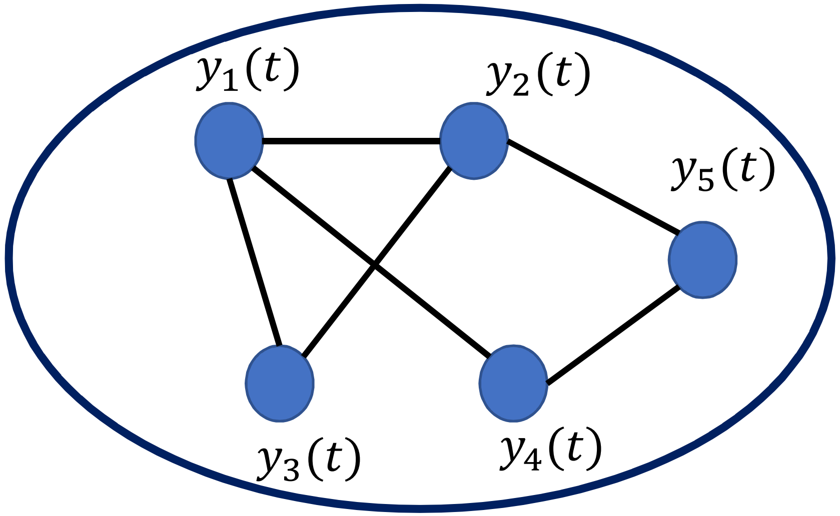
Under some assumption on the network,

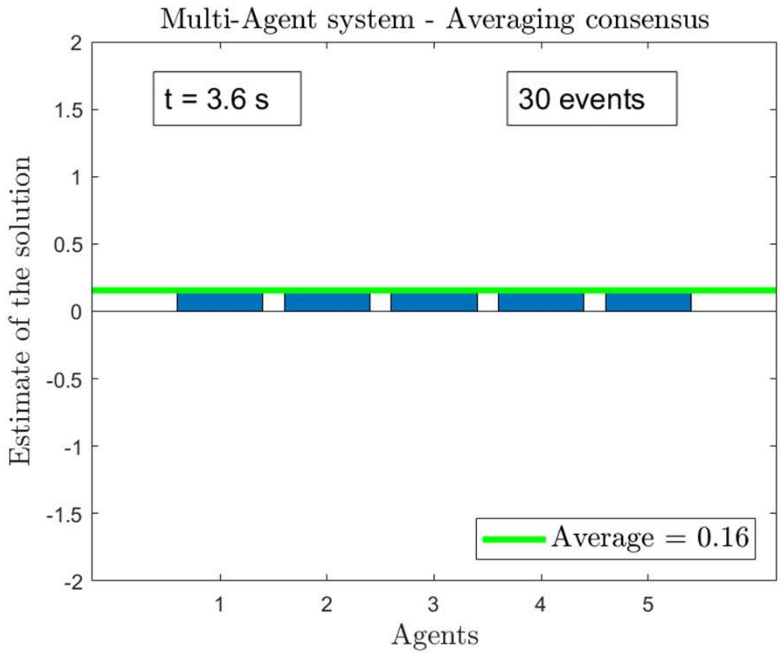
$$\lim_{t\to\infty} y(t) = y^* \qquad \text{For some desirable } y^*$$

e.g. for consensus: $y^* \in \text{span}\{1\}$

Multi-Agent systems

Closed system



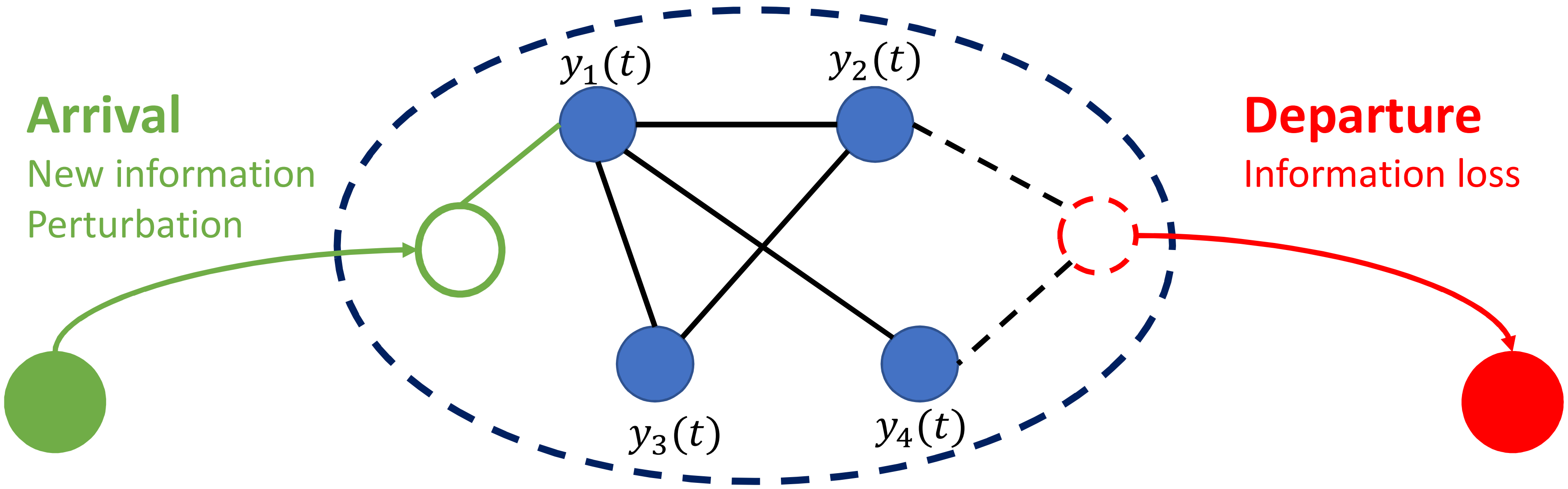


Under some assumption on the network,

$$\lim_{t \to \infty} y(t) = y^* \qquad \text{For some desirable } y^*$$

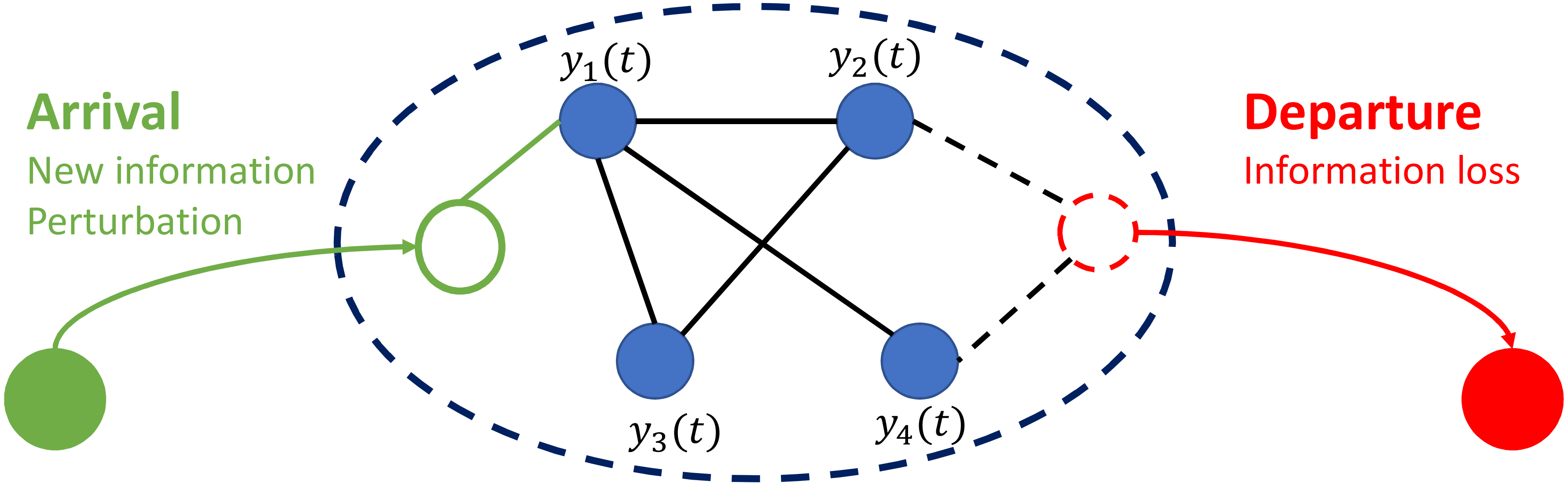
e.g. for consensus: $y^* \in \text{span}\{1\}$

Open Multi-Agent systems



Many multi-agent systems are **open by nature**

Open Multi-Agent systems



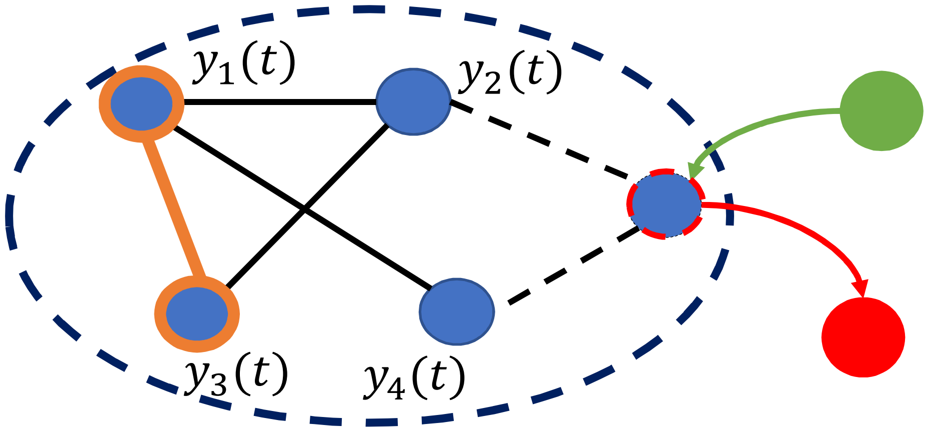
Challenging design and analysis:

- Variable dimension
- Variable objective
- No more convergence

Problem Statement

Each agent has a **noisy measurement** of $\boldsymbol{\mu}$: $z_i \sim \mathcal{N}(\mu, \sigma^2)$ (iid.)

GOAL: Estimating $\boldsymbol{\mu}$ $\qquad y_i(t) \to \mu$



Replacement assumption
→ **constant size system $\boldsymbol{n}$**

λ_r replacement rate

Communications:

Random, pairwise, asynchronous, symmetric

λ_c communication rate

Agents have:

Bounded memory
No identifier
Identical algorithms

Outline

1. **Performance Limitations**
2. Gossip averaging algorithm
3. Symmetric Push-Sum

Performance Limitations

Best possible way for agent i to estimate μ:

$$y_i^{best}(t) = \text{average}\,\{z_k \mid k \text{ has influenced } i \text{ at time } t\}$$

Ideal : requires a growing memory + identifiers

Lower bound on the expected performance of **any algorithm:**

Performance metric: $$MSE(t) = \frac{1}{n}\sum_{i=1}^{n}(y_i(t) - \mu)^2$$

$$\boxed{\mathbb{E}[MSE(t)] \geq \mathbb{E}[MSE_{best}(t)]}$$

Outline

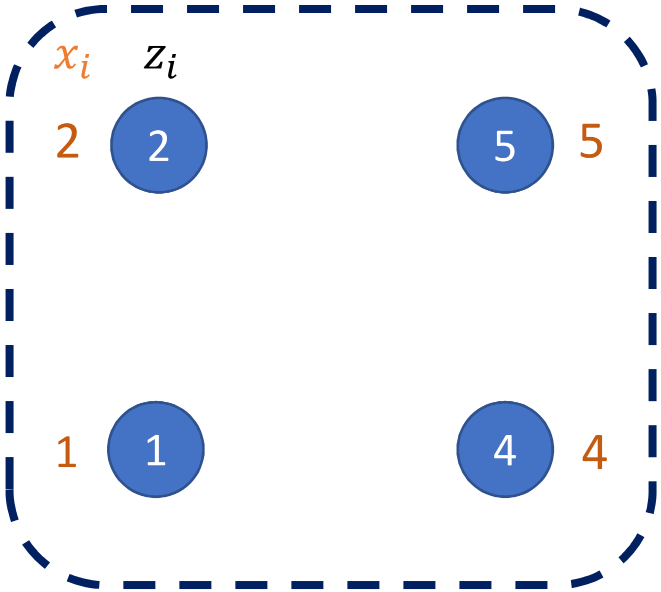
1. Performance Limitations
2. **Gossip averaging algorithm**
3. Symmetric Push-Sum

The gossip averaging algorithm

Initialization: $x_i = z_i$

Interaction $i - j$: $x_i^+ = x_j^+ = \frac{x_i + x_j}{2}$

Estimate of μ : $y_i(t) = x_i(t)$

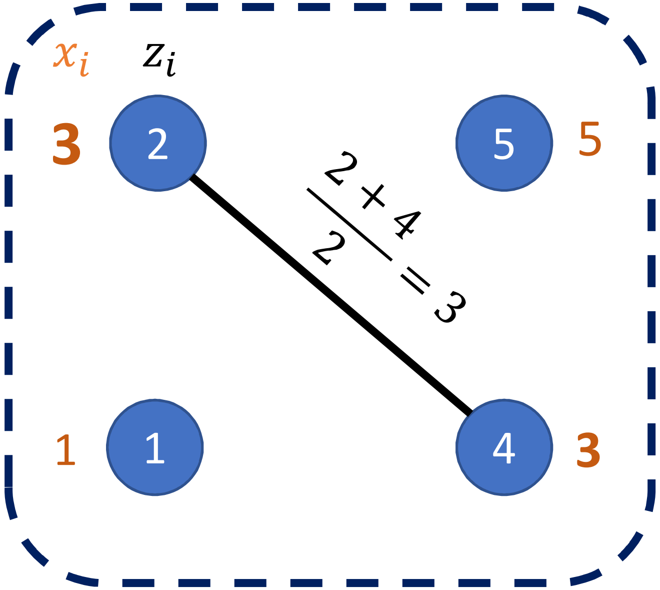


The gossip averaging algorithm

Initialization: $x_i = z_i$

Interaction $i - j$: $x_i^+ = x_j^+ = \frac{x_i + x_j}{2}$

Estimate of μ : $y_i(t) = x_i(t)$



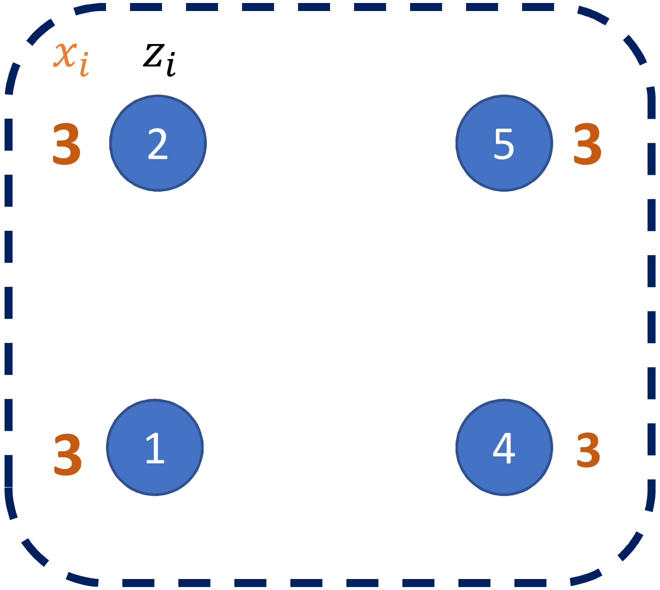
The gossip averaging algorithm

Initialization: $x_i = z_i$

Interaction $i - j$: $x_i^+ = x_j^+ = \frac{x_i + x_j}{2}$

Estimate of μ : $y_i(t) = x_i(t)$

Closed: all x_i converge to $\bar{x} = \frac{1}{n}\sum_{k=1}^{n} z_k$

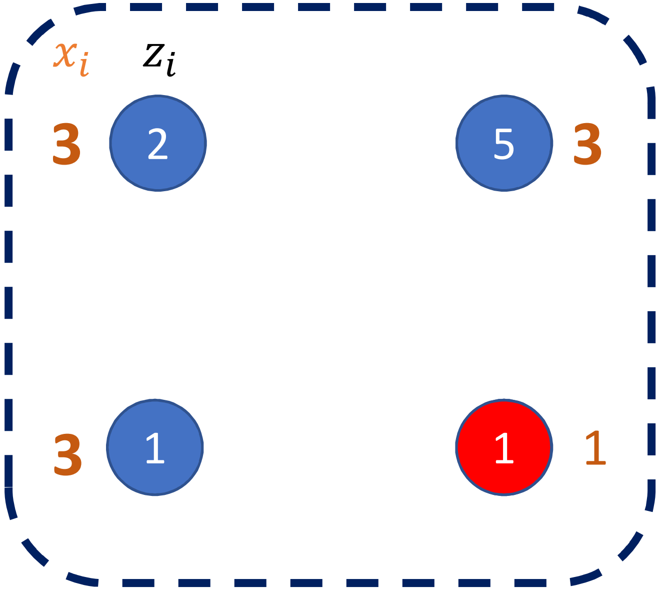


The gossip averaging algorithm

Initialization: $x_i = z_i$

Interaction $i - j$: $x_i^+ = x_j^+ = \frac{x_i + x_j}{2}$

Estimate of μ : $y_i(t) = x_i(t)$



Closed: all x_i converge to $\bar{x} = \frac{1}{n}\sum_{k=1}^{n} z_k$

Replacement of j by $n + 1$:

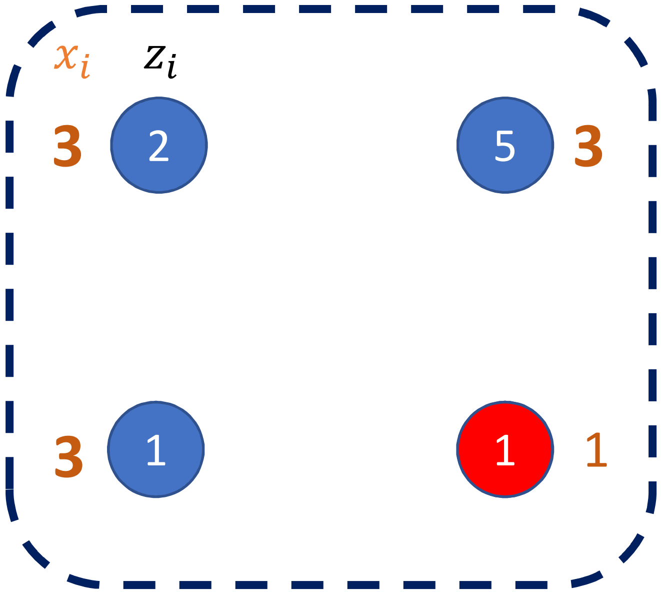
All x_i re-converge to $\frac{n-1}{n}\,\bar{x} + \frac{1}{n} z_{n+1}$

The gossip averaging algorithm

Initialization: $x_i = z_i$

Interaction $i - j$: $x_i^+ = x_j^+ = \frac{x_i + x_j}{2}$

Estimate of μ: $y_i(t) = x_i(t)$



Closed: all x_i converge to $\bar{x} = \frac{1}{n}\sum_{k=1}^{n} z_k$

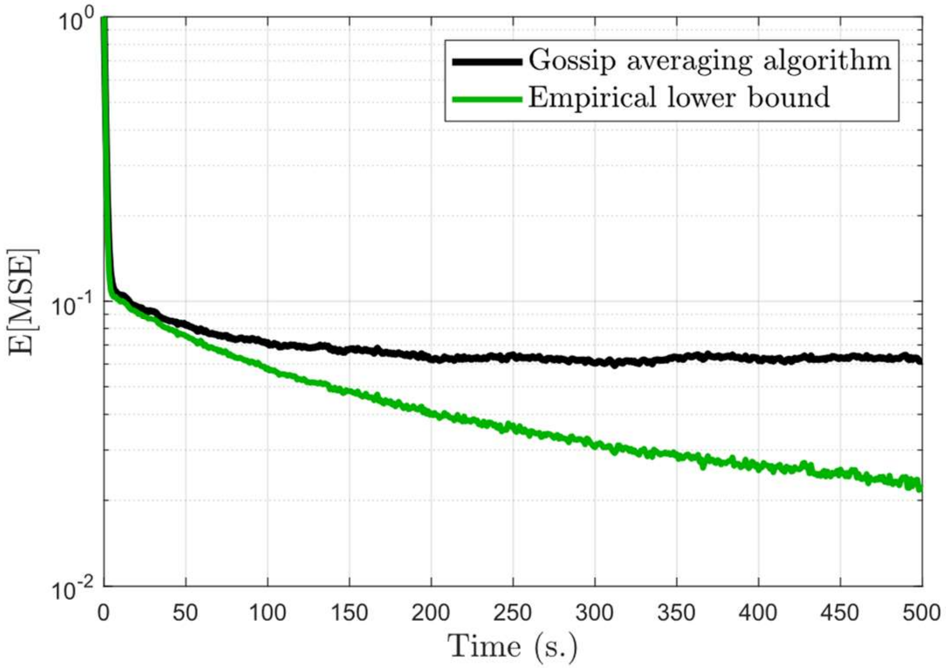
Replacement of j by $n+1$:

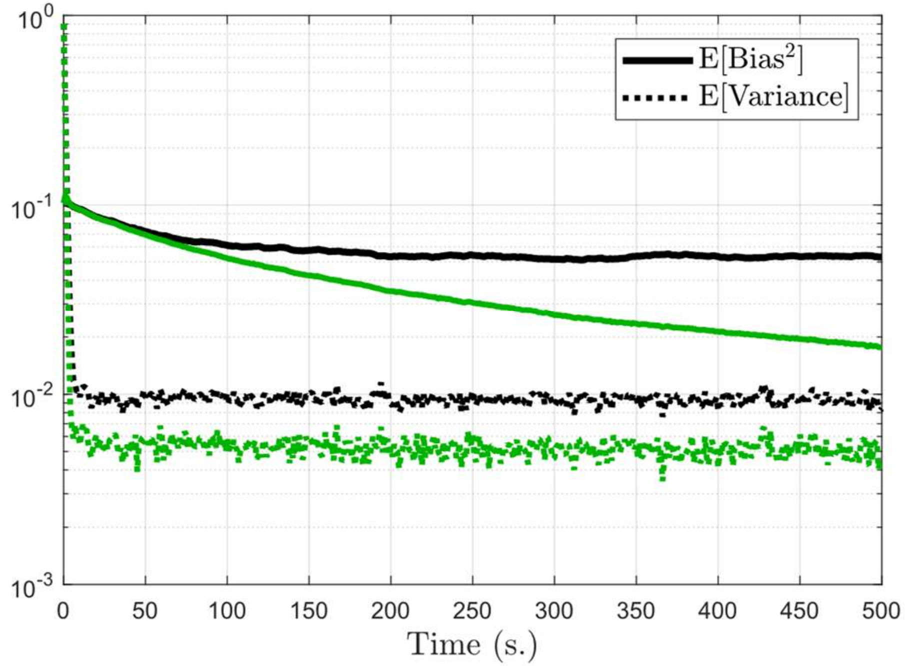
All x_i re-converge to $\frac{n-1}{n}\bar{x} + \frac{1}{n} z_{n+1}$

$$\frac{n-1}{n^2}\sum_{k=1}^{n} z_k + \frac{1}{n} z_{n+1} \neq \frac{1}{n+1}\sum_{k=1}^{n+1} z_k$$

New agent has more weight

Performance of the gossip algorithm





$n = 10, \frac{\lambda_c}{\lambda_r} = 100$, 5000 realizations

$\sim$ 10 com. / second

$$\text{Bias}^2(t) = \bar{y}(t) - \mu$$

$$\text{Var }(t) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i(t) - \bar{y}(t)\right)^2$$

The gossip algorithm: a baseline

Theoretical evolution of an open multi-agent system subject to gossip algorithm exists *[J.M. Hendirkx, S. Martin 2017]*

➡ Baseline algorithm with theoretical guarantees

Issues with the gossip algorithm

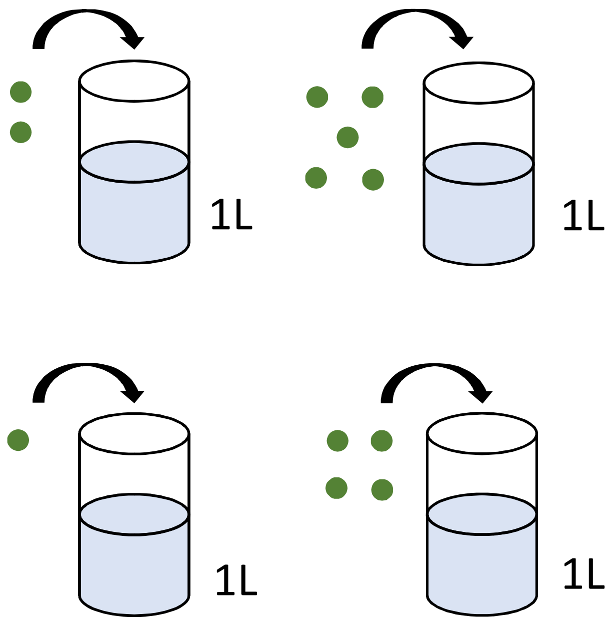
1. Not equal weight for all agents
   → Large squared bias
2. Disruptions due to replacements
   → Large Variance

Symmetric Push-Sum (SPS)

Gossip - Issue 1: too much weight for new agents

→ **Idea**: give decreasing weights to incoming agents

Symmetric Push-Sum: add a weight for each agent



x_i : quantity

w_i : volume

$y_i = \frac{x_i}{w_i}$: estimate

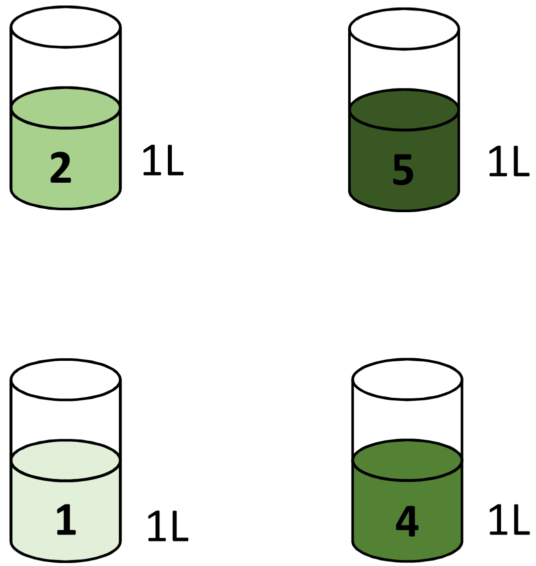
$y_i(0) = z_i$

Symmetric Push-Sum (SPS)

Gossip - Issue 1: too much weight for new agents

→ **Idea**: give decreasing weights to incoming agents

Symmetric Push-Sum: add a weight for each agent



x_i : quantity, w_i : volume

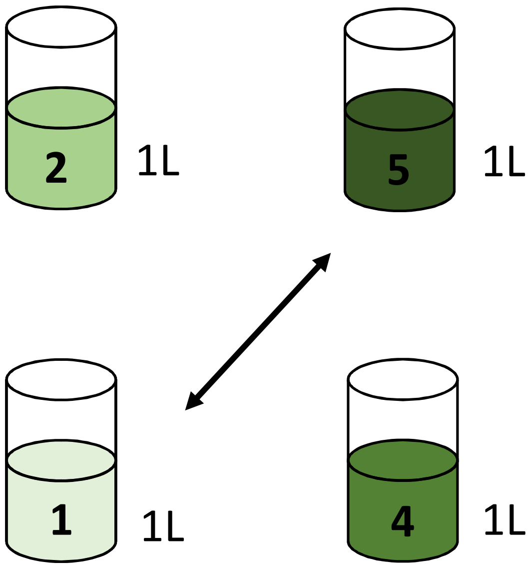
$y_i = \frac{x_i}{w_i}$: estimate

Symmetric Push-Sum (SPS)

Gossip - Issue 1: too much weight for new agents

→ **Idea**: give decreasing weights to incoming agents

Symmetric Push-Sum: add a weight for each agent



x_i : quantity, w_i : volume

$y_i = \frac{x_i}{w_i}$: estimate

Communication:
Mix their 'content' and share it equally

$$x_i^+ = x_j^+ = \frac{x_i + x_j}{2}$$

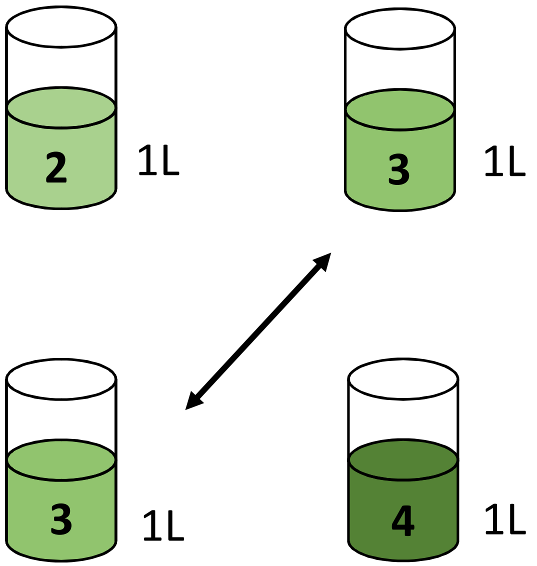
$$w_i^+ = w_j^+ = \frac{w_i + w_j}{2}$$

Symmetric Push-Sum (SPS)

Gossip - Issue 1: too much weight for new agents

→ **Idea**: give decreasing weights to incoming agents

Symmetric Push-Sum: add a weight for each agent



x_i : quantity, w_i : volume

$y_i = \frac{x_i}{w_i}$: estimate

Communication:
Mix their 'content' and share it equally

$$x_i^+ = x_j^+ = \frac{x_i + x_j}{2}$$

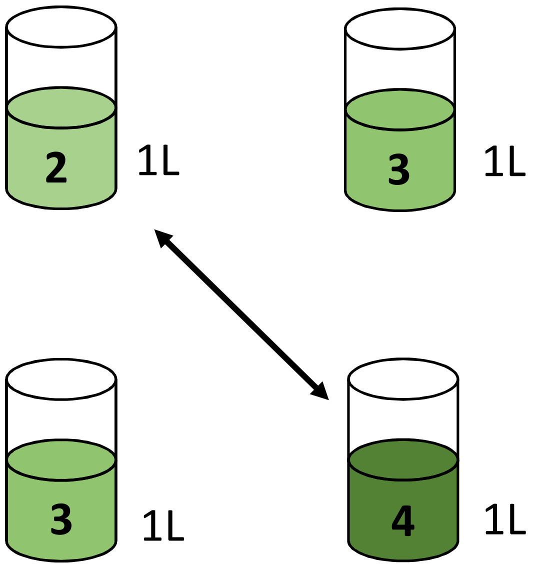
$$w_i^+ = w_j^+ = \frac{w_i + w_j}{2}$$

Symmetric Push-Sum (SPS)

Gossip - Issue 1: too much weight for new agents

→ **Idea**: give decreasing weights to incoming agents

Symmetric Push-Sum: add a weight for each agent



x_i : quantity, w_i : volume

$y_i = \frac{x_i}{w_i}$: estimate

Communication:
Mix their 'content' and share it equally

$$x_i^+ = x_j^+ = \frac{x_i + x_j}{2}$$

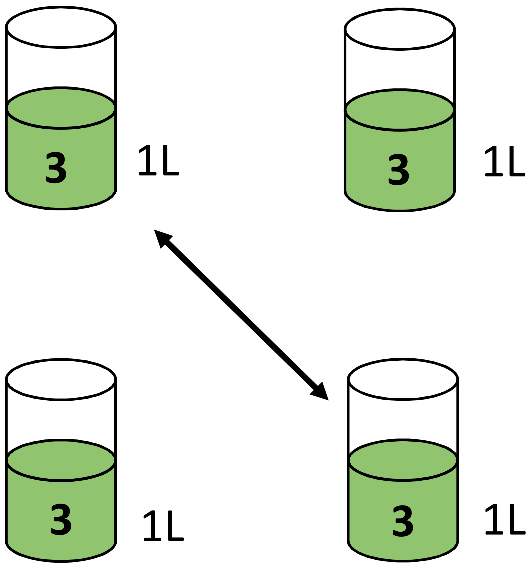
$$w_i^+ = w_j^+ = \frac{w_i + w_j}{2}$$

Symmetric Push-Sum (SPS)

Gossip - Issue 1: too much weight for new agents

→ **Idea**: give decreasing weights to incoming agents

Symmetric Push-Sum: add a weight for each agent



x_i : quantity, w_i : volume

$y_i = \frac{x_i}{w_i}$: estimate

Communication:
Mix their 'content' and share it equally

$$x_i^+ = x_j^+ = \frac{x_i + x_j}{2}$$

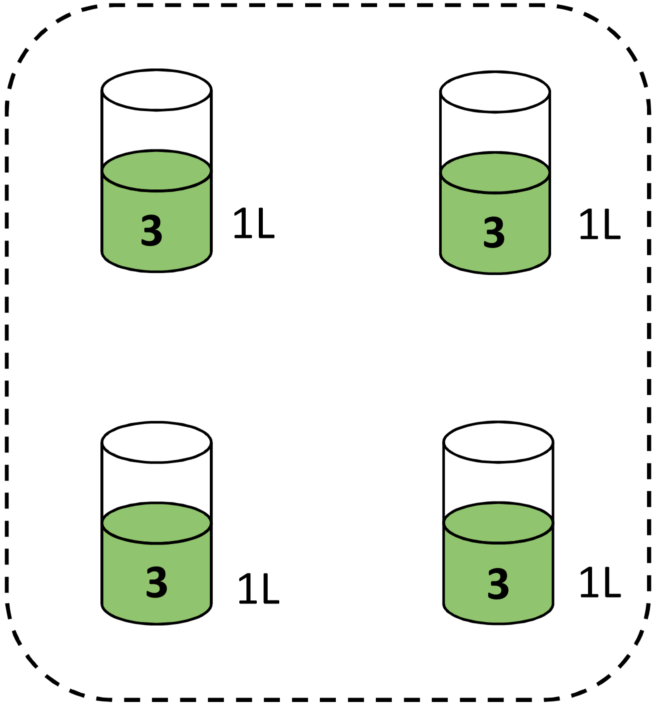
$$w_i^+ = w_j^+ = \frac{w_i + w_j}{2}$$

Symmetric Push-Sum (SPS)

Gossip - Issue 1: too much weight for new agents

→ **Idea**: give decreasing weights to incoming agents

Symmetric Push-Sum: add a weight for each agent



x_i : quantity, w_i : volume

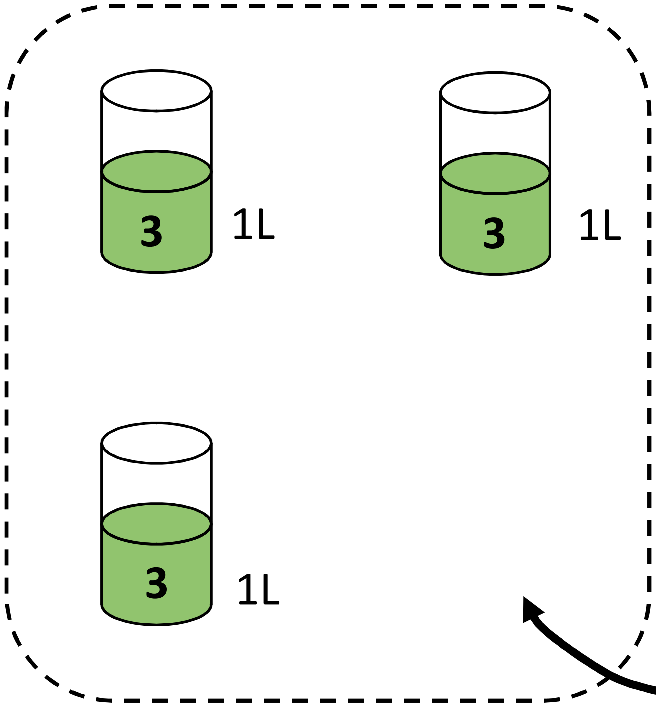
$y_i = \frac{x_i}{w_i}$: estimate

Symmetric Push-Sum (SPS)

Gossip - Issue 1: too much weight for new agents

→ **Idea**: give decreasing weights to incoming agents

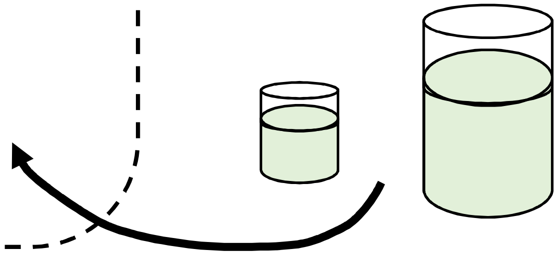
Symmetric Push-Sum: add a weight for each agent



x_i : quantity, w_i : volume

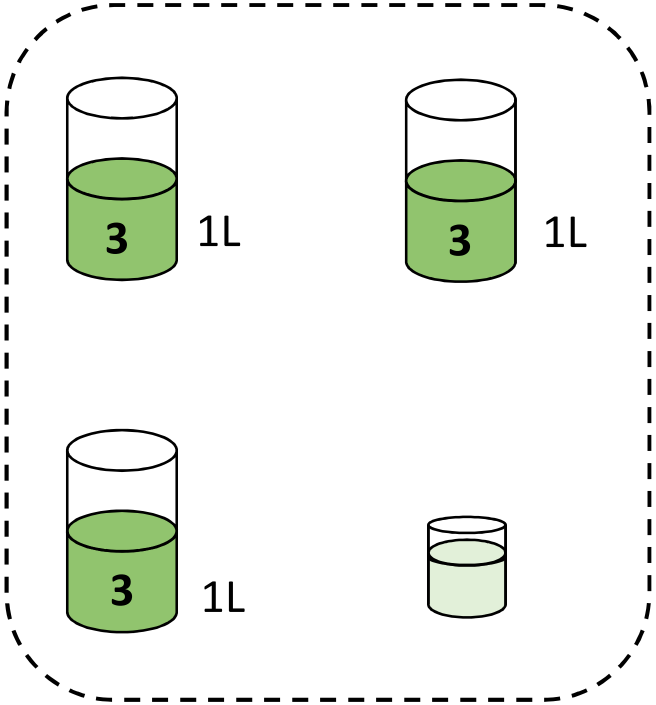
$y_i = \frac{x_i}{w_i}$: estimate

Replacement:
Which volume should we choose for the first new agent $n+1$?



$$y_{n+1}(0) = z_{n+1}$$

Symmetric Push-Sum (SPS)



Replacement:
Which volume should we choose for the first new agent $n+1$?

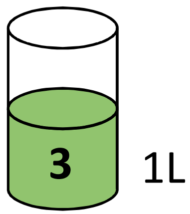
First new agent $n+1$ $\quad$ $y_{n+1}(0) = z_{n+1}$

We suppose for other agents $\quad$ $y_i = \frac{1}{n}\sum_{k=1}^{n} z_k$

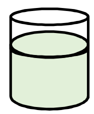
Goal: Compute the external average

$$\boxed{y_i \to \bar{z}_{ext} = \frac{1}{n+1}\sum_{k=1}^{n+1} z_k}$$

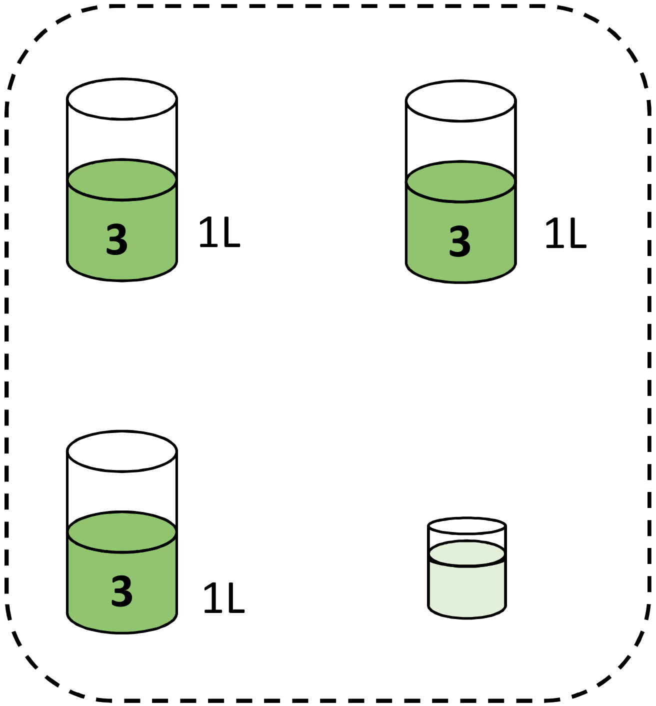
Past agent:



$$w_{n+1}(0) = \frac{n-1}{n}$$



Symmetric Push-Sum (SPS)



Replacement:
Which volume should we choose for the first new agent $n+1$?

First new agent $n+1$ $\quad y_{n+1}(0) = z_{n+1}$

We suppose for other agents $\quad y_i = \frac{1}{n}\sum_{k=1}^{n} z_k$

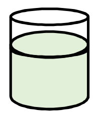
Goal: Compute the external average

$$y_i \rightarrow \bar{z}_{ext} = \frac{1}{n+1}\sum_{k=1}^{n+1} z_k$$

Effect of the weights ?

Weighted average

$$w_{n+1}(0) = \frac{n-1}{n}$$



Symmetric Push-Sum: Summary

Symmetric Push-Sum algorithm (SPS):

Initialization:	choose $w_i(0)$	$x_i = w_i(0)\, z_i$
Interaction $i - j$:	$w_i^+ = w_j^+ = \frac{w_i + w_j}{2}$	$x_i^+ = x_j^+ = \frac{x_i + x_j}{2}$
Estimate of μ:	$y_i(t) = \frac{x_i(t)}{w_i(t)}$	

How to choose $w_i(0)$?

$w_i(0) = 1$ for the n initial agents

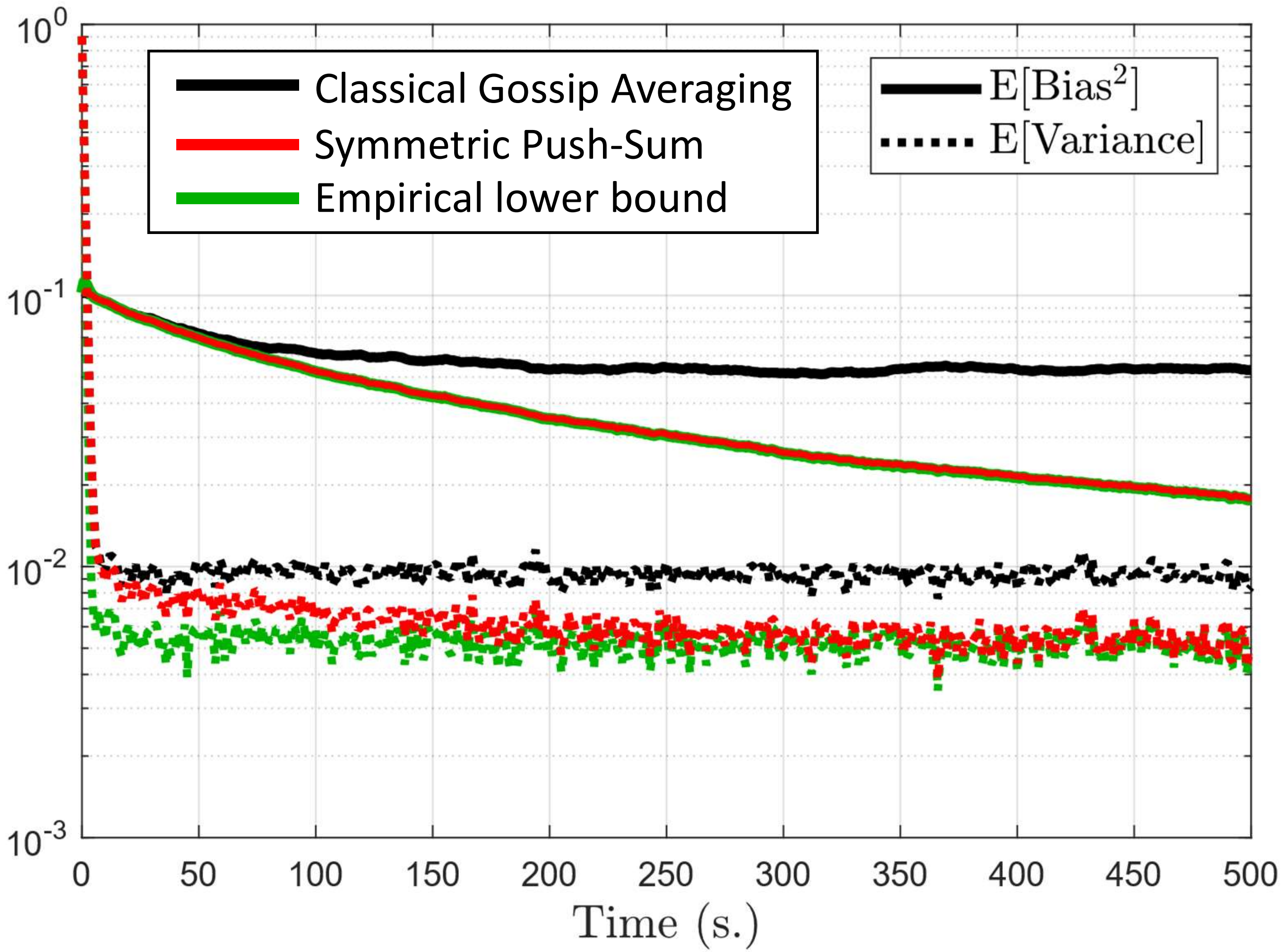
$\boxed{w_{n+k}(0) = \left(\frac{n-1}{n}\right)^k}$ for the k^{th} arriving agent

Supposing **convergence** occurs between each replacement**,**
(i.e. replacements are very infrequent)

these **decreasing weights** ensures that

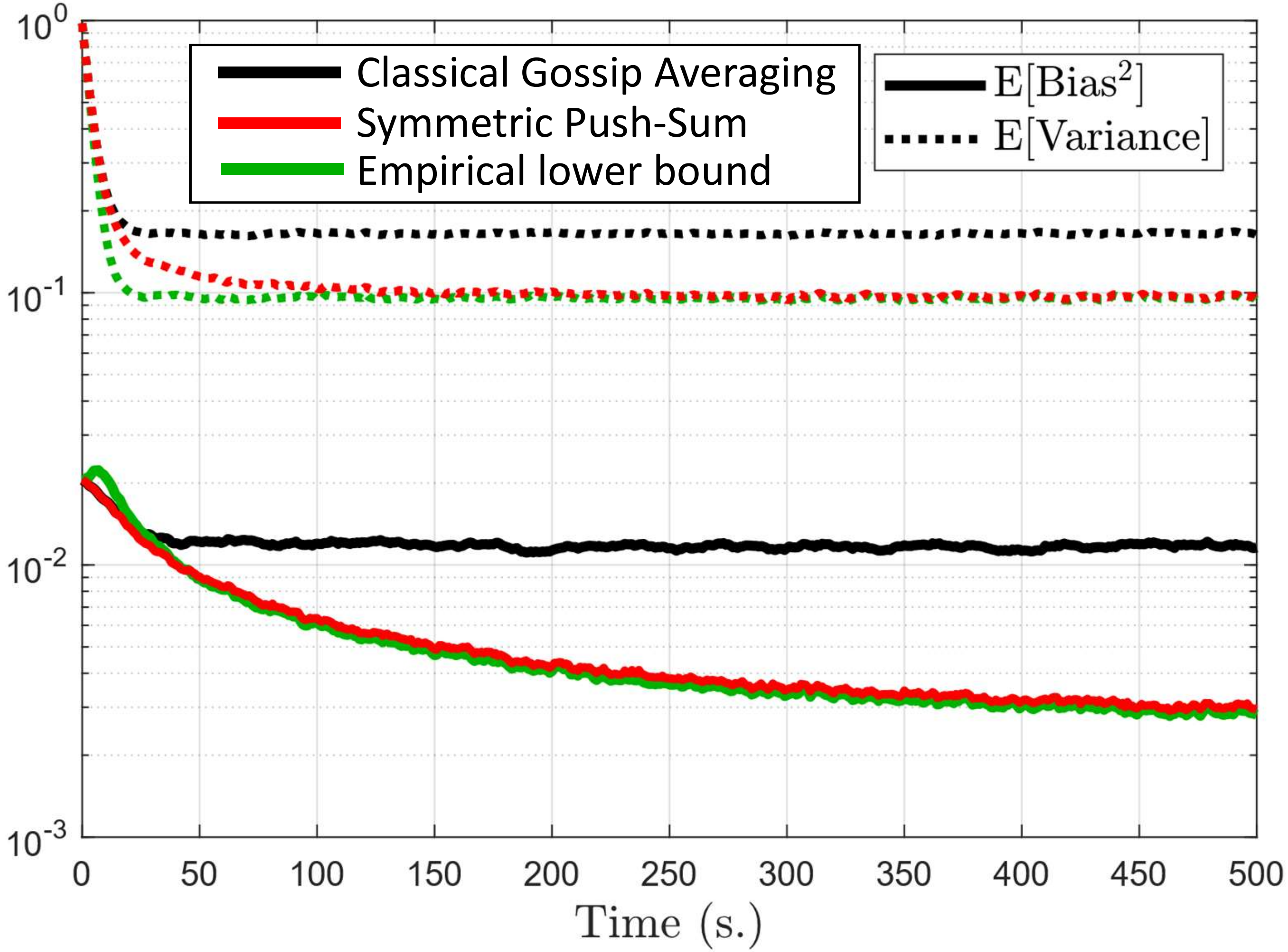
$$y_i(t) \to \frac{1}{n+k} \sum_{i=1}^{n+k} z_i$$ between each replacement

SPS: performance with infrequent replacements



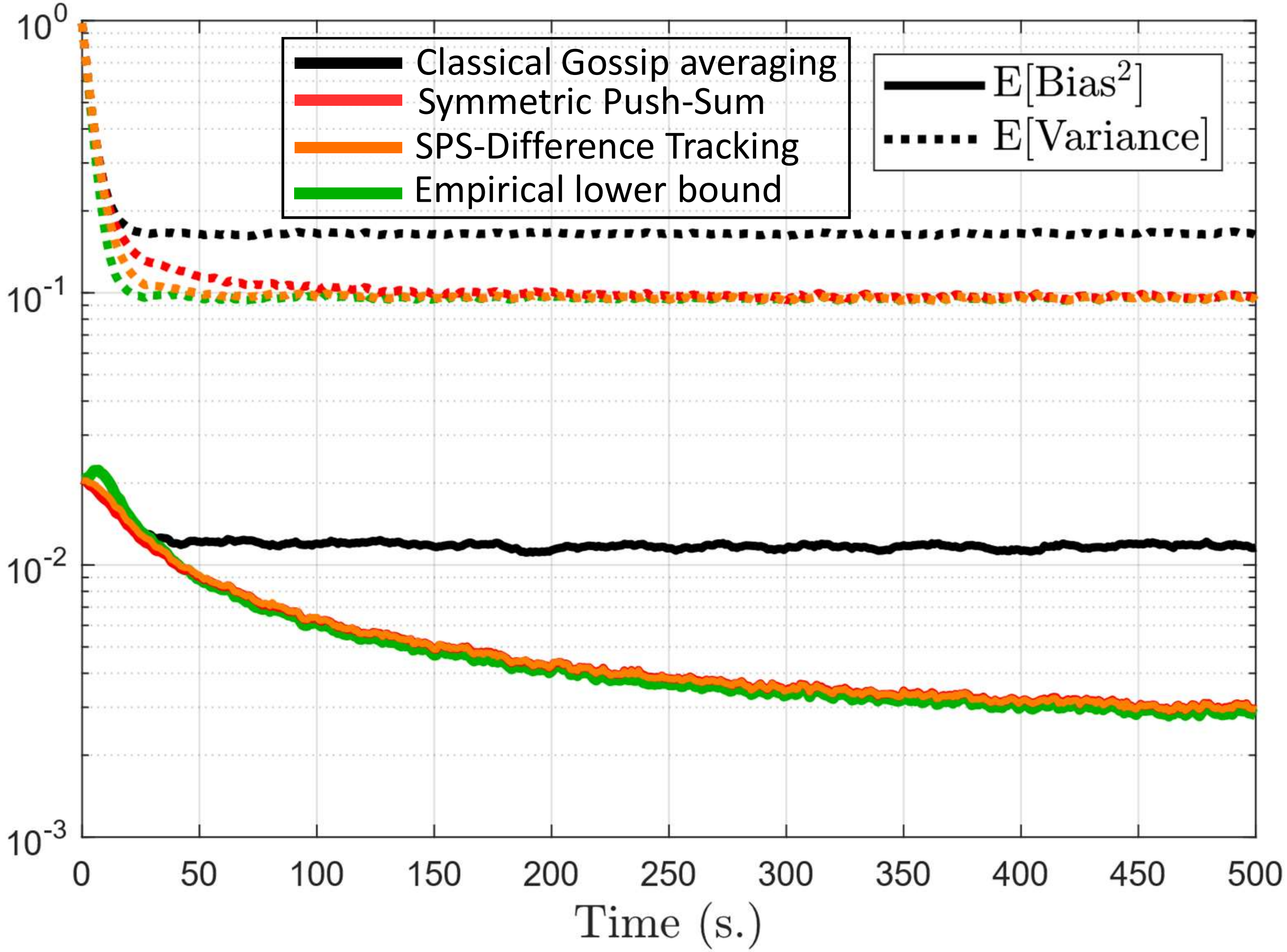
$n = 10, \frac{\lambda_c}{\lambda_r} = 100, \sim 10$ com./second, 5000 realizations

SPS: performance with frequent replacements



$n = 50, \frac{\lambda_c}{\lambda_r} = 5$, $\sim$ 10 com./second, 5000 realizations

Symmetric Push-Sum with difference tracking



$n = 50, \frac{\lambda_c}{\lambda_r} = 5, \sim$ 10 com./second, 5000 realizations

Conclusion

- **External averaging problem:** in the context of open multi-agent systems for decentralized estimation.
- Challenging problem with **limitations.**
- Well known **gossip averaging algorithm** does not answer challenges properly.
- **Symmetric Push-Sum** shows very good results to these challenges.

Open problems

- Develop **theoretical guarantees**
- Consider separate arrivals and departures