Automatic Performance Estimation for Decentralized Optimization

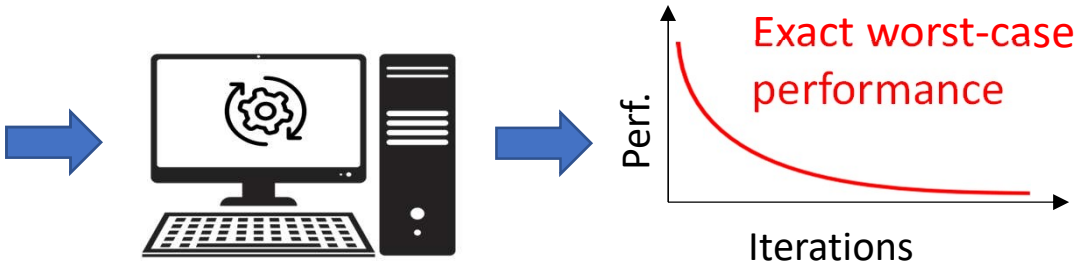
Sébastien Colla, Julien Hendrickx

Mathematical Engineering Department, UCLouvain

ECE, Miami University (OH) – October 3, 2022

```
function MyDecentralizedAlgo()
    N = 10; % number of agents
    x0 = init(N); % Initial point
    x = x0;

    for i=1:niter
        % any local computations
        % and local communications
        x = update(x,N);
    end
end
```

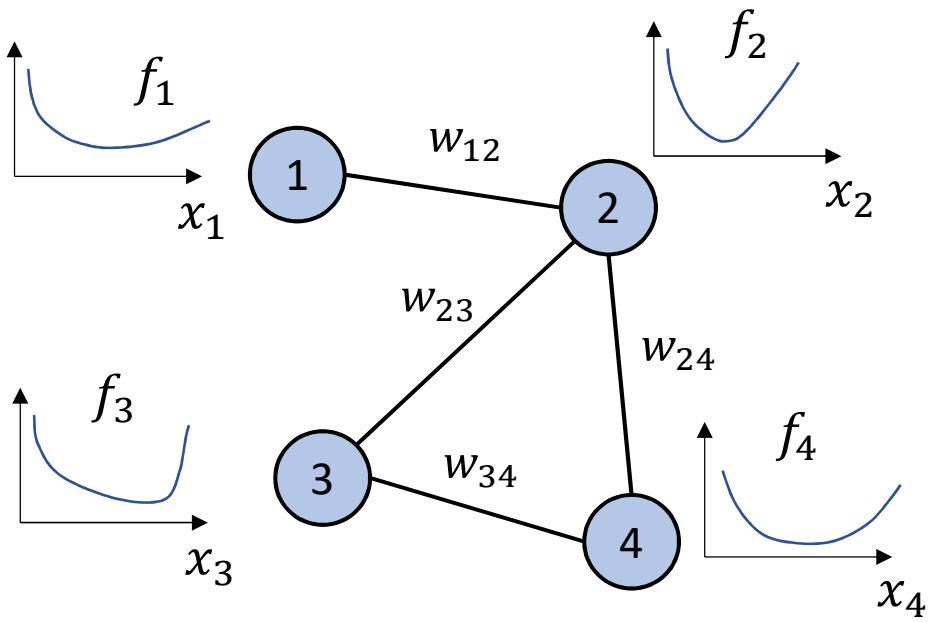


Decentralized Optimization

$$\min_{x} f(x) = \sum_{i=1}^{N} f_i(x)$$

$$\Updownarrow$$

$$\min_{x_1,\ldots,x_N} \sum_{i=1}^{N} f_i(x_i)$$
$$\text{s.t. } x_i = x_j \quad \forall\, (i,j) \text{ neighbors}$$



Decentralization

- Local function: f_i
- Local copy of x: x_i

Iterative algorithm

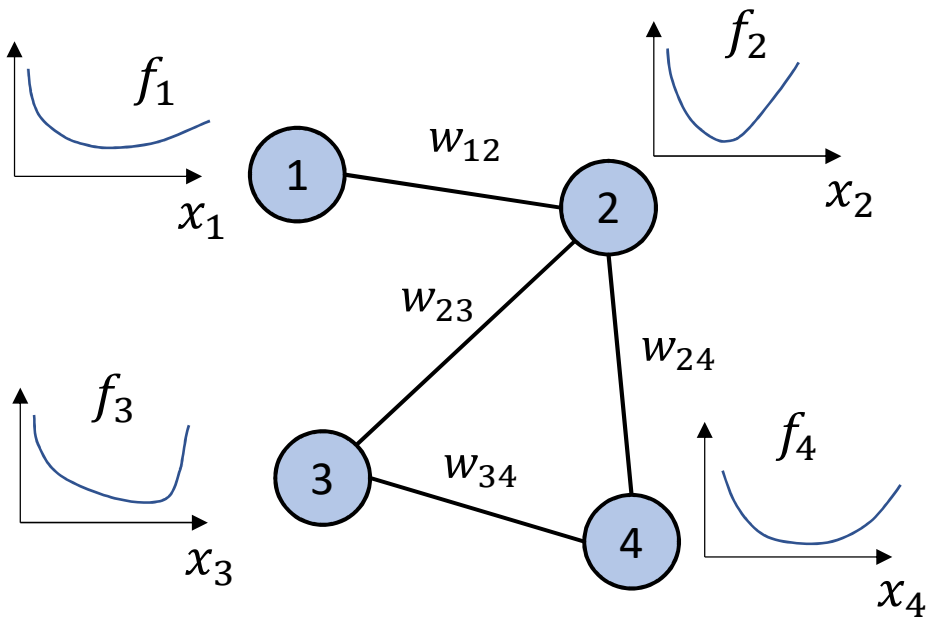
- Local computations
- Local communications (W)
  so that $x_i = x_j$ (eventually)

Decentralized Gradient Descent (DGD)

$$\min_{x_1,\ldots,x_N} \sum_i f_i(x_i)$$
$$\text{s.t. } x_i = x_j \quad \forall\, (i,j) \text{ neighbors}$$

Decentralization

- Local function: f_i
- Local copy of x: x_i



Decentralized Gradient Descent (DGD)

For each iteration k

$$y_i^k = \sum_j w_{ij}\, x_j^k \qquad \textit{Consensus step}$$

$$x_i^{k+1} = y_i^k - \alpha \nabla f_i(x_i^k) \qquad \textit{Local gradient step}$$

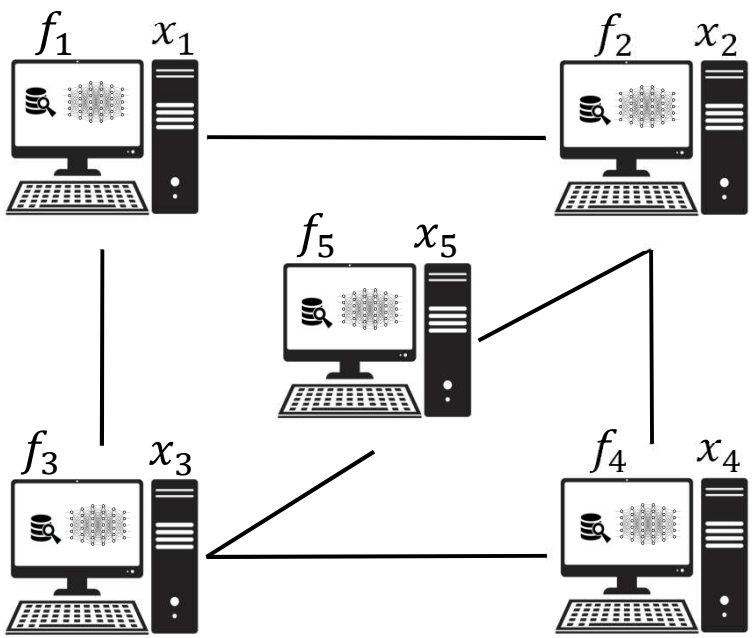
Motivations: Decentralized Machine Learning

Notations

- Model parameters x
- Data set $\{d \in \mathcal{D}\}$

Model training

$$\min_{x} \sum_{d \in D} \text{Error}(x, d) + \text{regul (x)}$$



Decentralization

- Part of the data $\mathcal{D}_i$
- Local function

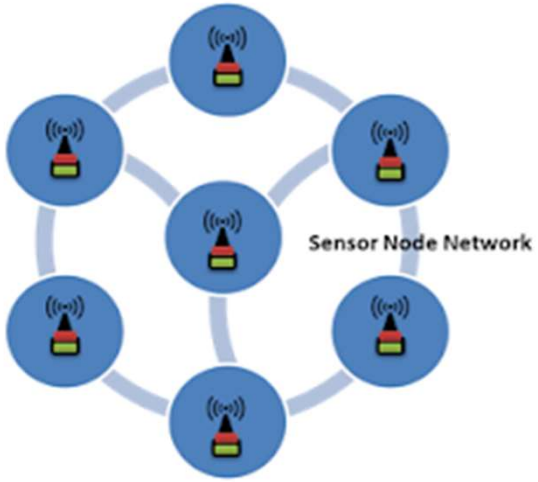
$$f_i(x) = \sum_{d \in \mathcal{D}_i} \text{Error}(x, d)$$

- Local copy of x

Motivations Big data – Privacy – Speed Up

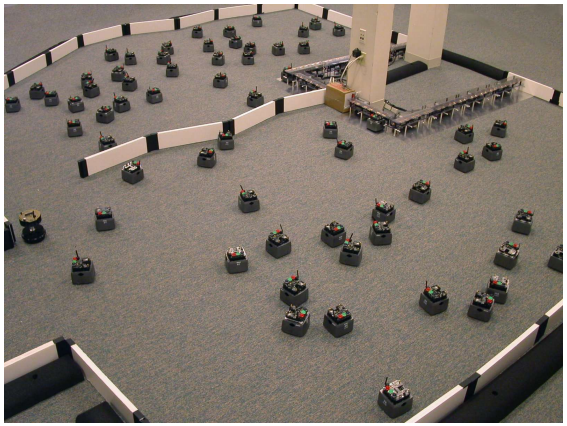
Other applications

Sensor Network



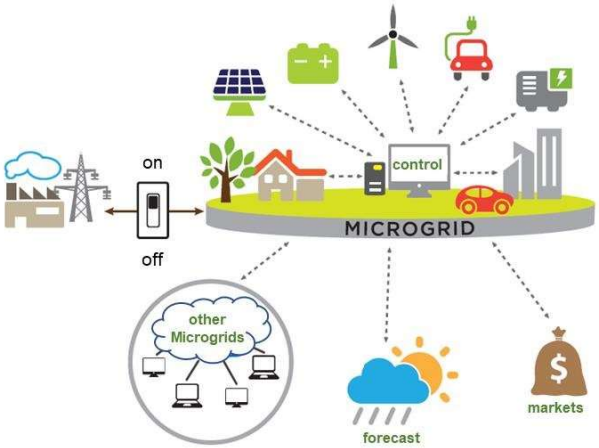
Informatics–Wireless sensor network

Multi-robot systems



Multi-Robot Systems Engineering
MIT , James McLurkin

Micro-Grid



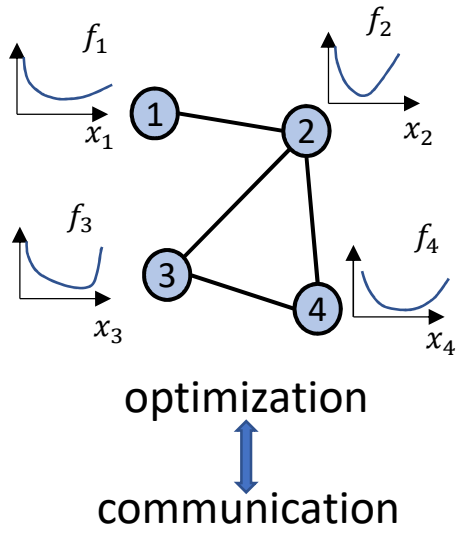
ResearchGate, Planning and implementation of bankable microgrids, Michael Stadler

Decentralized Optimization

Many challenges for better methods

BUT

Analysis highly complex

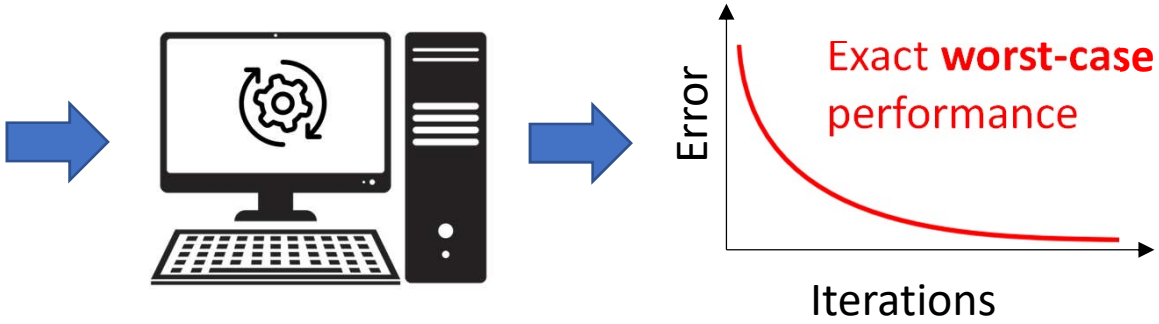


- Performance bounds: complex and **conservative**
- Difficult algorithms comparisons
- Difficult parameters tuning

🎯 Objective

```
function MyDecentralizedAlgo()
    N = 10;          % number of agents
    x0 = init(N);  % initial points
    x = x0;

    for k=1:niter
        % any local computations
        % any local communications
        x = update(x,N);
    end
end
```



Impact for decentralized optimization

- Access to **accurate performance** of methods
- Easier **comparison and tuning** of algorithms
- **Rapid exploration** of new algorithms.

Outline of the talk

- Performance Estimation Problem (**PEP**)
- **PEP** for **decentralized** optimization
- **Analysis** of Decentralized Algorithms

Performance Estimation Problem (PEP)

Idea: Worst-cases are solutions to optimization problems

$$\max_{f,\, x^0, \ldots, x^K} \quad \mathrm{perf}(f, x^0, \ldots, x^K) \overset{e.g.}{=} f(x^K) - f(x^*)$$

With	f	$\in$	class of functions
	x^0		initial condition
	x^k		from the algorithm analyzed

Performance Estimation Problem (PEP)

Idea: Worst-cases are solutions to optimization problems

$$\max_{f,\, x^0, \dots, x^K} \quad \mathrm{perf}(f, x^0, \dots, x^K) \overset{e.g.}{=} f(x^K) - f(x^*)$$

With	$f \in \mathcal{F}_{\mu,L}$	class of functions
	$\|x^0 - x^*\| \leq R$	initial condition
	$x^k = \mathrm{x}^{k-1} - \alpha \nabla f(x^{k-1})$	from the algorithm analyzed

---

Original idea by Yoel Drori, and Marc Teboulle (2014)

Further developments by Adrien B. Taylor, Julien M. Hendrickx, and François Glineur (2017)

Performance Estimation Problem (PEP)

Idea: Worst-cases are solutions to optimization problems

$$\max_{f,\, x^0, \dots, x^K} \quad \mathrm{perf}(f, x^0, \dots, x^K) \overset{e.g.}{=} f(x^K) - f(x^*)$$

With	$f \in \mathcal{F}_{\mu,L}$	class of functions
	$\|x^0 - x^*\| \leq R$	initial condition
	$x^k = x^{k-1} - \alpha \nabla f(x^{k-1})$	from the algorithm analyzed

Infinite-Dimensional problem

PEP can be solved exactly for a wide class of **centralized** first-order algorithms.

Existing toolbox (PESTO)

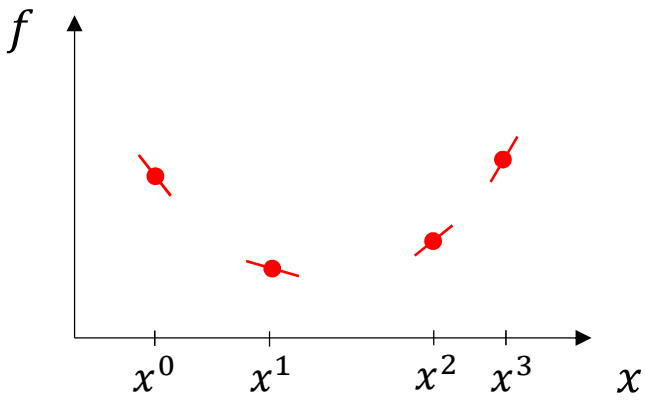
[Taylor et al. 2017]

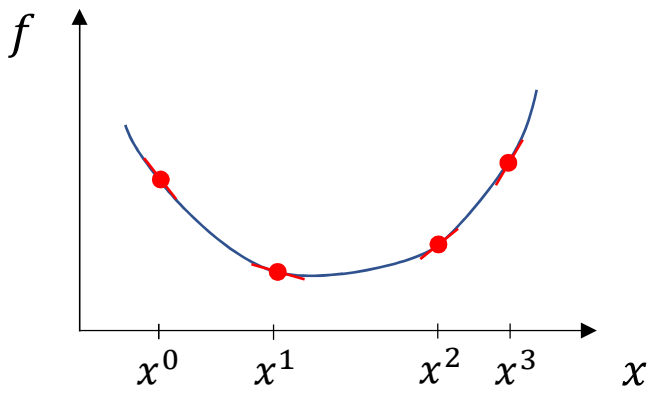
Can be used for analysis, tuning, design and proofs.

Finite dimensional PEP

Finite dimension $\{x^k, g^k, f^k\}_{k=1\ldots K}$

Interpolation conditions on $\{x^k, g^k, f^k\}_{k=1\ldots K}$ $\Leftrightarrow$ There is $f \in \mathcal{F}$ s.t. $f^k = f(x^k)$ and $g^k = \nabla f(x^k)$





Interpolation conditions for many classical function classes

[Taylor et al. 2017]

e.g., for $\mathcal{F}_\mu$, $f_j \geq f_i + g_i^T(x_j - x_i) + \frac{\mu}{2}\|x_j - x_i\|^2$ for all (i,j)

SDP formulation of PEP

PEP constraints may be quadratic and non-convex

SDP reformulation

Variables $F = [f^0 \ \dots \ f^K]$

$G = P^T P$ $\quad P = [x^0 \dots \ x^K \ g^0 \dots g^K]$

Gram Matrix

PEP $\max_{F,G} \ \mathrm{perf}(\mathrm{F}, \mathrm{G})$

With $G \succcurlyeq 0$

Interpolation

Initial

Algorithm

constraints linear in G and F

Efficient resolution

Outline of the talk

- Performance Estimation Problem (**PEP**)
- **PEP** for **decentralized** optimization
- **Analysis** of Decentralized Algorithms

Performance Estimation Problem (PEP)

Idea: Worst-cases are solutions to optimization problems

$$\max_{\substack{f_i,\, x_i^0,\dots,x_i^K,\, W \\ (i=1\dots N)}} \mathrm{perf}(f_i,\, x_i^0, \dots, x_i^K)$$

With	f_i	$\in$	class of functions
	x_i^0		initial condition
	x_i^k		from the algorithm analyzed
	W	$\in$	class of network matrices

How to represent a class of
communication network matrices *?*

PEP for DGD: network given a priori

$$\max_{\substack{f_i,\, x_i^0,\dots,x_i^K,\, W \\ y_i^0,\dots,y_i^{K-1}}} \quad \mathrm{perf}(f_i,\, x_i^0,\dots,x_i^K)$$

With $f_i \in$ class of functions

x^0 initial condition

W **given network matrix**

Iterates from DGD
$$\begin{cases} y_i^k = \sum_j w_{ij}\, x_j^k \\ x_i^{k+1} = y_i^k - \alpha \nabla f_i(x_i^k) \end{cases}$$

For all $i = 1 \dots N$,
For all $k = 0 \dots K-1$

Exact Formulation

PEP for DGD: class of networks

$$\max_{\substack{f_i,\, x_i^0,\dots,x_i^K,W \\ y_i^0,\dots,y_i^{K-1}}} \quad \mathrm{perf}(f_i,\, x_i^0,\dots,x_i^K)$$

With $f_i \in$ class of functions

x^0 initial condition

$\boldsymbol{W}$ **Any symmetric doubly stochastic matrix with given range of eigenvalues $[\lambda^-, \lambda^+]$**

Find constraints between y_i^k and x_i^k

Iterates from DGD
$$\begin{cases} y_i^k = \sum_j \overset{?}{w_{ij}} x_j^k \\ x_i^{k+1} = y_i^k - \alpha \nabla f_i(x_i^k) \end{cases}$$

For all $i = 1 \dots N$,
For all $k = 0 \dots K-1$

PEP for DGD: class of networks

$$\max_{\substack{f_i,\, x_i^0,\dots,x_i^K,\cancel{W} \\ y_i^0,\dots,y_i^{K-1}}} \quad \mathrm{perf}(f_i,\, x_i^0,\dots,x_i^K)$$

With $f_i \in$ class of functions

x^0 initial condition

$\cancel{W}$ ~~Any symmetric doubly stochastic matrix with given range of eigenvalues $[\lambda^-, \lambda^+]$~~

Find constraints between y_i^k and x_i^k

Iterates from DGD
$$\begin{cases} y_i^k = \cancel{\sum_j w_{ij}}^{\,?}\, x_j^k \\ x_i^{k+1} = y_i^k - \alpha \nabla f_i(x_i^k) \end{cases}$$

For all $i = 1 \dots N$,
For all $k = 0 \dots K-1$

Consensus steps in PEP

- Search Space for X and Y

(C1) $$y_i^k = \sum_{j=1}^{N} w_{ij}\, x_j^k$$

For each agent $i = 1 \dots N$,
For each consensus step $k = 0 \dots K-1$

compact notation $Y = WX$ with $Y_{ik} = y_i^k,\ X_{ik} = x_i^k.$

(C2) $W = [w_{ij}]$ is a symmetric and doubly-stochastic matrix with a given range of eigenvalues $[\lambda^-, \lambda^+]$

- **Necessary constraints** for describing (C1) and (C2)

$\bar{X}, \bar{Y}$: agents average vectors

$X_\perp, Y_\perp$: centered matrices

$$X_\perp = X - \mathbf{1}\bar{X}^T,\ \ Y_\perp = Y - \mathbf{1}\bar{Y}^{\mathrm{T}}$$

$$\bar{X} = \bar{Y} \qquad (1)$$

$$\lambda^- X_\perp^T X_\perp \preccurlyeq X_\perp^T Y_\perp \preccurlyeq \lambda^+ X_\perp^T X_\perp \qquad (2)$$

$$(Y_\perp - \lambda^- X_\perp)^T (Y_\perp - \lambda^+ X_\perp) \preccurlyeq 0 \qquad (3)$$

Simplification of (2) and (3) when $-\lambda^- = \lambda^+ = \lambda$: $Y_\perp^T Y_\perp \preccurlyeq \lambda^2 X_\perp^T X_\perp$

Consensus steps in PEP

Summary of the constraints for **consensus steps** $\boldsymbol{Y = WX}$

$$\bar{X} = \bar{Y} \tag{1}$$

$$\lambda^{-} X_{\perp}^{T} X_{\perp} \preccurlyeq X_{\perp}^{T} Y_{\perp} \preccurlyeq \lambda^{+} X_{\perp}^{T} X_{\perp} \tag{2}$$

$$(Y_{\perp} - \lambda^{-} X_{\perp})^{T} (Y_{\perp} - \lambda^{+} X_{\perp}) \preccurlyeq 0 \tag{3}$$

Advantages of our constraints

- ✓ **Independent** of the algorithm
- ✓ Link different consensus steps that **use the same matrix**
- ✓ Can be incorporated into SDP formulation of PEP, which can be **solved efficiently**

PEP for DGD: class of networks

$$\max_{\substack{f_i,\, x^0,\dots,x^K,\, \cancel{W} \\ y^0,\dots,y^{K-1}}} \quad \mathrm{perf}(f_i,\, x^0,\dots,x^K)$$

With $f_i \in$ class of functions

x^0 initial condition

~~W Any symmetric doubly stochastic matrix with given range of eigenvalues $[\lambda^-, \lambda^+]$~~

Iterates from DGD

$$\begin{cases} y_i^k = \cancel{\sum_j w_{ij}\, x_j^k} \\ x_i^{k+1} = y_i^k - \alpha \nabla f_i(x_i^k) \end{cases}$$

For all $i = 1 \dots N$,
For all $k = 0 \dots K-1$

PEP for DGD: Spectral formulation

(Relaxation)

$$\max_{\substack{f_i,\, x^0,\dots,x^K \\ y^0,\dots,y^{K-1}}} \quad \mathrm{perf}(f_i, x^0, \dots, x^K)$$

With $f_i \in$ class of functions

x^0 initial condition

Iterates from DGD $\left\{ x_i^{k+1} = y_i^k - \alpha \nabla f_i(x_i^k) \right.$ For all $i = 1 \dots N$, For all $k = 0 \dots K-1$

Consensus steps

$Y = WX$

W symmetric doubly stochastic $\lambda(W) \in [\lambda^-, \lambda^+]$

$$\left\{ \boxed{\begin{aligned} \bar{X} &= \bar{Y} \\ \lambda^- X_\perp^T X_\perp \preccurlyeq X_\perp^T Y_\perp &\preccurlyeq \lambda^+ X_\perp^T X_\perp \\ (Y_\perp - \lambda^- X_\perp)^T (Y_\perp - \lambda^+ X_\perp) &\preccurlyeq 0 \end{aligned}} \right.$$

Notations

$X_{ik} = x_i^k$

$Y_{ik} = y_i^k$

Upper bounds for the worst-case performance of DGD

Our tool for automatic performance estimation

Apply to any decentralized method using consensus $y = Wx$

Agent-dependent formulations

- Exact formulation

  exact worst-case performance,
  specific to a given matrix W

- Spectral formulation

  upper bound on worst-case performance,
  valid for an entire **spectral class** of network matrices

Available in PESTO

Agent-independent (spectral) formulation

Agent-Independent formulation

Local view

$$\boxed{\begin{aligned} &\min_{x_1,\dots,x_N} \sum_{i=1}^{N} f_i(x_i) \\ &\text{s.t. } x_i = x_j \quad \forall\, (i,j) \text{ neighbors} \end{aligned}}$$

$x_i \in \mathbb{R}^d,$

$f_i\colon \mathbb{R}^d \to \mathbb{R}$

$i = 1 \dots N$

Global view

$$\boxed{\begin{aligned} &\min_{\mathbf{x}} \quad F(\mathbf{x}) \\ &\text{s.t. } \mathbf{x} \in \mathrm{C} \end{aligned}}$$

$\mathbf{x} \in \mathbb{R}^{Nd}, \qquad \mathrm{F}\colon \mathbb{R}^{Nd} \to \mathbb{R}$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix} \in C \Leftrightarrow x_1 = \cdots = x_N$$

➡ **Change of variable** to decouple C and $C^{\perp}$

$${}_b\mathbf{x} = \begin{bmatrix} \bar{x} \\ x_{\perp} \end{bmatrix} \quad \text{with} \quad \bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \in \mathbb{R}^d$$

$$x_{\perp} \in \mathbb{R}^{(N-1)d}$$

Outline of the talk

- Performance Estimation Problem (**PEP**)
- **PEP** for **decentralized** optimization
- **Analysis** of Decentralized Algorithms

Results of PEP for DGD

Problem $\min_x f(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x)$ with optimal solution x^*

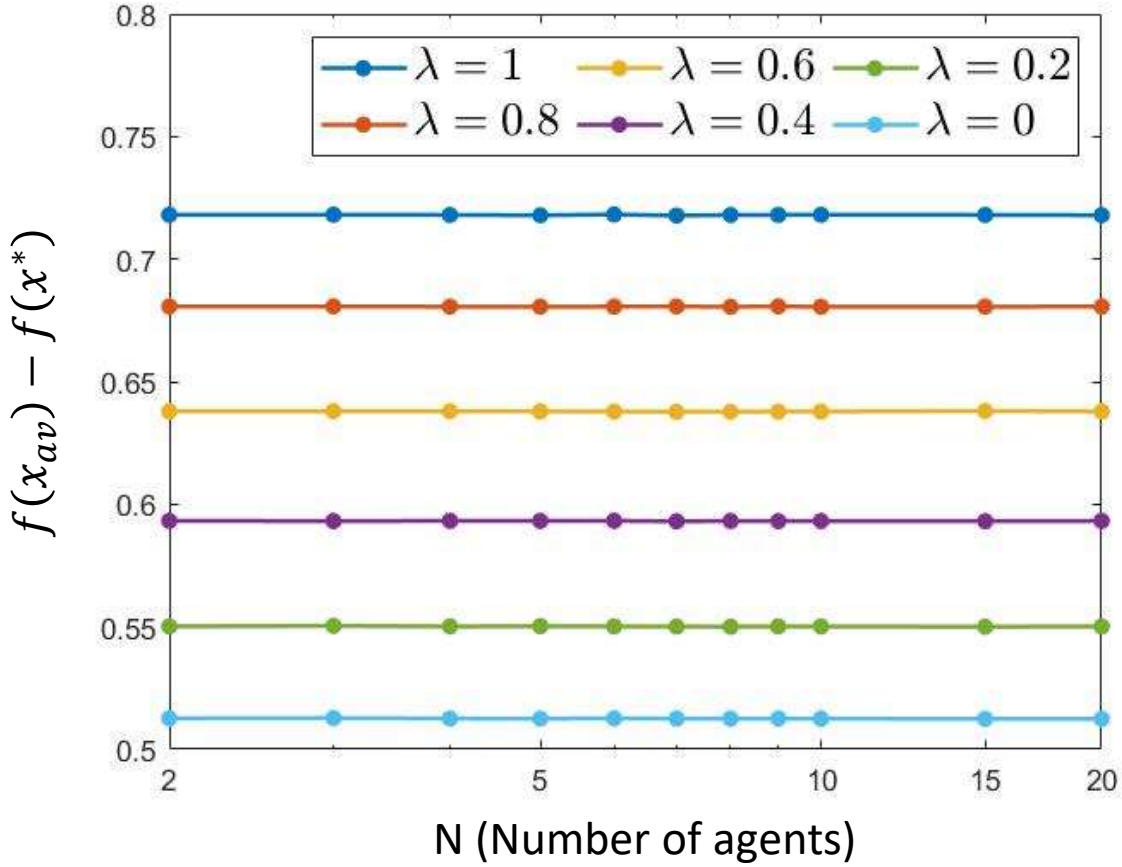
DGD Algorithm $x_i^{k+1} = \sum_{j=1}^{N} w_{ij}\, x_j^k - \alpha \nabla f_i(x_i^k)$

Settings K steps of DGD with

- Constant step-size: $\alpha = \frac{1}{\sqrt{K}}$
- Convex local functions f_i with bounded subgradients
- Identical starting points s.t. $\|x^0 - x^*\|^2 \leq 1$
- Symmetric doubly-stochastic network matrix W
  s.t. $\lambda(W) \in [-\lambda, \lambda]$ (except for $\lambda_1(W) = 1$)

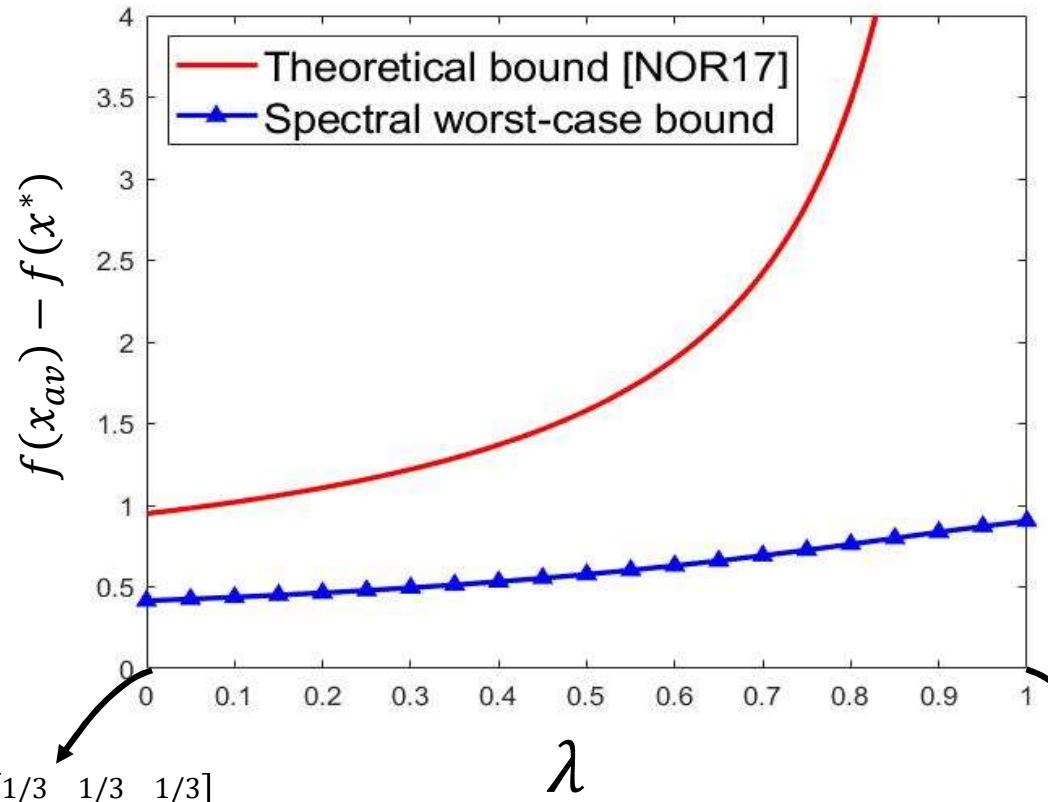
Performance criterion: $f(x_{av}) - f(x^*)$ where $x_{av} = \frac{1}{K}\sum_k \frac{1}{N}\sum_i x_i^k$

DGD – Spectral worst-case evolution with N



For $K = 5$ iterations and $\lambda(W) \in [-\lambda, \lambda]$

DGD – Spectral worst-case vs Theoretical bound



$$\begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix}$$

complete

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

disconnected

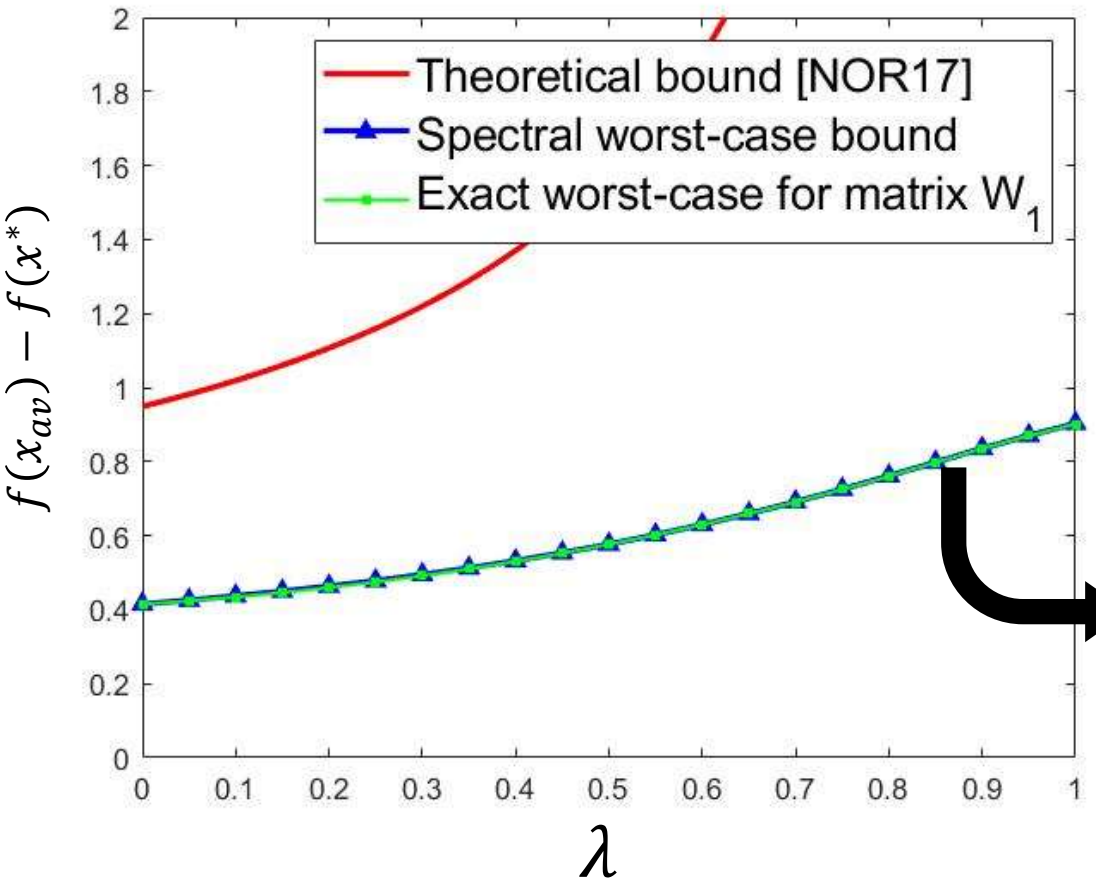
[NOR17] A. Nedic, A. Olshevsky, and M. G. Rabbat, "Network topology and communication computation tradeoffs in decentralized optimization", 2017.

Symmetric range of eigenvalues

$$-\lambda \leq \lambda_n(W) \leq \cdots \leq \lambda_2(W) \leq \lambda$$

For $K = 10$ iterations, $N = 3$ agents and $\lambda(W) \in [-\lambda, \lambda]$

Tightness Analysis



Worst-case matrix estimation

$$W_1 = YX^+$$

$$W_1 = \begin{bmatrix} d & c & c \\ c & d & c \\ c & c & d \end{bmatrix}$$

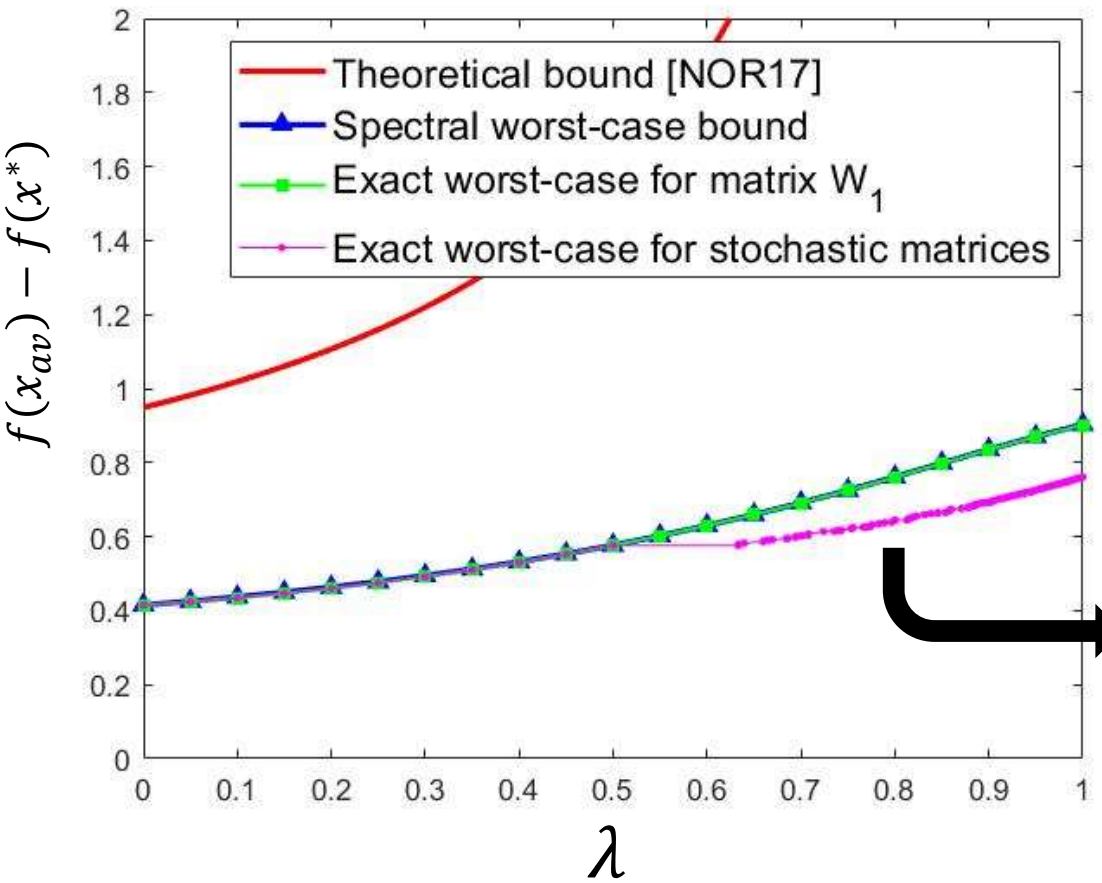
with $c = \frac{1+\lambda}{3}$, $d = 1 - 2c$

➡ $\lambda(W_1) = \{1, -\lambda, -\lambda\}$

EXACT for *spectrally* doubly-stochastic matrices

For $K = 10$ iterations, $N = 3$ agents and $\lambda(W) \in [-\lambda, \lambda]$

Tightness Analysis



Worst-case matrix estimation

$$W_1 = YX^+$$

For any N:

$$W_1 = \Pi + \lambda(\Pi - I)$$

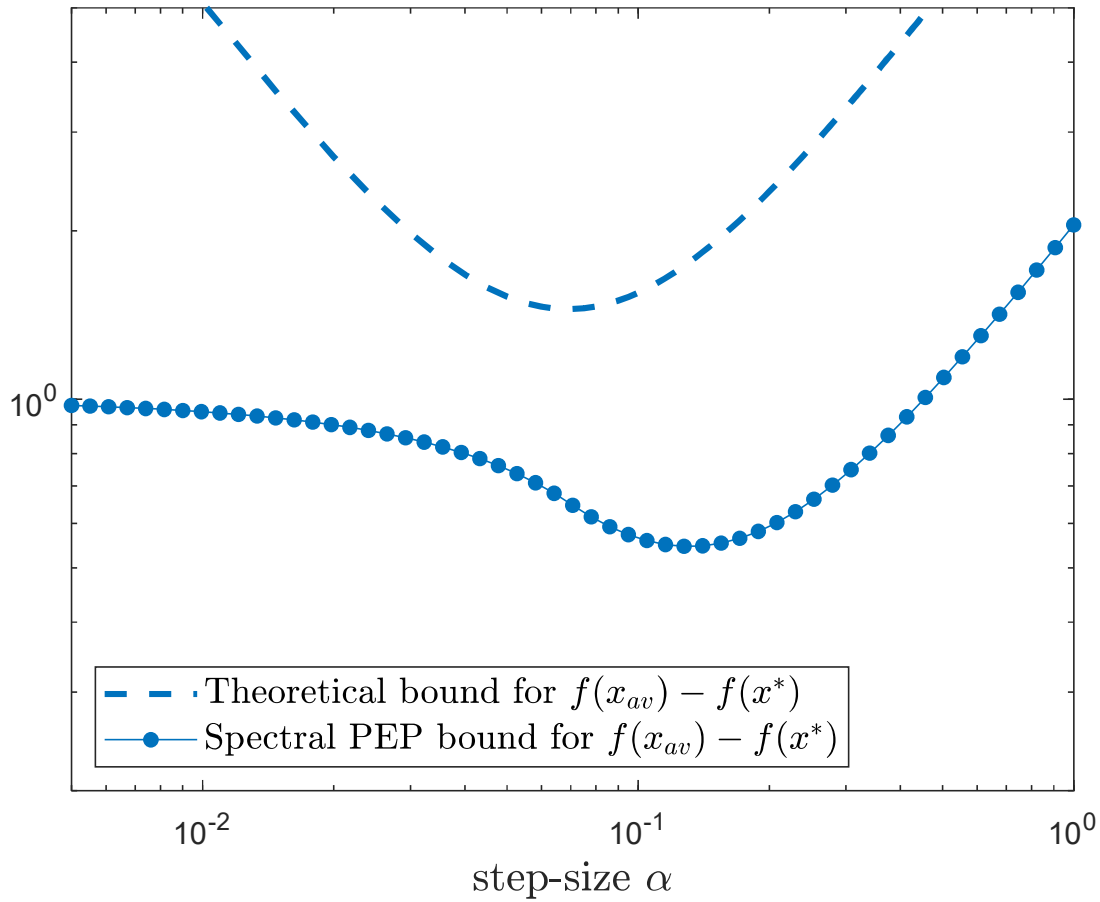
with $\Pi = \frac{1}{N}\mathbf{1}\mathbf{1}^T$

➡ $\lambda(W_1) = \{1, -\lambda, -\lambda\}$

CLOSE for doubly-stochastic matrices

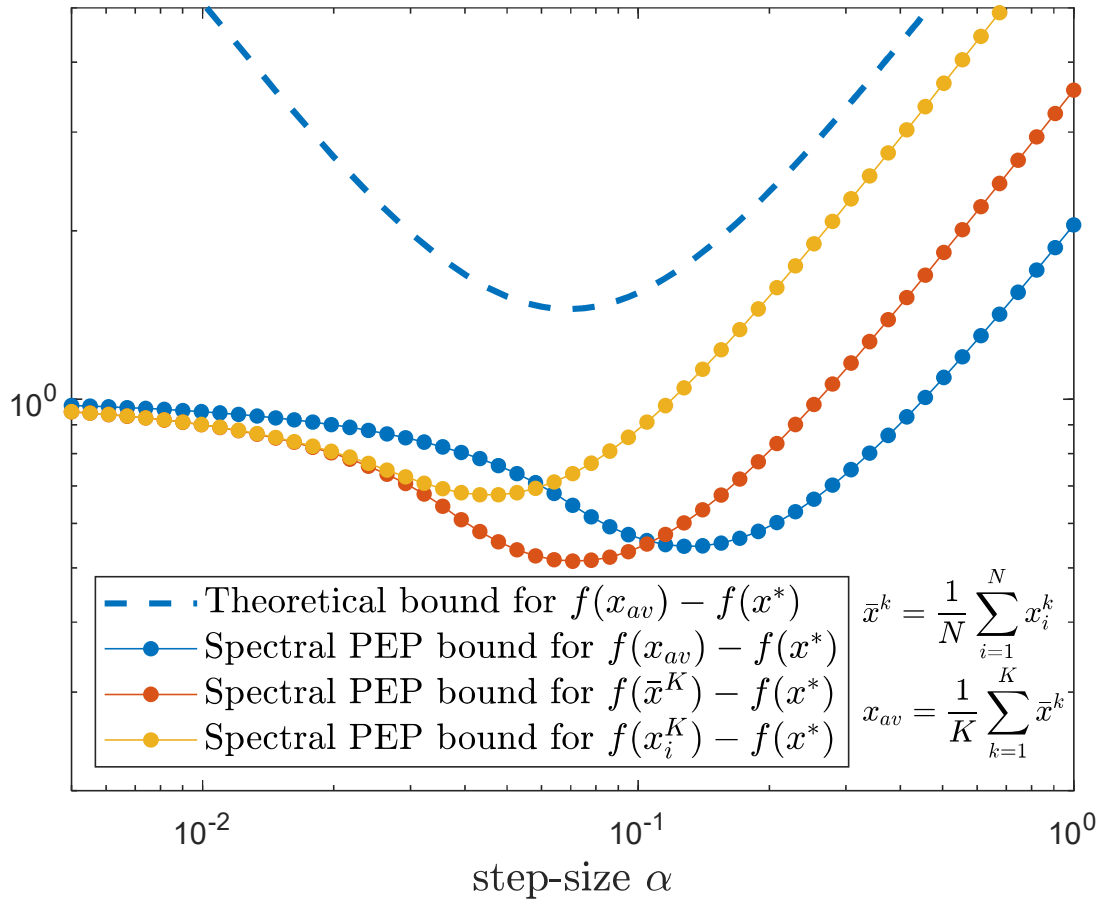
For $K = 10$ iterations, $N = 3$ agents and $\lambda(W) \in [-\lambda, \lambda]$

DGD – Spectral worst-case evolution with alpha



For $K = 10$ iterations and $\lambda(W) \in [-\lambda, \lambda]$

DGD – Spectral worst-case evolution with alpha



For $K = 10$ iterations and $\lambda(W) \in [-\lambda, \lambda]$

Results of PEP for DIGing

Problem

$$\min_x f(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x)$$ with optimal solution x^*

DIGing Algorithm

gradient tracking technique

$$x_i^{k+1} = \sum_{j=1}^{N} w_{ij}\, x_j^k - \alpha s_i^k$$

$$s_i^{k+1} = \sum_{j=1}^{N} w_{ij}\, s_j^k + \nabla f_i\left(x_i^{k+1}\right) - \nabla f_i\left(x_i^k\right)$$

Settings

- f_i are L-smooth and μ-strongly convex
- Initial: $\frac{1}{N}\sum_{i=1}^{N}\left\|x_i^0 - x^*\right\|^2 \le 1$ and $\frac{1}{N}\sum_{i=1}^{N}\left\|s_i^0 - \overline{\nabla f_i^0}\right\|^2 \le 1$
- Symmetric doubly-stochastic network matrix W
  s.t. $\lambda(W) \in [-\lambda, \lambda]$ (except for $\lambda_1(W) = 1$)

Performance criterion: $\frac{1}{N}\sum_{i=1}^{N}\left\|x_i^K - x^*\right\|^2$

Results of PEP for DIGing

Spectral PEP formulation

- **Independent** of the number of agents N

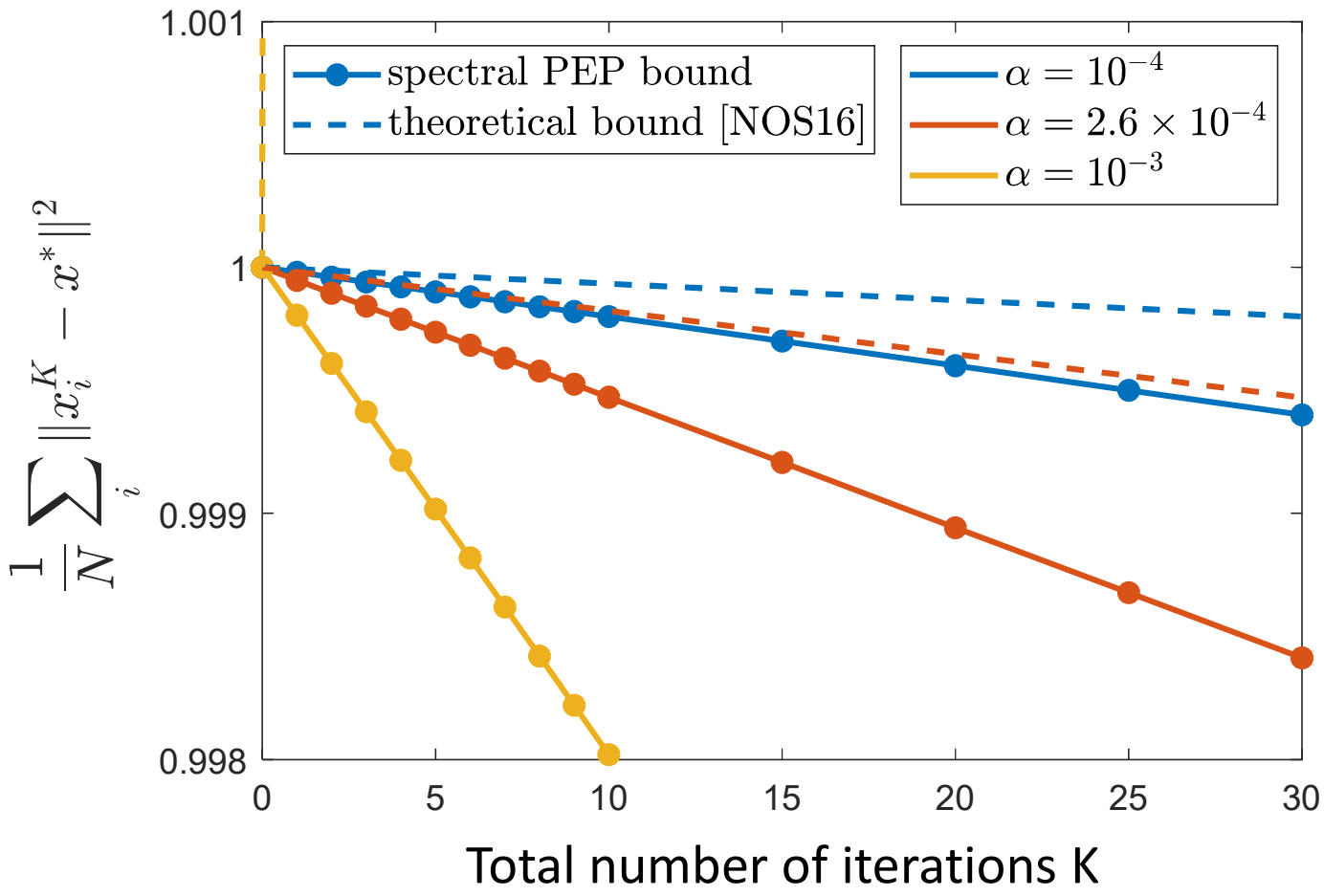
- Same worst-case matrix than DGD

$$W_1 = \Pi + \lambda(\Pi - I) \text{ with } \Pi = \frac{1}{N}\mathbf{1}\mathbf{1}^T$$

➡ $\lambda(W_1) = \{1, -\lambda, \dots, -\lambda\}$

- **Exact** for spectrally doubly-stochastic matrices

Results of PEP for DIGing

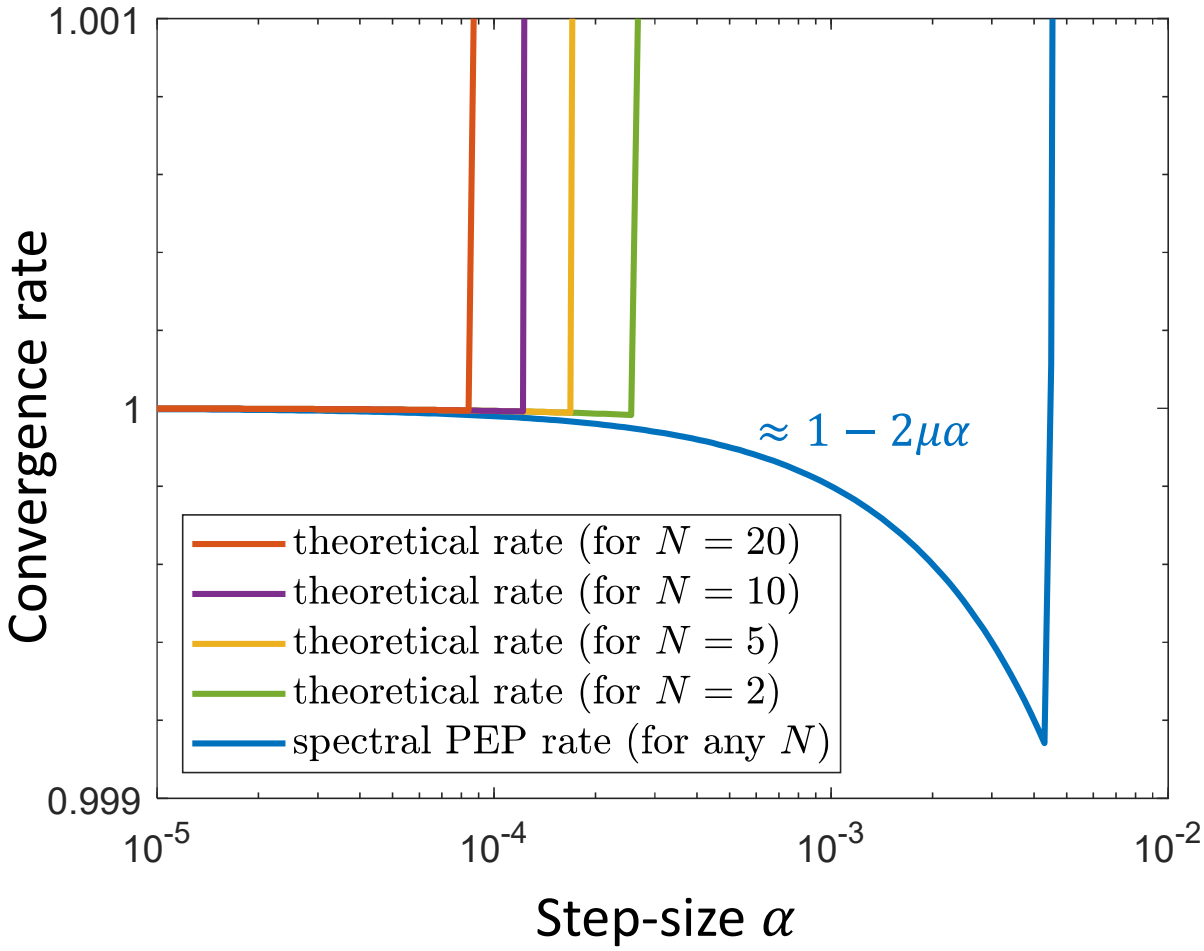


For $\mu = 0.1,\ L = 1$ and $\lambda(W) \in [-0.9, 0.9]$

Computed for $N = 2$.

[NOS16] A. Nedic, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," SIAM Journal on Optimization, 2016.

Results of PEP for DIGing

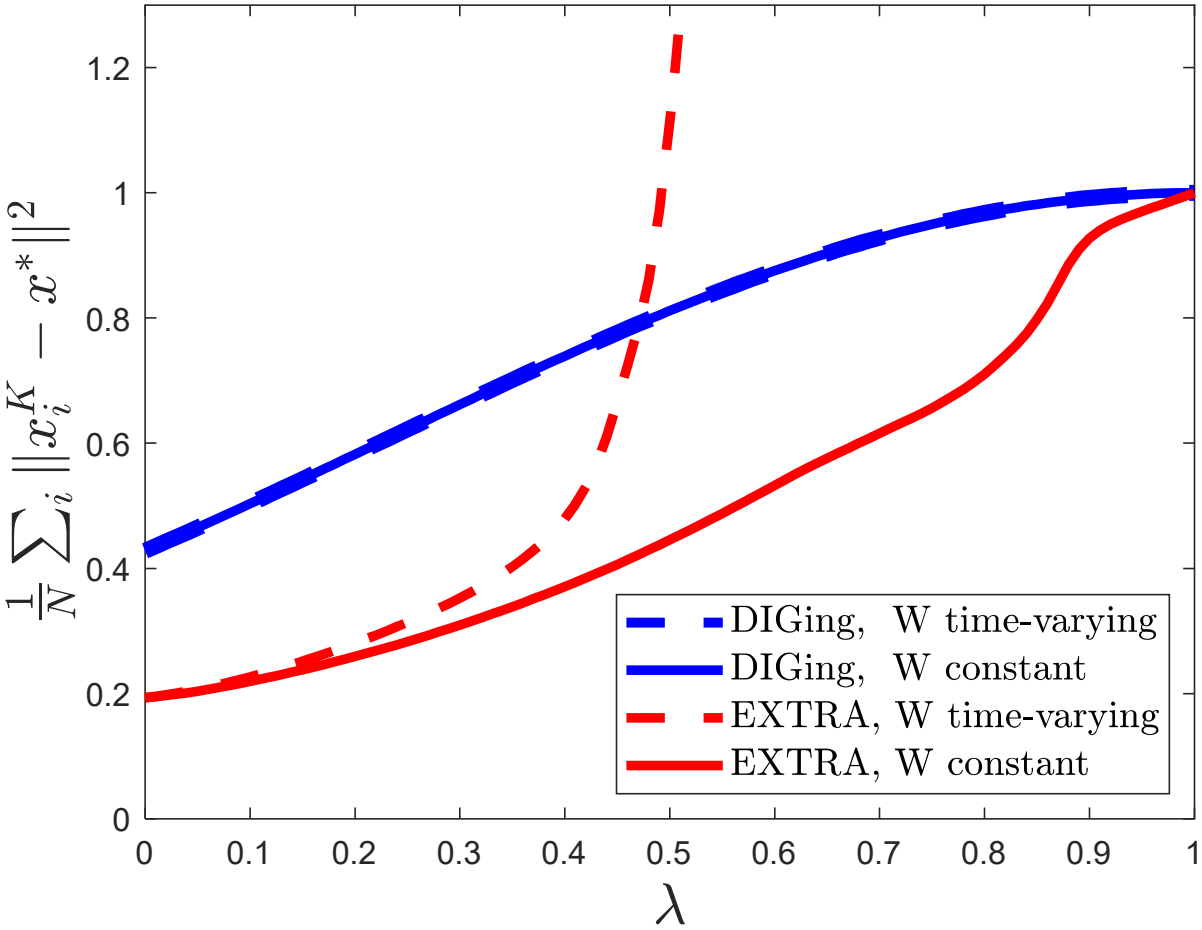


For $\mu = 0.1,\ L = 1$ and $\lambda(W) \in [-0.9, 0.9]$

Computed for $N = 2$.

[NOS16] A. Nedic, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," SIAM Journal on Optimization, 2016.

Algorithms comparison



For $K = 10$ iterations, $\mu = 0.1$, $L = 1$ and $\lambda(W) \in [-\lambda, \lambda]$.

Conclusion

Sébastien Colla

Toolbox PESTO

Numerical tool for **automatic performance computation** of decentralized optimization methods

PEP idea: worst-cases are solutions of optimization problems

SPECTRAL formulation	EXACT formulation
Spectral class of matrices	Given network matrix W
Relaxation of PEP	ALWAYS exact

For DGD and DIGing:
- ✓ Independent of N
- ✓ Tight when negative weights are allowed
- ✓ Improve on the literature bounds

We can answer a large diversity of (new) questions

Future works

- ❑ Other class of networks (any suggestion?)
- ❑ Agent-independent PEP formulation

References

[CH21] S. Colla, J. M. Hendrickx, “Automatic Performance Estimation for Decentralized Optimization”, preprint 2022.

[Taylor et al.] A. B. Taylor, J. M. Hendrickx, F. Glineur, “Exact worst-case performance of first-order methods for composite convex optimization,” SIAM Journal on Optimization, 2015

[NOR17] A. Nedic, A. Olshevsky, and M. G. Rabbat, “Network topology and communication computation tradeoffs in decentralized optimization”, 2017.

[NOS16] A. Nedic, A. Olshevsky, and W. Shi, “Achieving geometric convergence for distributed optimization over time-varying graphs,” SIAM Journal on Optimization, 2016.