Differentially Private Distributed Optimal Power Flow

Vladimir Dvorkin[†‡], Pascal Van Hentenryck[‡],
Jalal Kazempour[†], Pierre Pinson[†]

[†]Technical University of Denmark
[‡]Georgia Institute of Technology

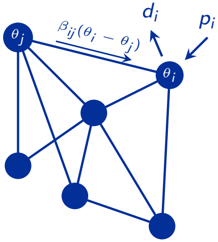
vladvo@elektro.dtu.dk

59th IEEE Conference on Decision and Control
December 2020

Background

- Growing digitalization of modern power systems boost the efficiency of operations
  - Operations are guided by the solution of the Optimal Power Flow (OPF) problem
  - System operators collect large amounts of power system *data* ...
  - ... and produce efficient generator set points
- OPF input datasets contains private information:
  - Power network parameters
  - Load profiles of network users
  - Generation and market parameters
- Distributed OPF computations to limit information exchange and preserve privacy [Molzahn et al., 2017]

Optimal power flow (OPF) problem

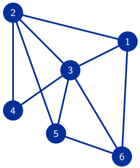


- Optimizes power systems at minimum cost while respecting system constraints
- We consider DC approximation of power flows

$$
\begin{aligned}
\min_{p,\theta} \quad & c(p) && \text{generation cost} \\
\text{s.t.} \quad & B\theta = p - d, && \text{nodal power balance} \\
& \theta \in \mathcal{F}, p \in \mathcal{P}, && \text{flow \& generation limit}
\end{aligned}
$$

- Central optimization requires all agents to share their data
- Solution? Distribute OPF computation [Conejo and Aguado, 1998, Biskas et al., 2005]

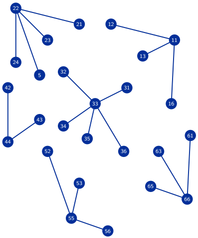
Distributed OPF computations



- Decompose network per node(s) ...
- ... by duplicating voltage angles
- ... and enforce consensus constraints

$$\begin{aligned} \min_{p,\theta} \quad & c(p) \\ \text{s.t.} \quad & B\theta = p - d \\ & \theta \in \mathcal{F}, p \in \mathcal{P} \end{aligned}$$

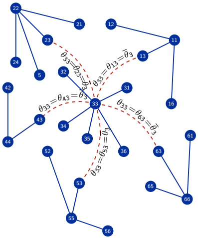
Distributed OPF computations



- Decompose network per node(s) ...
- ... by duplicating voltage angles
- ... and enforce consensus constraints

$$
\begin{aligned}
\min_{p,\theta} \quad & \sum_{i=1}^{N} c_i(p_i) \\
\text{s.t.} \quad & B_i^\top \theta_i = p_i - d_i, \quad \forall i = \{1, \ldots, N\} \\
& \theta_i \in \mathcal{F}_i, p_i \in \mathcal{P}_i, \quad \forall i = \{1, \ldots, N\}
\end{aligned}
$$

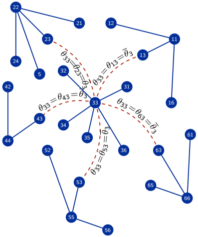
Distributed OPF computations



- Decompose network per node(s) ...
- ... by duplicating voltage angles
- ... and enforce consensus constraints

$$
\begin{aligned}
\min_{p,\theta,\overline{\theta}} \quad & \sum_{i=1}^{N} c_i(p_i) \\
\text{s.t.} \quad & B_i^\top \theta_i = p_i - d_i, \quad \forall i = \{1, \ldots, N\} \\
& \theta_i \in \mathcal{F}_i, p_i \in \mathcal{P}_i, \quad \forall i = \{1, \ldots, N\} \\
& \theta_i = \overline{\theta} : {\color{red}\mu_i}, \quad \forall i = \{1, \ldots, N\}
\end{aligned}
$$

Distributed OPF computations

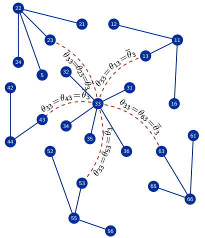


- Decompose network per node(s) ...
- ... by duplicating voltage angles
- ... and enforce consensus constraints

$$\max_{\mu_i} \min_{p,\theta,\overline{\theta}} \quad \overbrace{\sum_{i=1}^{N} c_i(p_i) + \underbrace{\sum_{i=1}^{N} \mu_i^\top (\theta_i - \overline{\theta})}_{\text{dualized consensus}} + \underbrace{\sum_{i=1}^{N} \frac{\rho}{2} \|\theta_i - \overline{\theta}\|_2^2}_{\text{regularization term}}}^{\text{Augmented Lagrangian: } \sum_{i=1}^{N} \mathcal{L}(p_i, \theta_i, \overline{\theta}, \mu_i)}$$

$$\begin{aligned} \text{s.t.} \quad & B_i^\top \theta_i = p_i - d_i, \quad \forall i = \{1, \dots, N\} \\ & \theta_i \in \mathcal{F}_i, p_i \in \mathcal{P}_i, \quad \forall i = \{1, \dots, N\} \end{aligned}$$

Distributed OPF computations



- Decompose network per node(s) ...
- ... by duplicating voltage angles
- ... and enforce consensus constraints

$$\max_{\mu_i} \min_{p,\theta,\overline{\theta}} \overbrace{\sum_{i=1}^{N} c_i(p_i) + \underbrace{\sum_{i=1}^{N} \mu_i^\top (\theta_i - \overline{\theta})}_{\text{dualized consensus}} + \underbrace{\sum_{i=1}^{N} \frac{\rho}{2} \|\theta_i - \overline{\theta}\|_2^2}_{\text{regularization term}}}^{\text{Augmented Lagrangian: } \sum_{i=1}^{N} \mathcal{L}(p_i,\theta_i,\overline{\theta},\mu_i)}$$

$$\text{s.t.} \quad B_i^\top \theta_i = p_i - d_i, \quad \forall i = \{1, \ldots, N\}$$

$$\theta_i \in \mathcal{F}_i, p_i \in \mathcal{P}_i, \quad \forall i = \{1, \ldots, N\}$$

Distributed ADMM algorithm [Boyd et al., 2011]:

1. Update θ_i for fixed $\overline{\theta}$ and μ_i:

$$\theta_i \leftarrow \underset{p_i,\theta_i \in \mathcal{O}_i}{\text{argmin}} \ \mathcal{L}(p_i, \theta_i, \overline{\theta}, \mu_i)$$

2. Update $\overline{\theta}$ for fixed θ_i and μ_i

$$\overline{\theta} \leftarrow \underset{\overline{\theta}}{\text{argmin}} \ \mathcal{L}(\theta_i, \overline{\theta}, \mu_i)$$

3. Update μ_i for fixed θ_i and $\overline{\theta}$

$$\mu_i \leftarrow \mu_i + \rho(\theta_i - \overline{\theta})$$

Does ADMM always preserve privacy of local OPF datasets?

Privacy attack model for distributed OPF

OPF sub-problem

$\bar{\theta} \longrightarrow$

$\mu_i \longrightarrow$

$$\begin{aligned} \min_{p_i,\theta_i} \quad & c_i(p_i) + {\mu_i}^\top \theta_i + \frac{\rho}{2}\|\theta_i - \bar{\theta}\|_2^2 \\ \text{s.t.} \quad & B_i^\top \theta_i = p_i - d_i, \\ & \theta_i \in \mathcal{F}_i, p_i \in \mathcal{P}_i \end{aligned}$$

$\longrightarrow \theta_i$

- Coordination signals
- Optimization variables
- Side information
- Private data

Privacy attack model for distributed OPF

OPF sub-problem

$$\overline{\theta},\ \mu_i \longrightarrow \boxed{\begin{aligned} \min_{p_i,\theta_i} \quad & c_i(p_i) + \mu_i^\top \theta_i + \frac{\rho}{2}\|\theta_i - \overline{\theta}\|_2^2 \\ \text{s.t.} \quad & B_i^\top \theta_i = p_i - d_i, \\ & \theta_i \in \mathcal{F}_i, p_i \in \mathcal{P}_i \end{aligned}} \longrightarrow \theta_i$$

- Coordination signals
- Optimization variables
- Side information
- Private data

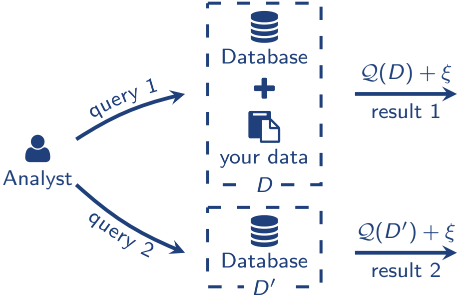
- The goal of privacy attack is to reconstruct the unknown data item
- Assume the side information and optimization structure are known
- Reconstruction of the unknown data item through optimization:

$$\begin{aligned} \min_{p_i,\theta_i,d_i \geqslant 0} \quad & c_i(p_i) + \mu_i^\top \theta_i + \frac{\rho}{2}\|\theta_i - \overline{\theta}\|_2^2 + \underbrace{\Upsilon\|\theta_i - \theta_i\|_2^2}_{\text{penalty term}} \\ \text{s.t.} \quad & B_i^\top \theta_i = p_i - d_i, \\ & \theta_i \in \mathcal{F}_i, p_i \in \mathcal{P}_i \end{aligned}$$

- Unknown d_i is optimized to replicate the OPF sub-problem response, i.e., $\|\theta_i - \theta_i\|_2^2 = 0$
- Refer to the extended arXiv paper for non-optimization models of privacy attacks

Formal privacy guarantees for distributed OPF

Differential privacy (definition)



- $\mathcal{Q}$ is a query computed on a dataset
- ξ is a carefully calibrated noise
- θ and θ' are stat. indistinguishable
- By observing θ or θ', analyst can't tell if your data is included

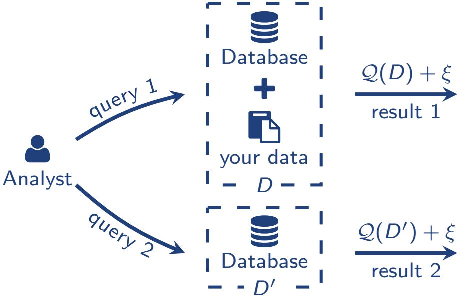
ε-differential privacy [Dwork et al., 2014]

A randomized query $\tilde{\mathcal{Q}} : \mathcal{S} \mapsto \mathcal{R}$ with domain $\mathcal{S}$ and range $\mathcal{R}$ preserves $\varepsilon-$differential privacy if for any output $\Theta \in \mathcal{R}$ and all adjacent datasets $\mathcal{D} \in \mathcal{S}$ and $\mathcal{D}' \in \mathcal{S}$, it holds that

$$\mathbb{P}\left[\tilde{\mathcal{Q}}(\mathcal{D}) \in \Theta\right] \leqslant \mathbb{P}\left[\tilde{\mathcal{Q}}(\mathcal{D}') \in \Theta\right] \exp(\varepsilon),$$

where probability is taken over runs of $\tilde{\mathcal{Q}}$.

Differential privacy (definition)



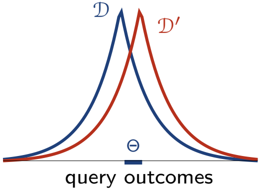
- $\mathcal{Q}$ is a query computed on a dataset
- ξ is a carefully calibrated noise
- θ and θ' are stat. indistinguishable
- By observing θ or θ', analyst can't tell if your data is included

ε-differential privacy [Dwork et al., 2014]

A randomized query $\tilde{\mathcal{Q}} : \mathcal{S} \mapsto \mathcal{R}$ with domain $\mathcal{S}$ and range $\mathcal{R}$ preserves $\varepsilon-$differential privacy if for any output $\Theta \in \mathcal{R}$ and all adjacent datasets $\mathcal{D} \in \mathcal{S}$ and $\mathcal{D}' \in \mathcal{S}$, it holds that

$$\mathbb{P}\left[\tilde{\mathcal{Q}}(\mathcal{D}) \in \Theta\right] \leqslant \mathbb{P}\left[\tilde{\mathcal{Q}}(\mathcal{D}') \in \Theta\right] \exp(\varepsilon),$$

where probability is taken over runs of $\tilde{\mathcal{Q}}$.



Differentially private distributed OPF

- We treat OPF sub-problems as queries

$$\mathcal{Q}_i : \mathcal{D}_i \mapsto \theta_i,$$

where

$$\mathcal{D}_i = \{\underbrace{c_{1i}, c_{2i}, B_i, \rho}_{\text{side info}}, \overbrace{\mu_i, \overline{\theta}_i}^{\text{input}}, \underbrace{d_i}_{\text{sensitive}}\}$$

- Adjacent datasets $\mathcal{D}$ and $\mathcal{D}'$:

$$\|\mathcal{D}_i - \mathcal{D}'_i\|_1 = \|d_i - d'_i\|_1 \leqslant \alpha$$

- Sensitivity of a query:

$$\Delta_{\mathcal{Q}_i} := \max_{\mathcal{D} \sim_\alpha \mathcal{D}'} \|\mathcal{Q}(\mathcal{D}) - \mathcal{Q}(\mathcal{D}')\|_1$$

Two method to achieve differential privacy

[Chaudhuri et al., 2011, Zhang and Zhu, 2016]

Output perturbation	**Query perturbation**
$\tilde{\mathcal{Q}}_i(\mathcal{D}_i) = \mathcal{Q}_i(\mathcal{D}_i) + \xi_i = \tilde{\theta}_i$	$\tilde{\mathcal{Q}}_i(\mathcal{D}_i; \xi_i) = \tilde{\theta}_i$
The output is purturbed by noise ξ_i	The query is purturbed itself by noise ξ_i
This presentation	Refer to arXiv extended paper

Differentially private distributed OPF

- We treat OPF sub-problems as queries

$$\mathcal{Q}_i : \mathcal{D}_i \mapsto \theta_i,$$

where

$$\mathcal{D}_i = \{\underbrace{c_{1i}, c_{2i}, B_i, \rho}_{\text{side info}}, \overbrace{\mu_i, \overline{\theta}_i}^{\text{input}}, \underbrace{d_i}_{\text{sensitive}}\}$$

- Adjacent datasets $\mathcal{D}$ and $\mathcal{D}'$:

$$\|\mathcal{D}_i - \mathcal{D}'_i\|_1 = \|d_i - d'_i\|_1 \leqslant \alpha$$

- Sensitivity of a query:

$$\Delta_{\mathcal{Q}_i} := \max_{\mathcal{D}\sim_\alpha \mathcal{D}'} \|\mathcal{Q}(\mathcal{D}) - \mathcal{Q}(\mathcal{D}')\|_1$$

Two method to achieve differential privacy
[Chaudhuri et al., 2011, Zhang and Zhu, 2016]

Output perturbation

$$\tilde{\mathcal{Q}}_i(\mathcal{D}_i) = \mathcal{Q}_i(\mathcal{D}_i) + \xi_i = \tilde{\theta}_i$$

The output is purturbed by noise ξ_i

This presentation

Query perturbation

$$\tilde{\mathcal{Q}}_i(\mathcal{D}_i; \xi_i) = \tilde{\theta}_i$$

The query is purturbed itself by noise ξ_i

Refer to arXiv extended paper

Differentially private distributed OPF

- We treat OPF sub-problems as queries

$$\mathcal{Q}_i : \mathcal{D}_i \mapsto \theta_i,$$

where

$$\mathcal{D}_i = \{\underbrace{c_{1i}, c_{2i}, B_i, \rho}_{\text{side info}}, \overbrace{\mu_i, \overline{\theta}_i}^{\text{input}}, \underbrace{d_i}_{\text{sensitive}}\}$$

- Adjacent datasets $\mathcal{D}$ and $\mathcal{D}'$:

$$\|\mathcal{D}_i - \mathcal{D}_i'\|_1 = \|d_i - d_i'\|_1 \leqslant \alpha$$

- Sensitivity of a query:

$$\Delta_{\mathcal{Q}_i} := \max_{\mathcal{D}\sim_\alpha \mathcal{D}'} \|\mathcal{Q}(\mathcal{D}) - \mathcal{Q}(\mathcal{D}')\|_1$$

Two method to achieve differential privacy
[Chaudhuri et al., 2011, Zhang and Zhu, 2016]

Output perturbation

$$\tilde{\mathcal{Q}}_i(\mathcal{D}_i) = \mathcal{Q}_i(\mathcal{D}_i) + \xi_i = \tilde{\theta}_i$$

The output is purturbed by noise ξ_i

This presentation

Query perturbation

$$\tilde{\mathcal{Q}}_i(\mathcal{D}_i; \xi_i) = \tilde{\theta}_i$$

The query is purturbed itself by noise ξ_i

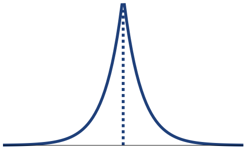
Refer to arXiv extended paper

Differentially private distributed OPF (cont'd)

Output perturbation

$$\mathcal{Q}_i(\mathcal{D}_i) + \xi_i \sim \frac{\varepsilon}{2\Delta_{\mathcal{Q}_i}} \exp\left(-\varepsilon \frac{|\xi_i - \theta_i|}{\Delta_{\mathcal{Q}_i}}\right),$$

where $\Delta_{\mathcal{Q}_i}$ is the output sensitivity to the value of load (adjusted by α)



Randomized ADMM for distributed OPF

1. Update $\theta_i \leftarrow \text{argmin}_{p_i,\theta_i} \mathcal{L}(p_i, \theta_i, \overline{\theta}, \mu_i)$
2. Output perturbation: $\tilde{\theta}_i(\boldsymbol{\xi_i}) \leftarrow \theta_i + \boldsymbol{\xi_i}$
3. Update $\overline{\theta} \leftarrow \text{argmin}_{\overline{\theta}} \mathcal{L}(\tilde{\theta}_i(\boldsymbol{\xi_i}), \overline{\theta}, \mu_i)$
4. Update $\mu_i \leftarrow \mu_i + \rho(\tilde{\theta}_i(\boldsymbol{\xi_i}) - \overline{\theta})$
5. Terminate if $\|\tilde{\theta}_i(\boldsymbol{\xi_i}) - \overline{\theta}\|_2 \leqslant \eta$

Main result

$$\mathbb{P}[\mathcal{Q}_i(\mathcal{D}_i) + \xi_i \in \tilde{\theta}_i] \leqslant \mathbb{P}[\mathcal{Q}_i(\mathcal{D}_i^{'}) + \xi_i \in \tilde{\theta}_i]\exp(\epsilon)$$

Extensions include

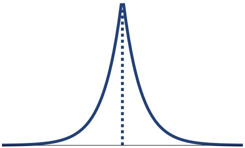
- Static or dynamic random perturbations
- Global or local query sensitivity
- Privacy preservation across iterations

Differentially private distributed OPF (cont'd)

Output perturbation

$$\mathcal{Q}_i(\mathcal{D}_i) + \xi_i \sim \frac{\varepsilon}{2\Delta_{\mathcal{Q}_i}} \exp\left(-\varepsilon \frac{|\xi_i - \theta_i|}{\Delta_{\mathcal{Q}_i}}\right),$$

where $\Delta_{\mathcal{Q}_i}$ is the output sensitivity to the value of load (adjusted by α)



Randomized ADMM for distributed OPF

1. Update $\theta_i \leftarrow \text{argmin}_{p_i,\theta_i} \mathcal{L}(p_i, \theta_i, \overline{\theta}, \mu_i)$
2. Output perturbation: $\tilde{\theta}_i(\boldsymbol{\xi_i}) \leftarrow \theta_i + \boldsymbol{\xi_i}$
3. Update $\overline{\theta} \leftarrow \text{argmin}_{\overline{\theta}} \mathcal{L}(\tilde{\theta}_i(\boldsymbol{\xi_i}), \overline{\theta}, \mu_i)$
4. Update $\mu_i \leftarrow \mu_i + \rho(\tilde{\theta}_i(\boldsymbol{\xi_i}) - \overline{\theta})$
5. Terminate if $\|\tilde{\theta}_i(\boldsymbol{\xi_i}) - \overline{\theta}\|_2 \leqslant \eta$

Main result

$$\mathbb{P}[\mathcal{Q}_i(\mathcal{D}_i) + \xi_i \in \tilde{\theta}_i] \leqslant \mathbb{P}[\mathcal{Q}_i(\mathcal{D}_i^{'}) + \xi_i \in \tilde{\theta}_i]\exp(\epsilon)$$

Extensions include

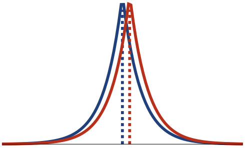
- Static or dynamic random perturbations
- Global or local query sensitivity
- Privacy preservation across iterations

Differentially private distributed OPF (cont'd)

Output perturbation

$$\mathcal{Q}_i(\mathcal{D}_i) + \xi_i \sim \frac{\varepsilon}{2\Delta_{\mathcal{Q}_i}} \exp\left(-\varepsilon \frac{|\xi_i - \theta_i|}{\Delta_{\mathcal{Q}_i}}\right),$$

where $\Delta_{\mathcal{Q}_i}$ is the output sensitivity to the value of load (adjusted by α)



Randomized ADMM for distributed OPF

1. Update $\theta_i \leftarrow \text{argmin}_{p_i,\theta_i}\ \mathcal{L}(p_i, \theta_i, \overline{\theta}, \mu_i)$
2. Output perturbation: $\tilde{\theta}_i(\boldsymbol{\xi_i}) \leftarrow \theta_i + \boldsymbol{\xi_i}$
3. Update $\overline{\theta} \leftarrow \text{argmin}_{\overline{\theta}}\ \mathcal{L}(\tilde{\theta}_i(\boldsymbol{\xi_i}), \overline{\theta}, \mu_i)$
4. Update $\mu_i \leftarrow \mu_i + \rho(\tilde{\theta}_i(\boldsymbol{\xi_i}) - \overline{\theta})$
5. Terminate if $\|\tilde{\theta}_i(\boldsymbol{\xi_i}) - \overline{\theta}\|_2 \leqslant \eta$

Main result

$$\mathbb{P}[\mathcal{Q}_i(\mathcal{D}_i) + \xi_i \in \tilde{\theta}_i] \leqslant \mathbb{P}[\mathcal{Q}_i(\mathcal{D}_i') + \xi_i \in \tilde{\theta}_i]\exp(\epsilon)$$

Extensions include

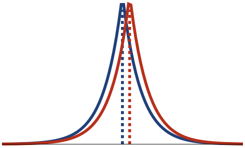
- Static or dynamic random perturbations
- Global or local query sensitivity
- Privacy preservation across iterations

Differentially private distributed OPF (cont'd)

Output perturbation

$$\mathcal{Q}_i(\mathcal{D}_i) + \xi_i \sim \frac{\varepsilon}{2\Delta_{\mathcal{Q}_i}} \exp\left(-\varepsilon \frac{|\xi_i - \theta_i|}{\Delta_{\mathcal{Q}_i}}\right),$$

where $\Delta_{\mathcal{Q}_i}$ is the output sensitivity to the value of load (adjusted by α)



Randomized ADMM for distributed OPF

1. Update $\theta_i \leftarrow \text{argmin}_{p_i,\theta_i} \mathcal{L}(p_i, \theta_i, \overline{\theta}, \mu_i)$
2. Output perturbation: $\tilde{\theta}_i(\boldsymbol{\xi_i}) \leftarrow \theta_i + \boldsymbol{\xi_i}$
3. Update $\overline{\theta} \leftarrow \text{argmin}_{\overline{\theta}} \mathcal{L}(\tilde{\theta}_i(\boldsymbol{\xi_i}), \overline{\theta}, \mu_i)$
4. Update $\mu_i \leftarrow \mu_i + \rho(\tilde{\theta}_i(\boldsymbol{\xi_i}) - \overline{\theta})$
5. Terminate if $\|\tilde{\theta}_i(\boldsymbol{\xi_i}) - \overline{\theta}\|_2 \leqslant \eta$

Main result

$$\mathbb{P}[\mathcal{Q}_i(\mathcal{D}_i) + \xi_i \in \tilde{\theta}_i] \leqslant \mathbb{P}[\mathcal{Q}_i(\mathcal{D}_i^{'}) + \xi_i \in \tilde{\theta}_i]\exp(\epsilon)$$

Extensions include

- Static or dynamic random perturbations
- Global or local query sensitivity
- Privacy preservation across iterations

Numerical experiments on 3-area IEEE 118-node RTS

Illustrations of privacy attacks

Experiment description:

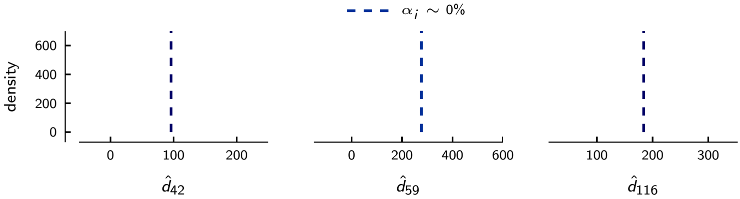
- Privacy loss is fixed $\varepsilon = 1$
- Adjacency coefficient α varies
- Privacy of local OPF datasets improve in α
- Privacy adversary infers individual loads

Illustrations of privacy attacks

Experiment description:

- Privacy loss is fixed $\varepsilon = 1$
- Adjacency coefficient α varies
- Privacy of local OPF datasets improve in α
- Privacy adversary infers individual loads

Nodal ADMM decomposition of the IEEE 118-node system:

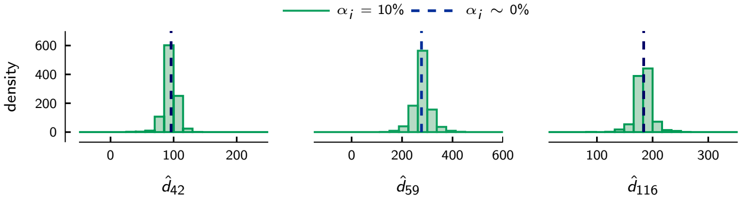


Illustrations of privacy attacks

Experiment description:

- Privacy loss is fixed $\varepsilon = 1$
- Adjacency coefficient α varies
- Privacy of local OPF datasets improve in α
- Privacy adversary infers individual loads

Nodal ADMM decomposition of the IEEE 118-node system:

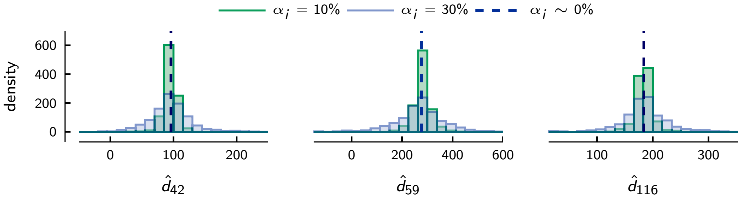


Illustrations of privacy attacks

Experiment description:

- Privacy loss is fixed $\varepsilon = 1$
- Adjacency coefficient α varies
- Privacy of local OPF datasets improve in α
- Privacy adversary infers individual loads

Nodal ADMM decomposition of the IEEE 118-node system:

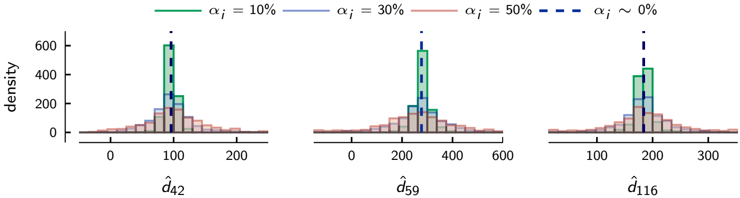


Illustrations of privacy attacks

Experiment description:

- Privacy loss is fixed $\varepsilon = 1$
- Adjacency coefficient α varies
- Privacy of local OPF datasets improve in α
- Privacy adversary infers individual loads

Nodal ADMM decomposition of the IEEE 118-node system:



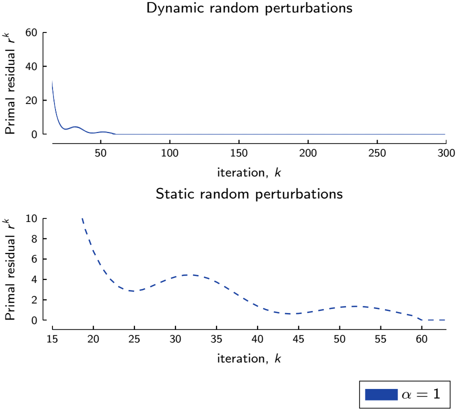
As α increases, adversarial load inference converges to random guessing

Convergence and optimality trade-offs

- Poorer convergence due to noise
- Noisy computations involve optimality loss
- The two can be traded off by using static or dynamically updated noise across iterations

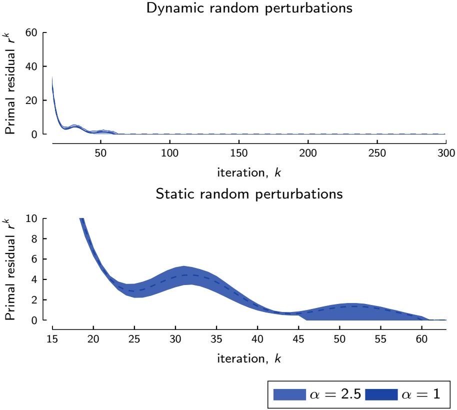
Convergence and optimality trade-offs

- Poorer convergence due to noise
- Noisy computations involve optimality loss
- The two can be traded off by using static or dynamically updated noise across iterations



Convergence and optimality trade-offs

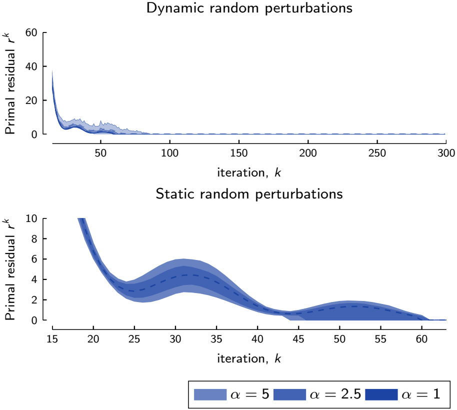
- Poorer convergence due to noise
- Noisy computations involve optimality loss
- The two can be traded off by using static or dynamically updated noise across iterations



Convergence and optimality trade-offs

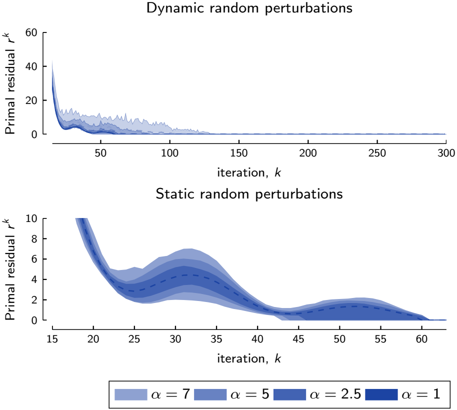
- Poorer convergence due to noise
- Noisy computations involve optimality loss

- The two can be traded off by using static or dynamically updated noise across iterations



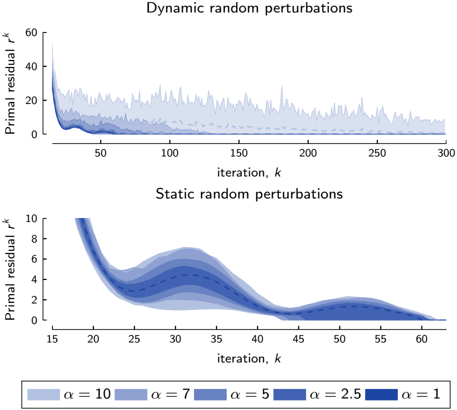
Convergence and optimality trade-offs

- Poorer convergence due to noise
- Noisy computations involve optimality loss
- The two can be traded off by using static or dynamically updated noise across iterations



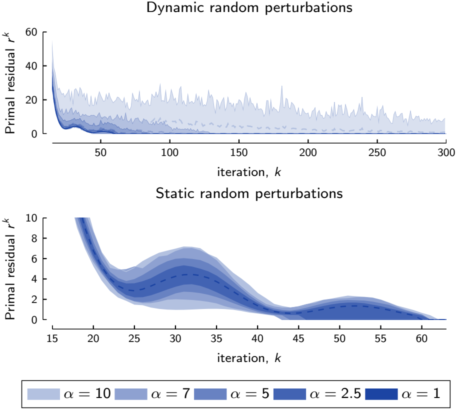
Convergence and optimality trade-offs

- Poorer convergence due to noise
- Noisy computations involve optimality loss
- The two can be traded off by using static or dynamically updated noise across iterations



Convergence and optimality trade-offs

- Poorer convergence due to noise
- Noisy computations involve optimality loss
- The two can be traded off by using static or dynamically updated noise across iterations



Optimality loss [%]

Adjacency coefficient α,%	1	2.5	5	7	10
Dynamic perturbations	0.48	0.92	1.23	1.51	3.83
Static perturbations	0.28	4.33	11.0	11.35	20.41

Privacy guarantee beyond one iteration

- Repeated computations on the same dataset accumulates privacy losses
- Attacker exploits all compromised iterations, e.g., last T iterations $k - T, \ldots, K$
- It thus offsets the effect of noise, i.e., $\mathbb{E}_{\boldsymbol{\xi}_i}[(\theta_i + \boldsymbol{\xi}_i)] = \theta_i$
- To avoid these privacy risks, we use decomposition of differential privacy:
  - We scale the noise by factor of T, i.e. $\xi_i \sim \text{Lap}(T \times \Delta_{\mathcal{Q}}/\varepsilon)$
  - and thus obtain $\varepsilon-$differential privacy after T iterations

Privacy guarantee beyond one iteration

- Repeated computations on the same dataset accumulates privacy losses
- Attacker exploits all compromised iterations, e.g., last T iterations $k - T, \ldots, K$
- It thus offsets the effect of noise, i.e., $\mathbb{E}_{\boldsymbol{\xi}_i}\left[(\theta_i + \boldsymbol{\xi}_i)\right] = \theta_i$
- To avoid these privacy risks, we use decomposition of differential privacy:
  - We scale the noise by factor of T, i.e. $\xi_i \sim \text{Lap}(T \times \Delta_{\mathcal{Q}}/\varepsilon)$
  - and thus obtain $\varepsilon-$differential privacy after T iterations

Inference RMSE **without** composition

Adjacency α, % \ Attack budget T	1	2	5	10	15
1.0	0.2	0.2	0.1	0.1	0.1
2.5	0.7	0.5	0.4	0.4	0.4
5.0	1.1	1	1	0.8	0.8
7.0	2.1	1.9	1.3	1.2	1.1
10.0	3.3	2.3	1.8	1.7	1.5

Privacy guarantee beyond one iteration

- Repeated computations on the same dataset accumulates privacy losses
- Attacker exploits all compromised iterations, e.g., last T iterations $k - T, \dots, K$
- It thus offsets the effect of noise, i.e., $\mathbb{E}_{\boldsymbol{\xi}_i}\left[(\theta_i + \boldsymbol{\xi}_i)\right] = \theta_i$
- To avoid these privacy risks, we use decomposition of differential privacy:
    - We scale the noise by factor of T, i.e. $\xi_i \sim \mathrm{Lap}(T \times \Delta_{\mathcal{Q}}/\varepsilon)$
    - and thus obtain $\varepsilon-$differential privacy after T iterations

Inference RMSE **without** composition

Adjacency α, % \ Attack budget T	1	2	5	10	15
1.0	0.2	0.2	0.1	0.1	0.1
2.5	0.7	0.5	0.4	0.4	0.4
5.0	1.1	1	1	0.8	0.8
7.0	2.1	1.9	1.3	1.2	1.1
10.0	3.3	2.3	1.8	1.7	1.5

Inference RMSE **with** composition

Adjacency α, % \ Attack budget T	1	2	5	10	15
1.0	0.2	0.6	1.2	1.2	1.1
2.5	0.7	1.7	2.3	2.4	2.9
5.0	1.1	2.6	3.8	4.8	6.5
7.0	2.1	4.3	6	7.6	9.9
10.0	3.3	5.3	8.5	11.4	16.6

Conclusions

- We develop differentially private distributed OPF algorithms ...
- ... to provide formal privacy gurantees for local OPF datasets
- The algorithms are open source and available at

  `https://github.com/wdvorkin/DP_D_OPF`

- Future research includes analyzing convergence rate as a function of privacy parameters

Thank you for your attention!

References I

Biskas, P. N., Bakirtzis, A. G., Macheras, N. I., and Pasialis, N. K. (2005).
A decentralized implementation of DC optimal power flow on a network of computers.
IEEE Transactions on Power Systems, 20(1):25–33.

Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J., et al. (2011).
Distributed optimization and statistical learning via the alternating direction method of multipliers.
Foundations and Trends® in Machine learning, 3(1):1–122.

Chaudhuri, K., Monteleoni, C., and Sarwate, A. D. (2011).
Differentially private empirical risk minimization.
Journal of Machine Learning Research, 12(Mar):1069–1109.

Conejo, A. J. and Aguado, J. A. (1998).
Multi-area coordinated decentralized DC optimal power flow.
IEEE Transactions on Power Systems, 13(4):1272–1278.

Dwork, C., Roth, A., et al. (2014).
The algorithmic foundations of differential privacy.
Foundations and Trends® in Theoretical Computer Science, 9(3–4):211–407.

Molzahn, D. K., Dörfler, F., Sandberg, H., Low, S. H., Chakrabarti, S., Baldick, R., and Lavaei, J. (2017).
A survey of distributed optimization and control algorithms for electric power systems.
IEEE Transactions on Smart Grid, 8(6):2941–2962.

References II

Zhang, T. and Zhu, Q. (2016).
Dynamic differential privacy for ADMM-based distributed classification learning.
IEEE Transactions on Information Forensics and Security, 12(1):172–187.