Privacy-Preserving Obfuscation for Distributed Power Systems

Terrence W.K. Mak*, Ferdinando Fioretto†, and Pascal Van Hentenryck*
* Georgia Institute of Technology, Atlanta, GA, USA
† Syracuse University, New York, NY, USA
wmak@gatech.edu, ffiorett@syr.edu, pvh@isye.gatech.edu

***Abstract*—This paper considers the problem of releasing privacy-preserving load data of a decentralized operated power system. The paper focuses on data used to solve Optimal Power Flow (OPF) problems and proposes a distributed algorithm that complies with the notion of *Differential Privacy*, a strong privacy framework used to bound the risk of re-identification. The problem is challenging since the application of traditional differential privacy mechanisms to the load data fundamentally changes the nature of the underlying optimization problem and often leads to severe feasibility issues. The proposed *differentially private distributed algorithm* is based on the Alternating Direction Method of Multipliers (ADMM) and guarantees that the released privacy-preserving data retains high fidelity and satisfies the AC power flow constraints. Experimental results on a variety of OPF benchmarks demonstrate the effectiveness of the approach.**

***Index Terms*—Differential Privacy, Optimal Power Flow, ADMM, Distributed computing**

I. Introduction

The availability of test cases representing high-fidelity power system networks is essential to foster research in several important power optimization problems, including optimal power flow (OPF), unit commitment, and transmission planning. However, the release of such datasets poses significant privacy risks. For instance, revealing the electrical load of a customer may disclose sensitive business activities and manufacturing processes, causing significant economic loss. Indirectly, it may also reveal how transmission operators operate their networks, raising security issues [1].

Differential Privacy (DP) [2] is a privacy framework that has been shown effective in protecting sensitive information during a data release process. It prevents the disclosure of sensitive information by introducing carefully calibrated noise to the result of a computation. While DP algorithms could be used *directly* to generate privacy-preserving power system data, they face significant challenges when the released data is required to preserve domain specific properties, such as preserving the optimal cost and the feasibility of an AC Optimal Power Flow (AC-OPF) problem. Naive noise addition can drastically degrade the fidelity to the original problem of interest and introduce severe feasibility issues, as shown in [1], [3], [4].

The authors would like to thank Kory Hedman for extensive discussions on various obfuscation techniques. This research is partly funded by the ARPA-E Grid Data Program under Grant 1357-1530.

Fig. 1, reported from [3], emphasizes these results. It shows the average load distance (as L_1) between the original and the privacy-preserving loads for a set of 29 networks, at varying of the obfuscation parameter α. The percentages of instances with a feasible AC-OPF solution are shown above the bars.

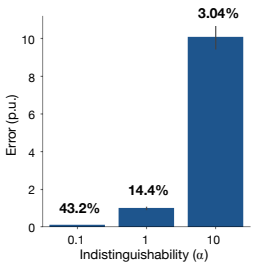

Fig. 1. Average L_1 error and percentages of feasible AC-OPF instances.

Interestingly, a recent body of work has shown that it is possible to release AC-feasible obfuscated load data that also satisfies the notion of differential privacy [3]–[5]. Despite the soundness and effectiveness of such data release techniques, these methods rely on the presence of a trusted data curator that can collect sensitive loads from all the system participants. However, this is impractical in very large systems with distributed loads and generators (e.g., multiple microgrids). Even if the power system is operated centrally, it is typically owned and controlled by various parties, e.g., load customers, transmission system operators (TSO), distribution system operators (DSO), and generation companies. These parties operate with specific customer and legal agreements, which render the transmission of proprietary data to a centralized server infeasible.

To overcome these limitations, this paper introduces the *Privacy-preserving Decentralized OPF (PD-OPF)*, a novel decentralized and privacy-preserving framework that allows multiple power system parties to release their data privately without relying on a trusted data curator. Crucially, the framework guarantees that the released data produces a feasible AC power flow and that its OPF cost is close to that of the original OPF. The heart of the mechanism is a distributed optimization procedure that relies on the *Alternating Direction Method of Multipliers (ADMM)* to redistribute the noise introduced by traditional DP algorithms to satisfy the desired properties. The proposed mechanism leverages standard DP primitives and can be easily adopted on top of *any* existing distributed power system optimization algorithm.

While the paper focuses on preserving the privacy of individual loads, the framework is general and can be used to protect other sensitive quantities (e.g., generator capabilities).

Contributions The key contributions of this work are as

follows: (1) It introduces PD-OPF, a novel, distributed mechanism that relies on ADMM to obfuscate the individual loads while ensuring AC-OPF feasibility on the obfuscated data. PD-OPF satisfies the notion of local differential privacy, providing strong privacy guarantees. (2) Experimental results on a large collection of OPF benchmarks illustrate that the proposed approach finds high-quality AC-feasible solutions, and that the results are comparable to those obtained with a centralized version with a data curator.

II. Related Work

There is a rich literature on theoretical results of DP (see for instance [6] and [7]). The literature on DP applied to power systems includes considerably fewer efforts. Ács and Castelluccia [8] exploit a direct application of the Laplace mechanism to hide user participation in smart meter data sets, achieving ϵ-DP. Zhao et al. [9] study a DP schema that exploits the ability of households to charge and discharge a battery to hide the real energy consumption of their appliances. Liao et al. [10] introduce Di-PriDA, a privacy-preserving mechanism for appliance-level peak-time load balancing control in the smart grid, aimed at masking the consumption of top-k appliances of a household. Halder et al. [11] propose an architecture for privacy-preserving thermal inertial load management as a service provided by load-serving entities. Finally, Zhou et al. [12] introduce the notion of monotonicity of DC-OPF operator, which requires that monotonic changes in the network loads induce monotonic changes in the DC-OPF objective cost. This enables a characterization of the network, which is useful to preserve the privacy of *monotonic networks*.

There are also related work on privacy-preserving implementations of the ADMM algorithm. Zhang et al. [13] proposed a version of the ADMM algorithm for privacy-preserving empirical risk minimization problems, a class of convex problems used for regression and classification tasks. Huang et al. [14] proposed an approach that combines an approximate augmented Lagrangian function with time-varying Gaussian noise for general objective functions. Finally, Ding et al. [15] proposed P-ADMM, to provide guarantees within a *relaxed* model of differential privacy (called zero-concentrated DP).

The privacy-preserving distributed learning literature focuses almost entirely on problems whose objective functions are smooth and strongly convex. Additionally, most approaches suffer one shortcoming: The privacy loss being provided as a guarantee is a function of the iteration counts of the algorithm, which can be huge if a large number of iterations is required to converge to a feasible solution. In contrast, this work provides bounded privacy loss irrespective of the number of iterations. It also ensures that the privacy-preserving data is AC-OPF feasible and that the solution cost stays close to the original ones.

Model 1 AC Optimal Power Flow: P_{OPF}

$$\textbf{variables:}\quad S_i^g,\ \forall i \in G;\quad V_i,\ \forall i \in N;\quad S_{ij},\ \forall (i,j) \in E \cup E^R$$

$$\textbf{minimize:}\quad \mathcal{O}(\boldsymbol{S^g}) = \sum_{i \in N} c_{2i}(\Re(S_i^g))^2 + c_{1i}\Re(S_i^g) + c_{0i} \tag{1}$$

$$\textbf{subject to:}\quad \angle V_s = 0,\ \ \exists s \in N \tag{2}$$

$$v_i^l \leqslant |V_i| \leqslant v_i^u\ \ \forall i \in N \tag{3}$$

$$-\theta_{ij}^{\Delta} \leqslant \angle(V_i V_j^*) \leqslant \theta_{ij}^{\Delta}\ \ \forall (i,j) \in E \tag{4}$$

$$S_i^{gl} \leqslant S_i^g \leqslant S_i^{gu}\ \ \forall i \in G \subseteq N \tag{5}$$

$$|S_{ij}| \leqslant s_{ij}^u\ \ \forall (i,j) \in E \cup E^R \tag{6}$$

$$S_i^g - S_i^d = \textstyle\sum_{(i,j) \in E \cup E^R} S_{ij}\ \ \forall i \in N \tag{7}$$

$$S_{ij} = Y_{ij}^* |V_i|^2 - Y_{ij}^* V_i V_j^*\ \ \forall (i,j) \in E \cup E^R \tag{8}$$

III. Preliminaries

A. Optimal Power Flow

Optimal Power Flow (OPF) is the problem of determining the most economic generator dispatch to meet the load demands in a power network. A power network $\boldsymbol{\mathcal{N}}$ can be viewed as a graph (N, E) where the set of buses $N = [n]$ represents the nodes and the set of lines and transformers $E \subseteq \{(i,j) \in N \times N\}$ represents the directed arcs. The paper denotes with G and L as for the set of generators and loads in the network, and uses E^R to indicate the set of arcs, but in the reverse direction. The AC-OPF problem (P_{OPF}) is specified in Model 1, where I, V, Y, and S denote the complex quantities for current, voltage, admittance, and power, respectively.

The model takes as input the power network $\boldsymbol{\mathcal{N}}$ and returns the optimal generator dispatch costs (with ties broken arbitrarily). The objective function $\mathcal{O}(\boldsymbol{S^g})$ captures the cost of the generator dispatch, with $\boldsymbol{S^g} = \langle S_1^g, \ldots, S_n^g \rangle$ denoting the vector of generator dispatch values. Constraint (2) sets the reference angle to zero for the slack bus $s \in N$ to eliminate numerical symmetries. Constraints (3) and (4) capture the voltage and phase angle difference bounds. Constraints (5) and (6) enforce the generator output and line flow limits. Finally, constraints (7) capture the Kirchhoff's Current Law and constraints (8) capture the Ohm's Law. The solution set satisfying constraints (2) to (8) for a given set of load demands $\boldsymbol{S}^d = \langle S_1^d, \ldots, S_n^d \rangle$ is denoted by $\mathscr{C}_{PF}(\boldsymbol{S}^d)$. Table I summarizes the common notations used throughout the paper.

B. Alternating Direction of Multipliers Method (ADMM)

ADMM is a widely used distributed procedure solving optimization problems with coupling constraints. Consider an optimization problem of the following form:

$$\begin{aligned} \min_{x \in \mathcal{X}, z \in \mathcal{Z}} \quad & f(\boldsymbol{x}) + g(\boldsymbol{z}) \\ \text{s.t.} \quad & A\boldsymbol{x} + B\boldsymbol{z} = \boldsymbol{c}, \end{aligned} \tag{9}$$

where $\mathcal{X} \subseteq \mathbb{R}^n$ and $\mathcal{Z} \subseteq \mathbb{R}^m$ are two disjoint sets, $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{z} \in \mathbb{R}^m$ denote variable vectors owned by two distinct groups of agents, and $A\boldsymbol{x} + B\boldsymbol{z} = \boldsymbol{c}$ describes the set of coupling constraints *between* the two groups of agents with $A \in \mathbb{R}^{\ell \times n}$, $B \in \mathbb{R}^{\ell \times m}$, and $\boldsymbol{c} \in \mathbb{R}^{\ell}$. The functions f and

TABLE I
COMMON NOTATION USED IN THE PAPER.

$\mathcal{N}$	Power network	ϵ	Privacy budget
$\boldsymbol{S}^g$	Vector of power generator dispatch	α	Indistinguishability value
$\boldsymbol{S}^d$	Vector of load demands	β	Faithfulness value/parameter
P_{OPF}	Function solving AC-OPF, with input $\boldsymbol{S}^d$ and output $\boldsymbol{S}^g$	$\mathcal{O}^*$	The optimal costs of the original problem
$\mathscr{C}_{\text{PF}}$	The set of feasible AC power flow for P_{OPF}	$\boldsymbol{x}$	A vector of variables/values
$\mathcal{M}_x$	A mechanism of x	x^l, x^u	Upper and lower bounds of quantity x
P_x	The optimization problem for the accuracy phase of $\mathcal{M}_x$	$\Re(\cdot), \Im(\cdot)$	Real / imaginary component of a complex number
Y^*, I^*, V^*	Conjugate of admittance matrix Y, current I, and voltage V	c_2, c_1, c_0	Cost function coefficients
$\lambda^d, \lambda^g, \lambda^V, \lambda^S$	Lagrange multiplier for load, generation, voltage, and power flow	ρ	ADMM penalty parameter

g denote the objectives over $\boldsymbol{x}$ and $\boldsymbol{z}$, respectively, and are commonly assumed to be convex. The augmented Lagrange function $L_\rho(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\lambda})$ of (9) is:

$$f(\boldsymbol{x}) + g(\boldsymbol{z}) + \boldsymbol{\lambda}^\intercal(A\boldsymbol{x} + B\boldsymbol{z} - \boldsymbol{c}) + \frac{\rho}{2}\|A\boldsymbol{x} + B\boldsymbol{z} - \boldsymbol{c}\|_2^2$$

where $\boldsymbol{\lambda} \in \mathbb{R}^\ell$ is a vector of *Lagrangian multipliers*, with $\rho > 0$ representing the penalty parameter. The vector of Lagrangian multipliers are the dual variables associated with the coupling constraints $A\boldsymbol{x} + B\boldsymbol{z} = \boldsymbol{c}$.

Given a solution tuple $(\boldsymbol{x}^k, \boldsymbol{z}^k, \boldsymbol{\lambda}^k)$ at iteration k, ADMM [16] proceeds to the next iteration, $k+1$, computing $(\boldsymbol{x}^{k+1}, \boldsymbol{z}^{k+1}, \boldsymbol{\lambda}^{k+1})$ as follows, in three sequential steps:

$$\boldsymbol{x}^{k+1} = \underset{\boldsymbol{x}\in\mathcal{X}}{\operatorname{argmin}}\, L_\rho(\boldsymbol{x}, \boldsymbol{z}^k, \boldsymbol{\lambda}^k) \tag{10}$$

$$\boldsymbol{z}^{k+1} = \underset{\boldsymbol{z}\in\mathcal{Z}}{\operatorname{argmin}}\, L_\rho(\boldsymbol{x}^{k+1}, \boldsymbol{z}, \boldsymbol{\lambda}^k) \tag{11}$$

$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \rho(A\boldsymbol{x}^{k+1} + B\boldsymbol{z}^{k+1} - \boldsymbol{c}). \tag{12}$$

The algorithm terminates when a desired termination condition (e.g., an iteration limit or a convergence factor) is reached. The quality of the solution at iteration k can be measured by the primal infeasibility (residue) vector [17]

$$\boldsymbol{r}_p^k = A\boldsymbol{x}^k + B\boldsymbol{z}^k - \boldsymbol{c}, \tag{13}$$

indicating the distance to a primal feasible solution, and the dual infeasibility (residue) vector [17]

$$\boldsymbol{r}_d^k = \rho A^T B(\boldsymbol{z}^k - \boldsymbol{z}^{k-1}), \tag{14}$$

indicating the distance from the previous local minima. When both infeasibility vectors are zero, ADMM converges to a (local) optimal and feasible solution.

C. Differential Privacy Notions

The need for data privacy emerges in two main contexts: the *global* context, as in when institutions release datasets containing information of several users or answer queries on such datasets (e.g., US Census queries [18], [19]), and the *local* context, as in when individuals disclose their personal data to some data curator (e.g., Google Chrome data collection process [20]). In both contexts, privacy is achieved through a randomizer $\mathcal{M}$ adding noise to the data before releasing.

Differential privacy [2] (DP) is an algorithmic property that characterizes and bounds the privacy loss of an individual when its data participates into a computation. It has originally been proposed in the global privacy context and, informally, ensures that an adversary would not be able to reliably infer whether or not a particular individual participates in the dataset, even with an unbounded computational power and access to every other entry of the dataset. The setting adopted in this work studies the local privacy context (Local Differential Privacy [21], aka LDP), in which each load customer i holds a datum, $S_i^d \in \mathbb{C}$, describing the complex load consumption of the bus $i \in N$. While the standard local differential privacy framework is concerned with protecting the participation of an individual into a dataset, in a power system, the individual identity is not a sensitive information: It is a public knowledge that each bus may connect to a demand. The sensitive information is represented by the load magnitude. To accommodate such notion of privacy risk, the paper uses the definition of *generalized differential privacy for metric spaces* [22] and adapts it to the local differential privacy context. Without loss of generality, we focus on Lebesgue spaces $\boldsymbol{L}^1$, and in particular, consider the complex space $\mathbb{C}$ equipped with norm 1. For a given value $\alpha > 0$, a randomized mechanism $\mathcal{M}$ is ϵ-LDP for α distances (a.k.a. local α-indistinguishable), if for all x and $x' \in \mathbb{C}$ s.t. $\|x - x'\|_1 \leqslant \alpha$, and for any output response $o \in \mathbb{C}$:

$$\Pr[\mathcal{M}(x) = o] \leqslant e^\epsilon \Pr[\mathcal{M}(x') = o]. \tag{15}$$

where the probability is computed over the randomness of $\mathcal{M}$. A mechanism satisfying Definition 15 ensures that its output cannot differ to much, if run on similar inputs (i.e., inputs differing on a single value by at most α). In other words, the definition ensures that an attacker obtaining access to a privacy-preserving load value cannot detect, with high probability, the distance between the privacy-preserving value and its original value. The level of *privacy* is controlled by the privacy loss parameter $\epsilon \geqslant 0$, with small values denoting strong privacy. The level of *indistinguishability* is controlled by the parameter $\alpha > 0$. The above definition allows us to *obfuscate* load values that are close to one another while retaining the distinction between those that are far apart. Local Differential Privacy (LDP), including its extension for generic metric spaces, satisfies several important properties. In particular, it is immune to post-processing as defined in the following theorem.

Theorem *1 (Post-Processing Immunity):* [6] Let $\mathcal{M}$ be an ϵ-(local) differentially private mechanism and g be an arbitrary mapping from the set of possible output sequences to an arbitrary set. Then $g \circ \mathcal{M}$ is ϵ-(local) differentially private.

IV. Decentralized Load Obfuscation

The decentralized load obfuscation problem is the problem of coordinating the release of privacy-preserving load data in a power system owned and controlled by multiple parties. We consider a set of agents, each coordinating some power system component, e.g., loads, generators, buses, or power lines. The goal of the problem is to release load data, which is controlled by the load agents.

The problem has three desiderata. (1) It requires obfuscation of the loads up to some amount $\alpha > 0$. (2) It requires that the AC-OPF objective induced by the obfuscated loads is close to that attained using the original data. (3) It requires its agents to coordinate the data release process using a decentralized and confined communication process.

Formally, the decentralized load obfuscation problem finds the active and reactive, privacy-preserving load values $\hat{S}_i^d$ for each load agent $i \in N$ that satisfy the following criteria:

1) ***Privacy***: The original load S_i^d and its privacy-preserving counterpart $\hat{S}_i^d$ are local α-indistinguishable, for every load $i \in N$.
2) ***Fidelity***: For every generator i, the optimal AC-OPF dispatch cost $\mathcal{O}(\hat{S}_i^g)$ obtained by using the obfuscated loads $\hat{S}_i^d$ is required to be close to the original AC-OPF dispatch cost $\mathcal{O}(S_i^g)$ up to a user-defined factor $\beta > 0$:

$$|\mathcal{O}(\hat{S}_i^g) - \mathcal{O}(S_i^g)| \leqslant \beta \mathcal{O}(S_i^g) \quad \forall i \in N.$$

Finally, it requires the ***computation mechanism*** to be performed in a decentralized fashion. In the following, we denote with $\mathcal{O}_i^*$ as for the original optimal generation costs $\mathcal{O}(S_i^g)$, which are assumed to be publicly known [5] (e.g., from the market information). When such information is not publicly available, then it could be (privately) estimated. Notice that, as studied in [3] when some information is publicly available, the attacker posterior about the input dataset after observing the output of a DP algorithm remain bounded with respect to its prior [1].

V. The PD-OPF Mechanism

This section introduces the *Privacy-preserving Distributed OPF (PD-OPF)* mechanism to solve the decentralized load obfuscation problem. PD-OPF agents operate in two phases:

1) **Privacy Phase** During the first phase, each load agent $i \in N$ applies a LDP protocol to obtain an α-local obfuscated version $\tilde{S}_i^d$ of its original load S_i^d. This process is executed independently and autonomously by each load agent in the system.
2) **Fidelity Phase** In the second phase, the agents coordinate a distributed process to adjust the private load values $\tilde{S}_i^d$, to new values $\hat{\boldsymbol{S}}^d$ that achieve the fidelity goal, while deviating as little as possible from the local α-obfuscated loads $\tilde{S}_i^d$.

The next sections describe in details the PD-OPF phases.

A. Privacy Phase

In the privacy phase, each (load) agent i perturbs its load data S_i^d, independently from other agents, so to generate an α-local indistinguishable load $\tilde{S}_i^d$. To do so, the agents use a version of the *Laplace Mechanism*, a method used to guarantee an ϵ-LDP private responses to numeric functions [6]. The Laplace distribution with 0 mean and scale ξ, denoted by $\text{Lap}(\xi)$, has a probability density function $\text{Lap}(x|\xi) = \frac{1}{2\xi} e^{-\frac{|x|}{\xi}}$. Consider a function f a numeric function that maps datasets to $\mathbb{R}$ and let z be a random variable drawn from $\text{Lap}(\xi)$ where $\xi = \frac{\alpha}{\epsilon}$. The following theorem provides an LDP mechanism.

Theorem *2 (Laplace Mechanism):* The Laplace mechanism that outputs $f(x) + z$ achieves α-local indistinguishability.

Since the load data is represented in the complex form, agents use the Polar Laplacian mechanism [3], [23], which is a generalization of the Laplace mechanism to Euclidean spaces. The mechanism satisfies α-local obfuscation [3], [22]. For simplicity, the paper refers to the the Laplace mechanism as for the Polar Laplace mechanism.

B. Fidelity Phase

While simply adding Laplace noise to each load satisfies local α-indistinguishability, the resulting power system data may no longer be AC feasible, nor it may induce a similar optimal dispatch costs. To find a set of loads $\hat{\boldsymbol{S}}^d$ that satisfy the fidelity criteria, a post-processing step that uses a bi-level program P_{BL} can be formulated as follows [3]:

$$P_{BL} = \min \ \|\hat{\boldsymbol{S}}^d - \tilde{\boldsymbol{S}}^d\|^2 \tag{16}$$

$$\text{s.t.:} \quad \left|\mathcal{O}(\boldsymbol{S}^g) - \mathcal{O}^*\right| \leqslant \beta \mathcal{O}^* \tag{17}$$

$$\boldsymbol{S}^g = P_{\text{OPF}}(\hat{\boldsymbol{S}}^d). \tag{18}$$

The upper level objective Eq. (16) minimizes the L2 distance between the noisy loads $\tilde{\boldsymbol{S}}^d$ and the (post-processed) load variables $\hat{\boldsymbol{S}}^d$. Constraint (18) captures the AC-OPF requirement. It computes an AC optimal generator dispatch $\boldsymbol{S}^g$ for the post-processed loads $\hat{\boldsymbol{S}}^d$. Finally, Constraint (17) requires the generator dispatch to satisfy the fidelity goal.

Solving bilevel programs is challenging computationally, being strongly NP-Hard [24]. To address the underlying computational challenge, an efficient relaxation of problem P_{BL} can be provided as in [3]:

$$P_{\text{RBL}} = \min \ \|\hat{\boldsymbol{S}}^d - \tilde{\boldsymbol{S}}^d\|^2 \tag{19}$$

$$\text{s.t.:} \quad \left|\mathcal{O}(\boldsymbol{S}^g) - \mathcal{O}^*\right| \leqslant \beta \mathcal{O}^* \tag{20}$$

$$\text{AC Power Flow: (2) – (8).} \tag{21}$$

It relaxes the optimality requirement Eq. (18) and only requires AC feasibility (Eq. (21)). The mechanism restores feasibility of the loads and ensures the existence of a dispatch whose cost is close to the optimal one.

[1] A straightforward corollary is that if the prior knowledge is the dataset itself, then no privacy mechanism can be used to modify such knowledge. An example will be revealing the dispatch cost of a small power grid containing only one generator and one load.

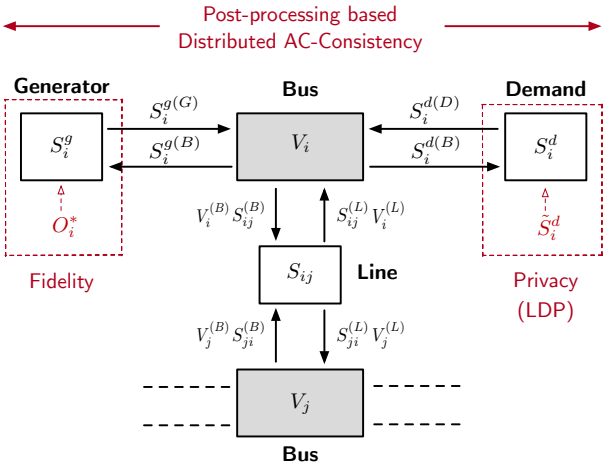

Fig. 2. The ADMM-based LDP post-processing step of PD-OPF.

C. Decentralized Fidelity Phase

To coordinate the resolution of problem P_{RBL} in a decentralized fashion, the problem is expressed into the format of Eq. (9) and solved using an ADMM protocol. The ADMM mechanism used follows the component-based dual decomposition framework [17], [25] and models each power system component as an individual agent. The framework considers four types of agents: load demand agents $\mathcal{D}$, generator agents $\mathcal{G}$, line agents $\mathcal{L}$, and bus agents $\mathcal{B}$. Figure 2 illustrates the ADMM communication scheme adopted by each agent ($i \in N$, if it is a bus, load, or generator agent), or ($(i,j) \in E$, if it is a line agent).

It is summarized in the following three steps. At each iteration:

1) Load, generator, and line agents compute their individual *consensus variables*, respectively, $S_i^{d(D)}$, for load agent i, $S_i^{g(G)}$, for generator agent i, and $S_{ij}^{(L)}, V_{ij}^{(L)}$ (and $S_{ji}^{(L)}, V_{ji}^{(L)}$ for the reverse direction) for line agent (ij). Collectively, they form a *consensus vector* $\boldsymbol{x} = \langle S_i^{d(D)}, S_i^{g(G)}, S_{ij}^{(L)}, V_{ij}^{(L)}, S_{ji}^{(L)}, V_{ji}^{(L)} \rangle$ (see Eq. (10)), which is sent to their connecting bus agents.
2) Upon receiving its neighboring load, generator, and line consensus variables, bus agent i computes the response value $\boldsymbol{z} = \langle S_i^{d(B)}, S_i^{g(B)}, S_{ij}^{(B)}, V_i^{(B)} \rangle$ (see Eq. (11)) and send value $S_i^{d(B)}$ to load agent i, value $S_i^{g(B)}$ to generator agent i, and values $S_{ij}^{(B)}, V_i^{(B)}$ to line agent (ij), for each line (i,j) connected to bus i.
3) Finally, each agent updates its corresponding dual variables in the complex domain $\mathbb{C}$: λ_i^d, for load agent i, λ_i^g, for generator agent i, and $\lambda_{ij}^V, \lambda_{ij}^S$, for line agent (i,j). Collectively, they are identified with $\boldsymbol{\lambda} = \langle \lambda_i^d, \lambda_i^g, \lambda_{ij}^V, \lambda_{ij}^S \rangle$, using the notation in Eq. (12).

The goal of the coupling constraints $A\boldsymbol{x} + B\boldsymbol{z} = \boldsymbol{c}$ (see Constraint (9)) is that of matching the states values $\boldsymbol{z}$ of the bus agents to those of their connected components $\boldsymbol{x}$. The fidelity constraint (Eq. (20)) and the AC Power Flow constraints (Eq. (21)) are enforced as local constraints by each agent. Finally, the load agents control the minimization term Objective (19) of problem P_{RBL}, to control the deviation of the new, post-processed load w.r.t. the Laplace obfuscated counterpart. A detailed description of the individual optimization problems computing the local Lagrange functions (Eq. 10) for the load, generator, and line agents, and (Eq. 11) for the bus agents is given as follows. [2]

Model 2 ADMM: Load agent $\mathcal{D}_i(P_{load})$

inputs: $\langle \rho, \lambda_i^d, \tilde{S}_i^d, S_i^{d(B)} \rangle$
variables: $S_i^{d(D)}$
minimize: $\|S_i^{d(D)} - \tilde{S}_i^d\|^2 + \lambda_i^d \cdot S_i^{d(D)} + \frac{\rho}{2}\|S_i^{d(D)} - S_i^{d(B)}\|^2$ (22)

Model 3 ADMM: Generator agent $\mathcal{G}_i(P_{gen})$

inputs: $\langle \rho, \lambda_i^g, \mathcal{O}_i^*, S_i^{g(B)} \rangle$
variables: $S_i^{g(G)}$
minimize: $\lambda_i^g \cdot S_i^{g(G)} + \frac{\rho}{2}\|S_i^{g(G)} - S_i^{g(B)}\|^2$ (23)
local constraints: $S_i^{gl} \leqslant S_i^{g(G)} \leqslant S_i^{gu}$ (24)
$\mathcal{O}_i^*(1-\beta) \leqslant O(S_i^{g(G)}) \leqslant \mathcal{O}_i^*(1+\beta)$ (25)

Load agent The optimization step performed by each load agent i ($i \in N$), at each iteration, produces a load value $S_i^{d(D)}$ and is shown in Model 2. Eq. (22) captures the load augmented Lagrange function (see Eq. (19)) and the agent coupling constraints described as penalty terms. The first term of the objective is the L2 distance between the load value $S_i^{d(D)}$ and the Laplace load value $\tilde{S}_i^d$. The remaining terms correspond to the load coupling constraint, matching the load values $S_i^{d(D)}$ to the feedback signal $S_i^{d(B)}$ from the connecting bus.

Generator agent The objective of the generator agent i ($i \in N$), at each iteration, is that of producing a dispatch value $S_i^{g(G)}$ that matches the feedback signal $S_i^{g(B)}$ from the connecting bus. The problem is reported in Model 3. Therein, Eq. (23) describes the generator agent coupling constraints as penalty terms. The optimization model ensures that the dispatch values satisfy the feasible bounds (Eq. (24)), and that the dispatch cost stays within the fidelity requirement (Eq. (25)).

Line agent The objective of the line agent (ij) ($(i,j) \in E$), is that of finding flow values $S_{ij}^{(L)}$ and $S_{ji}^{(L)}$ and voltage values $V_{ij}^{(L)}$ and $V_{ji}^{(L)}$ (i.e. voltages on each side of line (i,j)) that match the corresponding feedback signals $S_{ij}^{(B)}$, $V_i^{(B)}$ and $S_{ji}^{(B)}$, $V_j^{(B)}$, computed by the buses i and j, respectively. The optimization is illustrated in Model 4. It describes four coupling constraints: two associated to the voltage values and two associated to the flow values (Eq. (32)). The model also ensures the voltages and power flows are within the

[2] To simplify the notations, $(\cdot)$ is used to represent the complex dot product.

Model 4 ADMM: Line agent $\mathcal{L}_{i,j}(P_{line})$

inputs: $\langle \rho, \lambda^S_{ij}, \lambda^V_{ij}, S^{(B)}_{ij}, V^{(B)}_{ij}, \lambda^S_{ji}, \lambda^V_{ji}, S^{(B)}_{ji}, V^{(B)}_{ji} \rangle$

variables: $S^{(L)}_{ij}, S^{(L)}_{ji}, V^{(L)}_{ij}, V^{(L)}_{ji}$

minimize:
$$\sum_{(e,f)\in\{(i,j),(j,i)\}} [\lambda^S_{ef} \cdot S^{(L)}_{ef} + \lambda^V_{ef} \cdot V^{(L)}_{ef} + \frac{\rho}{2}(\|S^{(L)}_{ef} - S^{(B)}_{ef}\|^2 + \|V^{(L)}_{ef} - V^{(B)}_{ef}\|^2)] \quad (26)$$

local constr.
$$\angle V^{(L)}_{ij} = 0, \text{ if } i = s; \angle V^{(L)}_{ji} = 0, \text{ if } j = s; \quad (27)$$
$$v^l_e \leqslant |V^{(L)}_{ef}| \leqslant v^u_e, \ \forall (e,f) \in \{(i,j),(j,i)\} \quad (28)$$
$$-\theta^{\Delta}_{ef} \leqslant \angle (V^{(L)}_{ef} V^{(L)*}_{fe}) \leqslant \theta^{\Delta}_{ef} \quad (29)$$
$$|S^{(L)}_{ef}| \leqslant s^u_{ef}, \ \forall (e,f) \in \{(i,j),(j,i)\} \quad (30)$$
$$S^{(L)}_{ef} = Y^*_{ef} |V^{(L)}_{ef}|^2 - Y^*_{ef} V^{(L)}_{ef} V^{(L)*}_{ef} \quad \forall (e,f) \in \{(i,j),(j,i)\} \quad (31)$$

Model 5 ADMM: Bus agent $\mathcal{B}_i(P_{bus})$

inputs: $\langle \rho, \ \lambda^d_i, S^{d(D)}_i, \lambda^g_i, S^{g(G)}_i \rangle$, $\langle \lambda^S_{ij}, S^{(L)}_{ij}, \lambda^V_{ij}, V^{(L)}_{ij} | \forall (i,j) \in E \cup E^R \rangle$

variables: $S^{d(B)}_i, S^{g(B)}_i, V^{(B)}_i, S^{(B)}_{ij} \forall (i,j) \in E \cup E^R$

minimize:
$$\lambda^d_i \cdot S^{d(B)}_i + \frac{\rho}{2}\|S^{d(B)}_i - S^{d(D)}_i\|^2 + \lambda^g_i \cdot S^{g(B)}_i + \frac{\rho}{2}\|S^{g(B)}_i - S^{g(G)}_i\|^2 + \sum_{(i,j)\in E\cup E^R} [\lambda^S_{ij} \cdot S^{(B)}_{ij} + \frac{\rho}{2}\|S^{(B)}_{ij} - S^{(L)}_{ij}\|^2 + \lambda^V_{ij} \cdot V^{(B)}_i + \frac{\rho}{2}\|V^{(B)}_i - V^{(L)}_{ij}\|^2] \quad (32)$$

local constraint:
$$S^{g(B)}_i - S^{d(B)}_i = \sum_{(i,j)\in E\cup E^R} S^{(B)}_{ij} \quad (33)$$

feasible bounds (Eqs. (28) to (30)), and that the AC power flow constraints are satisfied (Eq. (31)). The voltage angle $\angle V^{(L)}_{ij}/\angle V^{(L)}_{ji}$ is zero if it connects to a slack bus (Eq. (27)).

Bus agent At each iteration, bus agent i performs the optimization described in Model 5. Its objective is that of finding load value $S^{d(B)}_i$, generator value $S^{g(B)}_i$, voltage value $V^{(B)}_i$, and flow values $S^{(B)}_{ij}$, for each connecting line $(i,j) \in E \cup E^R$, that match the state variables sent from the load, generator, and line agents, respectively. The model also ensures the satisfaction of the flow balance constraint (Eq. (33)).

ADMM Coordination Process The ADMM algorithm that coordinates all agents is illustrated in Algorithm 1. Lines 1 to 3 initialize all variables associated with the load, generator, and line agents, respectively. Each of these agents, hence, perform their optimization step (lines 6 to 8), independently from one another, and send their (consensus) variables to the corresponding bus agents. Upon receiving the consensus variables from each of their neighboring agents, the bus agents perform their associated local optimization step and send the feedback values back to the corresponding load, generator, and line agents (line 10). Finally, the multipliers variables λ are updated by each individual agent (lines 12 to 14). At the end of each iteration, the parameter ρ can be updated. The algorithm

Algorithm 1: ADMM: Main routine

Inputs : $\langle \mathcal{N}, \rho_{init}, t_{max} \rangle$, $\langle \tilde{S}^d_i | \forall \mathcal{D}_i \rangle$, $\langle O^*_i | \forall \mathcal{G}_i \rangle$

1 $\rho \leftarrow \rho_{init}$

2 $\langle \lambda^d_i, S^{d(B)}_i \rangle \leftarrow \langle 0,0 \rangle \ \forall \mathcal{D}_i$; $\langle \lambda^g_i, S^{g(B)}_i \rangle \leftarrow \langle 0,0 \rangle \ \forall \mathcal{G}_i$;

3 $\langle \lambda^S_{ij}, S^{(B)}_{ij} \rangle \leftarrow \langle 0,0 \rangle \wedge \langle \lambda^V_{ij}, V^{(B)}_{ij} \rangle \leftarrow \langle 0,0 \rangle \ \forall (i,j) \in \mathcal{L}_{i,j}$

4 **for** $t = 1, 2, \ldots, t_{max}$ **do**

5 Optimization of load, generator, and line agents

6 $\forall \mathcal{D}_i : S^{d(D)}_i \leftarrow P_{load}(\langle \rho, \lambda^d_i, \tilde{S}^d_i, S^{d(B)}_i \rangle)$

7 $\forall \mathcal{G}_i : S^{g(G)}_i \leftarrow P_{gen}(\langle \rho, \lambda^g_i, O^*_i, S^{g(B)}_i \rangle)$

8 $\forall \mathcal{L}_{i,j} : S^{(L)}_{ij}, V^{(L)}_{ij}, S^{(L)}_{ji}, V^{(L)}_{ji} \leftarrow P_{line}(\langle \rho, \lambda^S_{ij}, \lambda^V_{ij}, S^{(B)}_{ij}, V^{(B)}_{ij}, \lambda^S_{ji}, \lambda^V_{ji}, S^{(B)}_{ji}, V^{(B)}_{ji} \rangle)$

9 Optimization of bus agents

10 $\forall \mathcal{B}_i : S^{d(B)}_i, S^{g(B)}_i, V^{(B)}_i, S^{(B)}_{ef} \leftarrow P_{bus}(\langle \rho, -\lambda^d_i, S^{d(D)}_i, -\lambda^g_i, S^{g(G)}_i \rangle, \langle -\lambda^S_{ef}, S^{(L)}_{ef}, -\lambda^V_{ef}, V^{(L)}_{ef} \rangle)$

11 Lagrange multiplier update

12 $\forall \mathcal{D}_i$ and $\mathcal{B}_i : \lambda^d_i \leftarrow \lambda^d_i + (S^{d(D)}_i - S^{d(B)}_i)$

13 $\forall \mathcal{G}_i$ and $\mathcal{B}_i : \lambda^g_i \leftarrow \lambda^g_i + (S^{g(G)}_i - S^{g(B)}_i)$

14 $\forall \mathcal{L}_{i,j}$ and $\mathcal{B}_i/\mathcal{B}_j : \lambda^S_{ij} \leftarrow \lambda^S_{ij} + (S^{(L)}_{ij} - S^{(B)}_{ij}), \lambda^V_{ij} \leftarrow \lambda^V_{ij} + (V^{(L)}_{ij} - V^{(B)}_i)$

15 Coordinating agent penalty ρ update (optional)

16 $\rho \leftarrow$ update_p()

Output : S^d_i

is executed for t_{max} iterations and the overall communication complexity is in $O(t_{max}(|N| + |E|))$.

The ADMM coordination process described above is similar to that introduced in [17]. However, differently from other proposals, that use ADMM for solving an OPF, the ADMM scheme used by PD-OPF is used for the distributed resolution of problem P_{RBL}. It redistributes the noise introduced by the Laplace mechanism optimally to satisfy the fidelity criteria.

Theorem *3:* PD-OPF satisfies local α-indistinguishability.

Proof. By Theorem 2, the load values obtained by the application of the Laplace mechanism satisfy α-local indistinguishability. The ADMM mechanism makes use of exclusively the privacy-preserved load values $\tilde{\boldsymbol{S}}^d$ (computed by the application of the Laplace mechanism), as well as additional public information (e.g. the local cost values $\mathcal{O}^*_i$). Therefore, by post-processing immunity of differential privacy, PD-OPF satisfies local α-indistinguishability. □

VI. Experimental Results

This section reports on the obfuscation quality and ability to converge of PD-OPF. Additionally, the proposed method is compared with a centralized version that solves problem P_{RBL}, thus admitting the presence of a centralized data curator. The experiments are performed on a variety of NESTA [26] benchmarks. Parameter ϵ is fixed to 1.0, the *indistinguishability level* α varies from 0.01 to 0.1 in p.u. (i.e. 1 MVA to 10 MVA), and the *fidelity level* β varies from 10^{-2} to 10^{-1} (i.e. from 1% to 10% of the optimal cost difference). PD-OPF is limited to use 5000 iterations. All the models are implemented using PowerModels.jl [27] in Julia with nonlinear solver IPOPT [28].

Choosing a fixed penalty factor ρ to drive convergence is challenging [17]. Thus, the experimental routine adjusts ρ dynamically, using the maximum primal and dual infeasibility

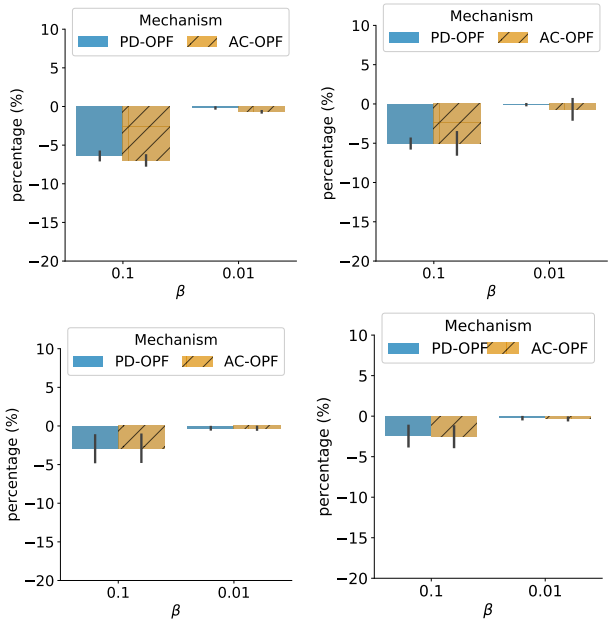

Fig. 3. Dispatch costs differences between the optimal and the PD-OPF solution (PD-OPF) and its centralized AC-OPF counterpart. IEEE 39 (top) & IEEE 57 bus (bottom), $\alpha = 0.01$ (left), 0.1 (right), $\beta = [0.1, 0.01]$. PD-OPF mechanism: blue bars; centralized AC-OPF: brown bars.

values, $\epsilon_p = \max \boldsymbol{r}_p$ and $\epsilon_d = \max \boldsymbol{r}_d$, respectively (in spirit of [17]). Higher values of ρ encourage the satisfaction of the primal constraints, while lower values shift weights to the objectives and reduce the dual infeasibilities [17]. The heuristic adopted changes ρ when the distance between ϵ_p and ϵ_d becomes too large:

$$\rho = \min\{(1+c)\rho, \overline{\rho}\}, \qquad \text{if } \epsilon_p > c_t \epsilon_d,$$
$$\rho = \max\{\frac{\rho}{1+c}, \underline{\rho}\}, \qquad \text{if } \epsilon_d > c_t \epsilon_p.$$

The scaling factor c is set to 2%, the threshold parameter c_t to 7.0, and upper $\overline{\rho}$ and lower $\underline{\rho}$ bounds to 10^6 and 5, respectively.

To allow PD-POPF to restore primal feasibility, a feasibility boosting procedure is implemented as follows. When the iteration counter 4500 iterations, if the maximum primal infeasibility is larger than 10^{-3}, ρ will be increased by: $\min\{(1+c)\rho, \overline{\rho}\}$. We call this phase *feasibility boosting*.

A. Quality of Fidelity Restoration

Let $\mathcal{O}^*$ and $\hat{\mathcal{O}}$ to be the optimal dispatch costs for the original and obfuscated loads respectively. Figure 3 illustrates the average percentage difference on the dispatch cost differences between the original and obfuscated loads produced by PD-OPF: $100 \times \frac{\hat{\mathcal{O}} - \mathcal{O}^*}{\mathcal{O}^*}$. Since a PD-OPF implements a relaxation of Constraint (18), the Figure also reports a comparison using a centralized procedure that solves an AC-OPF with the PD-OPF loads as input. The experimental results indicate that PD-OPF is able to restore the problem fidelity well, even when the fidelity requirement β are as small as 0.01% of the original costs.

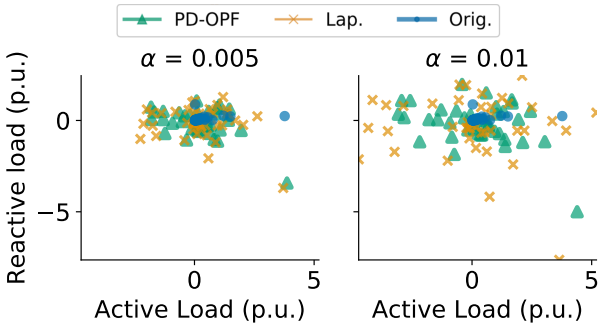

Fig. 4. Loads from the original dataset and the Laplace & the PD-OPF Mechanisms on the IEEE-57 bus system, at varying of the indistinguishability value $\alpha = 0.005$ (left) and $\alpha = 0.01$ (right) with $\beta = 0.1$. PD-OPF mechanism: green triangle; Laplace mechanism: yellow cross; Original: blue dot.

B. Quality of Load Demand Obfuscation

Figure 4 depicts the original load values (Orig.) associated to the IEEE-57 bus systems, and compares them with those generated by the Laplace mechanism (Lap.) and by PD-OPF. The figure illustrates that the post-processing step used in PD-OPF modifies the original loads. Since the Laplace mechanism does not converge to an AC feasible solution, PD-OPF further modifies the Laplace-generated loads. The figure does not report the AC-feasible loads due to large overlaps with PD-OPF values.

C. Quality of Privacy Loss Minimization

Figure 5 illustrates the difference between the loads produces by PD-OPF and those produces by a centralized implementation of problem P_{RBL} [5]. The difference is measured in terms of distance from the Laplace obfuscated loads (averaged over 50 instances). The differences in the IEEE-39 test case are due to the feasibility boosting phase, activated to improve the primal feasibility. In the other test cases the differences between the two approaches are negligible, thus validating the use of a decentralized solution for releasing loads when a centralized trusted data curator is unavailable.

D. Convergence Quality & Runtime

Finally, table II presents the maximum and dual infeasibilities (in p.u.), before and after (marked with $*$) activating the feasibility boosting procedure. The table clearly illustrates the benefits of the boosting procedure. It is able to reduce the primal infeasibility of up to two order of magnitude, albeit at a cost of a larger dual infeasibility.

Figure 6 illustrates the details of one run on the IEEE-39 benchmark. After a few iterations, both the primal and the dual infeasibilities stabilize in the range $[10^1, 10^{-1}]$ (top-left), and the generator costs stabilize after 2000 iterations (bottom-left). When the feasibility boosting is activated, the coordination agent increases the parameter ρ (bottom right), inducing all agents to re-optimize with a higher penalty for violating the coupling constraints. This is obtained at a cost of a larger dual feasibility (top-right).

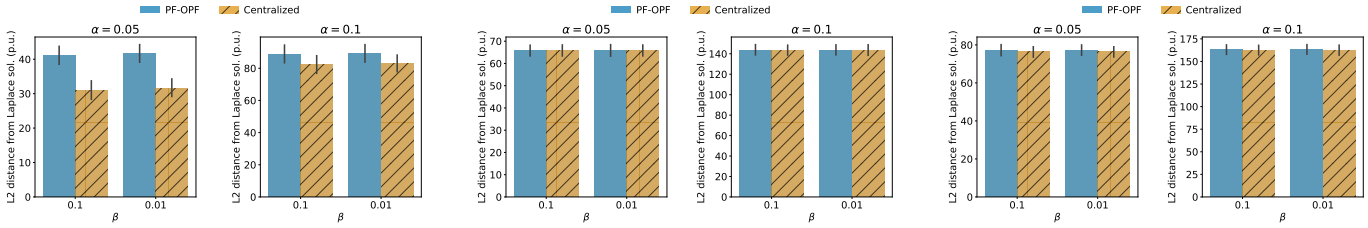

Fig. 5. L2 distance between the Laplace obfuscated data and the PD-OPF and the Centralized obfuscated data. IEEE-39 (left), IEEE-57 (center), IEEE-189 (right). $\alpha = [0.05, 0.1]$, $\beta = [0.01, 0.1]$. PD-OPF mechanism: blue bars; Centralized mechanism: brown bars.

TABLE II
PRIMAL & DUAL INFEASIBILITY, AND SIM. RUNTIME. $\alpha = 0.1, \beta = 0.1$.

	Primal	Primal∗	Dual	Dual∗	Time (min.)
nesta_case3_lmbd	0.036	0.001	0.173	0.079	1.147
nesta_case4_gs	0.023	0.001	0.092	13.953	2.110
nesta_case5_pjm	1.580	0.015	3.094	380.243	3.501
nesta_case6_c	0.203	0.001	0.835	7.088	2.607
nesta_case6_ww	0.094	0.001	0.419	7.919	3.215
nesta_case9_wscc	0.197	0.001	1.224	5.908	2.776
nesta_case14_ieee	0.579	0.001	2.228	19.762	5.141
nesta_case24_ieee_rts	0.293	0.006	1.276	540.403	11.157
nesta_case29_edin	0.216	0.128	2.393	3027.724	38.686
nesta_case30_as	0.386	0.001	1.685	15.223	12.592
nesta_case30_fsr	0.416	0.001	2.215	14.001	10.738
nesta_case30_ieee	0.621	0.001	2.831	37.137	11.167
nesta_case39_epri	0.291	0.026	1.597	1849.358	15.273
nesta_case57_ieee	0.584	0.001	2.951	62.776	19.614
nesta_case73_ieee_rts	0.402	0.008	2.762	691.576	45.131
nesta_case118_ieee	0.968	0.004	4.427	394.885	82.160
nesta_case189_edin	3.214	0.017	14.780	871.245	86.908

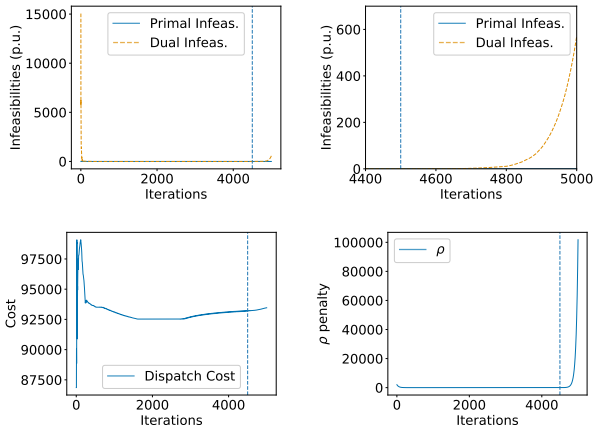

Fig. 6. IEEE-39 bus: Primal ϵ_p and dual ϵ_d infeasibilities (Full-scale: top left, $>$ 4400 iterations: top right), generator dispatch costs (bottom left), penalty ρ (bottom right); $\alpha = 0.1$, $\beta = 0.1$. The vertical dotted line marks the activation of the boosting procedure.

VII. CONCLUSION

This paper presented a framework for releasing privacy-preserving OPF load data of a decentralized operated power system. The proposed framework, called Privacy-preserving Distributed OPF (PD-OPF), is based on the Alternating Direction Method of Multipliers (ADMM) and satisfies the notion of differential privacy to guarantee strong privacy protection for the customer loads while also ensuring that the released data retains high fidelity and satisfies the AC power flow constraints. The key component of PD-OPF is a distributed optimization procedure that redistributes the noise introduced by traditional DP algorithms to satisfy the desired properties. Extensive experimental evaluations on the NESTA benchmark showed that the PD-OPF can be used to generate privacy-preserving data providing high-quality AC-feasible solutions and that the results attained are comparable with those obtained by a centralized routine approach.

A fine-tuning of the ADMM parameters, including the effect of variable initialization will be subject of future investigation.

APPENDIX

A. PD-OPF with the Piecewise Mechanism

The PD-OPF framework can also be extended to work with other Local Differential Privacy mechanism (LDP). Instead of using Polar Laplace mechanism in the Privacy Phase, this section showcases another LDP mechanism: the Piecewise Mechanism [29]. The Piecewise Mechanism also satisfies the $\boldsymbol{L}^1\epsilon$-LDP for α distances definition. It requires all input data x_i to be normalized within $[-1, 1]$ from $[\underline{x_i}, \overline{x_i}]$. Let:

$$C = \frac{e^{\epsilon/2\alpha} + 1}{e^{\epsilon/2\alpha} - 1},$$
$$L(x_i) = \frac{C+1}{2} x_i - \frac{C-1}{2}, \text{ and}$$
$$R(x_i) = L(x_i) + C - 1.$$

The mechanism perform obfuscation based as in Algorithm 2. To implement the Piecewise Mechanism, linear transformations are used by each of the load agents $\mathcal{D}_i$ to normalize active and reactive parts of the load value S_i^d into [-1,1]. To transform from a bounded domain $x_i \in [\underline{x_i}, \overline{x_i}]$ to $y_i \in [-1, 1]$ (and vice versa), the following equation is used: $y_i = 2\frac{x_i - \underline{x_i}}{\overline{x_i} - \underline{x_i}} - 1$.

Table III shows the primal and dual convergence quality and simulation runtime similar as in previous section. Figure 7

Algorithm 2: Piecewise Mechanism for LDP

Sample $p \sim$ Uniform($[0, 1]$)
if $p \leqslant \frac{e^{\epsilon/2\alpha}}{e^{\epsilon/2\alpha}+1}$ **then**
  Sample $\tilde{x}_i \sim$ Uniform($[L(x_i), R(x_i)]$)
else
  Sample $\tilde{x}_i \sim$ Uniform($[-C, L(x_i)] \cup [R(x_i), C]$)
Return $\tilde{x}_i$

shows the fidelity can again be restored by the ADMM mechanism. Figure 8 shows comparable obfuscation quality when comparing to the Laplace mechanism in Figure 4. Finally, Figure 9 shows the ADMM algorithm can achieve comparable privacy loss minimization results to centralized optimization.

TABLE III
PRIMAL AND DUAL INFEASIBILITY, AND SIMULATION RUNTIME.
$\alpha = 0.1, \beta = 0.1$.

	Primal	Primal∗	Dual	Dual∗	Time (min.)
nesta_case3_lmbd	0.001	0.001	0.015	0.015	0.089
nesta_case4_gs	0.031	0.001	0.151	11.733	3.505
nesta_case5_pjm	1.820	0.015	3.290	382.929	3.416
nesta_case6_c	0.006	0.001	0.038	0.180	0.479
nesta_case6_ww	0.217	0.072	1.064	20667.869	4.165
nesta_case9_wscc	0.023	0.001	0.119	1.596	1.445
nesta_case14_ieee	0.085	0.001	0.392	5.402	8.722
nesta_case24_ieee_rts	0.133	0.008	0.859	611.856	11.294
nesta_case29_edin	0.197	0.098	2.676	3810.460	82.134
nesta_case30_as	0.161	0.001	0.847	5.090	9.244
nesta_case30_fsr	0.050	0.001	0.250	1.525	9.824
nesta_case30_ieee	0.211	0.001	1.074	9.788	9.714
nesta_case39_epri	0.920	0.020	4.371	1029.756	37.142
nesta_case57_ieee	1.201	0.001	4.947	104.336	40.772
nesta_case73_ieee_rts	0.219	0.011	1.654	777.139	43.815
nesta_case189_edin	1.432	0.016	6.904	799.463	83.319

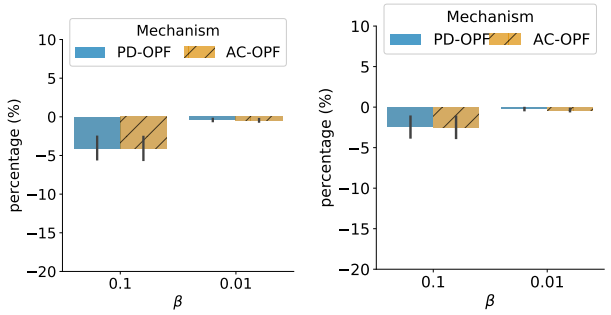

Fig. 7. IEEE 57 bus. Percentage Difference on the Dispatch Costs after ADMM mechanism and AC validation: α = 0.01 (left), 0.1 (right), β=[0.1,0.01]. Average over 50 instances. PD-OPF: blue bars; AC-OPF: brown bars.

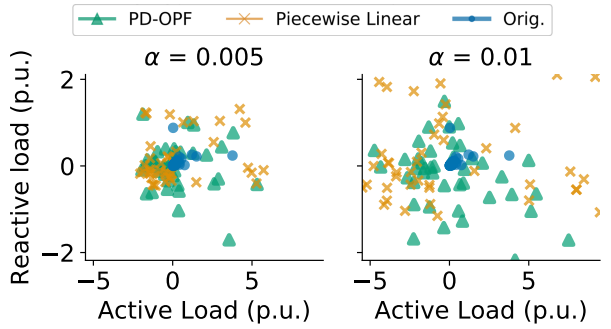

Fig. 8. Loads from the original dataset and the Piecewise Linear & ADMM Mechanisms on the IEEE-57 bus system, at varying of the indistinguishability value α = 0.005 (left) and α = 0.01 (right) with β = 0.1. PD-OPF mechanism: green triangle; Piecewise Linear mechanism: yellow cross; Original: blue dot.

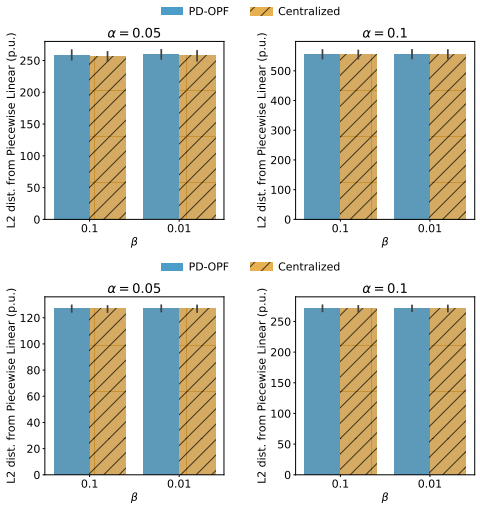

Fig. 9. L2 distance between ADMM/Centralized Mechanisms and Piecewise Linear Obfuscated Dataset on the IEEE-39 (top) and IEEE-57 (bottom) bus systems, with $\alpha = [0.05, 0.1]$, and $\beta = [0.01, 0.1]$. PD-OPF: blue bars; AC-OPF: brown bars.

REFERENCES

[1] F. Fioretto, T. W. K. Mak, and P. V. Hentenryck, "Privacy-preserving obfuscation of critical infrastructure networks," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2019, pp. 1086–1092.

[2] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *TCC*, vol. 3876. Springer, 2006, pp. 265–284.

[3] T. Mak, F. Fioretto, L. Shi, and P. Van Hentenryck, "Privacy-preserving power system obfuscation: A bilevel optimization approach," *IEEE Transactions on Power Systems*, vol. 35, no. 2, pp. 1627–1637, March 2020.

[4] F. Fioretto, T. W. K. Mak, and P. Van Hentenryck, "Differential privacy for power grid obfuscation," *IEEE Transactions on Smart Grid*, vol. 11, no. 2, pp. 1356–1366, March 2020.

[5] F. Fioretto and P. Van Hentenryck, "Constrained-based differential privacy: Releasing optimal power flow benchmarks privately," in *Proceedings of Integration of Constraint Programming, Artificial Intelligence, and Operations Research (CPAIOR)*, 2018, pp. 215–231.

[6] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Theoretical Computer Science*, vol. 9, no. 3-4, pp. 211–407, 2013.

[7] S. Vadhan, "The complexity of differential privacy," in *Tutorials on the Foundations of Cryptography*. Springer, 2017, pp. 347–450.

[8] G. Ács and C. Castelluccia, "I have a dream!(differentially private smart metering)." in *Information hiding*, vol. 6958. Springer, 2011, pp. 118–132.

[9] J. Zhao, T. Jung, Y. Wang, and X. Li, "Achieving differential privacy of data disclosure in the smart grid," in *INFOCOM, 2014 Proceedings IEEE*. IEEE, 2014, pp. 504–512.

[10] X. Liao, P. Srinivasan, D. Formby, and A. R. Beyah, "Di-prida: Differentially private distributed load balancing control for the smart grid," *IEEE Transactions on Dependable and Secure Computing*, 2017.

[11] A. Halder, X. Geng, P. R. Kumar, and L. Xie, "Architecture and algorithms for privacy preserving thermal inertial load management by a load serving entity," *IEEE Transactions on Power Systems*, vol. 32, no. 4, pp. 3275–3286, July 2017.

[12] F. Zhou, J. Anderson, and S. H. Low, "Differential privacy of aggregated DC optimal power flow data," *arXiv preprint arXiv:1903.11237*, 2019.

[13] T. Zhang and Q. Zhu, "Dynamic differential privacy for ADMM-based distributed classification learning," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 1, pp. 172–187, 2016.

[14] Z. Huang, R. Hu, Y. Guo, E. Chan-Tin, and Y. Gong, "DP-ADMM: ADMM-based distributed learning with differential privacy," *IEEE Transactions on Information Forensics and Security*, 2019.

[15] J. Ding, Y. Gong, M. Pan, and Z. Han, "Optimal differentially private ADMM for distributed machine learning," *arXiv preprint arXiv:1901.02094*, 2019.
[16] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein *et al.*, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.
[17] S. Mhanna, G. Verbič, and A. C. Chapman, "Adaptive admm for distributed ac optimal power flow," *IEEE Transactions on Power Systems*, vol. 34, no. 3, pp. 2025–2035, May 2019.
[18] J. M. Abowd, "The us census bureau adopts differential privacy," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 2867–2867.
[19] F. Fioretto and P. V. Hentenryck, "Differential privacy of hierarchical census data: An optimization approach," in *Principles and Practice of Constraint Programming - 25th International Conference, CP*, 2019, pp. 639–655.
[20] G. Fanti, V. Pihur, and Ú. Erlingsson, "Building a rappor with the unknown: Privacy-preserving learning of associations and data dictionaries," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 3, pp. 41–61, 2016.
[21] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2006, pp. 486–503.
[22] K. Chatzikokolakis, M. E. Andrés, N. E. Bordenabe, and C. Palamidessi, "Broadening the scope of differential privacy using metrics," in *International Symposium on Privacy Enhancing Technologies Symposium*. Springer, 2013, pp. 82–102.
[23] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: Differential privacy for location-based systems," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 2013, pp. 901–914.
[24] A. Sinha, P. Malo, and K. Deb, "A review on bilevel optimization: from classical to evolutionary approaches and applications," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 2, pp. 276–295, 2018.
[25] S. Mhanna, A. C. Chapman, and G. Verbič, "Component-based dual decomposition methods for the opf problem," *Sustainable Energy, Grids and Networks*, vol. 16, pp. 91 – 110, 2018.
[26] C. Coffrin, D. Gordon, and P. Scott, "Nesta, the NICTA energy system test case archive," *CoRR*, vol. abs/1411.0359, 2014. [Online]. Available: http://arxiv.org/abs/1411.0359
[27] C. Coffrin, R. Bent, K. Sundar, Y. Ng, and M. Lubin, "Powermodels.jl: An open-source framework for exploring power flow formulations," in *PSCC*, June 2018.
[28] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.
[29] N. Wang, X. Xiao, Y. Yang, J. Zhao, S. C. Hui, H. Shin, J. Shin, and G. Yu, "Collecting and analyzing multidimensional data with local differential privacy," in *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 2019, pp. 638–649.