Fast Inverter Control by Learning the OPF Mapping using Sensitivity-Informed Gaussian Processes

Mana Jalali, *Graduate Student Member, IEEE*, Manish K. Singh, *Member, IEEE*, Vassilis Kekatos, *Senior Member, IEEE*, Georgios B. Giannakis, *Fellow, IEEE*, and Chen-Ching Liu, *Fellow, IEEE*,

Abstract—**Fast inverter control is a desideratum towards the smoother integration of renewables. Adjusting inverter injection setpoints for distributed energy resources can be an effective grid control mechanism. However, finding such setpoints optimally requires solving an optimal power flow (OPF), which can be computationally taxing in real time. Previous works have proposed learning the mapping from grid conditions to OPF minimizers using Gaussian processes (GPs). This GP-OPF model predicts inverter setpoints when presented with a new instance of grid conditions. Training enjoys closed-form expressions, and GP-OPF predictions come with confidence intervals. To improve upon data efficiency, we uniquely incorporate the sensitivities (partial derivatives) of the OPF mapping into GP-OPF. This expedites the process of generating a training dataset as fewer OPF instances need to be solved to attain the same accuracy. To further reduce computational efficiency, we approximate the kernel function of GP-OPF leveraging the concept of random features, which is neatly extended to sensitivity data. We perform sensitivity analysis for the second-order cone program (SOCP) relaxation of the OPF, whose sensitivities can be computed by merely solving a system of linear equations. Extensive numerical tests using real-world data on the IEEE 13- and 123-bus feeders corroborate the merits of GP-OPF.**

Index Terms—**Gaussian processes; second-order cone program; sensitivity analysis; random features; learning-to-optimize.**

I. INTRODUCTION

Rapid fluctuations in power injections by solar photovoltaics and other distributed energy resources (DERs) induce undesirable voltage deviations in distribution grids. Reactive power compensation and active power curtailment by the smart inverters interfacing DERs have been suggested as an effective fast-responding voltage control mechanism. Nonetheless, finding the optimal power injection setpoints for hundreds of inverters is a computationally formidable task. It requires solving the OPF in near real time to account for varying solar and loading conditions. To expedite optimal inverter control, this work aims at learning the OPF mapping using GPs.

Per the IEEE 1547 standard, inverter setpoints can be selected upon Volt-VAR, Watt-VAR, or Volt-Watt curves driven by local data [1]. Although such rules have been shown to be stable, their equilibria may not be optimal [2], [3], or even perform worse than the no-reactive support option [4]. Given the current grid conditions, inverter setpoints can be optimally decided upon solving an OPF. Albeit non-convex, the OPF can be relaxed to a convex second-order cone program (SOCP) or a semidefinite program (SDP); see [5] for a survey. To reduce computational time, one may resort to a linearized feeder model and trade modeling accuracy for complexity, to eventually express the OPF as a linear or quadratic program (LP/QP); see e.g., [6] and references therein.

To further expedite optimal inverter control, recent works rely on machine learning (ML) techniques to shift some of the computational burden of the OPF from real time to offline. According to this *learning-to-optimize* paradigm, kernel–based regression has been utilized to learn inverter control rules in [7] and [8]. Deep neural networks (DNNs) have been trained to predict OPF solutions under linearized [9], [10], and the exact AC grid models [11], [12]. The utility may be interested in the average or probabilistic performance of inverters across a range of uncertain grid conditions. Under this stochastic setup, inverter control rules have been obtained using reinforcement learning strategies [13], [14]. References [15] and [16] develop inverter control rules driven by incomplete or noisy data, by training a DNN using primal/dual updates based on the Lagrangian function of a stochastic OPF.

Under the deterministic setup, learning-to-optimize schemes proceed in two steps: *s1)* They first solve a large number of OPFs to build a labeled dataset; and *s2)* Train the ML model. Although it occurs offline, training has to be repeated afresh with any alteration of the OPF structure due to a topology reconfiguration or when a large DER goes offline. To expedite *s1)*, in [17] and [18], we proposed training a DNN to match not only the OPF solutions, but also their partial derivatives with respect to grid conditions. Owing to this sensitivity-informed training, the DNN attained the same prediction accuracy using a much smaller training dataset. Therefore step *s1)* required solving fewer OPF instances. Reference [19] trained DNNs to learn OPF minimizers by properly penalizing optimality conditions. The latter approach was combined with sensitivity-informed learning in [20]. Reference [21] leverages multi-parametric programming to classify OPF inputs and design DNNs with reduced complexity for each class. In any case, training a DNN during *s2)* itself requires solving an optimization. Moreover, DNN-based OPF predictions come

without confidence intervals. To overcome these shortcomings, we rely on modeling the OPF mapping using GPs previously suggested in [22] for dealing with a probabilistic OPF.

The motivation for using GPs is twofold: First, the weights of a GP model can be computed in closed-form, while the few kernel parameters involved are found by solving small-scale minimizations. Second, by virtue of its Bayesian nature, a GP model not only predicts an OPF minimizer, but can also quantify the uncertainty of such prediction in the form of variance. This is important as it may warn the operator not to trust a specific prediction and opt for solving one more instance of the OPF instead. Uncertainty quantification can also be used to identify undersampled areas of the parameter space of the OPF, or areas where the OPF mapping is more complex. Then, additional samples from those areas can be drawn to be solved to strategically enrich the dataset.

GPs have been utilized in the power systems literature before. For example, [23] and [24] infer frequency oscillations from synchrophasor data in transmission systems using GPs. For distribution grids, the inverse power flow mapping from power injections to voltages in active distribution grids is modeled as a GP in [25]. Reference [26] pursues inverter-based voltage control by approximating voltages as affine functions of injections using a GP having a linear kernel. GP learning has also been used previously for uncertainty propagation through the OPF in [22]. Building on [22], we suggest learning the deterministic OPF mapping in a physics-informed manner by uniquely incorporating the sensitivities of the OPF solutions with respect to problem parameters, i.e., the grid conditions.

While the proposed use of OPF sensitivities for improving GP-OPF estimates is novel, there has been significant interest in computing such sensitivities for other applications [27], [28]. These works exploited OPF sensitivities to efficiently compute OPF minimizers and look into binding constraints for a given trajectory of load variations. Thus, these works confined their focus on scalar parameterization of loads. Beyond the power systems literature, extensive developments have been reported in the general area of sensitivity analysis of continuous optimization problems [29], [30]. Building upon these seminal works, and relaxing some of their underlying assumptions, a convenient approach for sensitivity analysis of the OPF has been recently proposed in [18]. However, the developed approach applies to continuous optimization problems with twice-differentiable *scalar* constraint functions. Although differentiating through convex cone constraints is possible [31], the approach gets more involved. Fortunately, simple reformulations allow us for the first time to compute the sensitivities for the minimizers of the SOCP-based OPF.

Adopting the idea of [22] from the probabilistic to the deterministic setup, this work learns the deterministic OPF mapping using GPs for near-optimal real-time inverter control. As with [22], during training, the GP-OPF model is computed in closed-form. During operation, GP-OPF provides point predictions and confidence intervals for OPF solutions also in closed-form (Section III). Beyond the application setup, the technical contribution of this work is on three fronts:

i) We incorporate the sensitivities of OPF solutions with respect to OPF parameters to arrive at a *sensitivity-informed* GP-OPF (SI-GP-OPF). It essentially augments labeled data per solved OPF instance, and can thus attain the same prediction accuracy given a smaller dataset. It thus reduces the number of OPFs to be solved during training (Section IV).

ii) To reduce computational complexity, we approximate kernel functions using the concept of *random features (RF)* and obtain an RF-based GP-OPF (RF-GP-OPF). Random features are neatly extended to OPF sensitivities to derive an RF-SI-GP-OPF (Section V).

iii) We perform sensitivity analysis of the SOCP-OPF with respect to load demand and solar generation. Finding the sensitivities of OPF solutions is as easy as solving a system of linear equations (Section VI).

Extensive numerical tests on the IEEE 13- and 123-bus feeders corroborate our findings (Section VII).

Notation: Column vectors (matrices) are denoted by lower-(upper-) case letters. Symbol $(\cdot)^\top$ stands for transposition; $\mathbf{I}_N$ is the $N \times N$ identity matrix; $\mathbb{E}$ is the expectation operator; and $\|\mathbf{x}\|$ is the ℓ_2-norm of vector $\mathbf{x}$.

II. OPTIMAL INVERTER CONTROL

Consider a distribution feeder with $N + 1$ buses hosting a combination of inelastic loads and DERs. Buses are indexed by set $\mathcal{N} := \{1, \ldots, N\}$, while the substation is indexed as $n = 0$. Suppose there are N_g buses hosting DERs and their indexes are collected in set $\mathcal{N}_g \subseteq \mathcal{N}$. Let v_n denote the squared voltage magnitude at bus n, and $p_n + jq_n$ the complex power injected at bus n. Power injections at buses in $\mathcal{N}_g$ can be decomposed to controllable inverter-interfaced DER generation and uncontrollable loads as

$$p_n = p_n^g - p_n^\ell \quad \text{and} \quad q_n = q_n^g - q_n^\ell, \quad \forall n \in \mathcal{N}_g. \quad (1)$$

Given load demands $\{(p_n^\ell, q_n^\ell)\}_{n \in \mathcal{N}}$ at all buses, the task of optimal inverter control amounts to finding the setpoints $\{(p_n^g, q_n^g)\}_{n \in \mathcal{N}_g}$ for inverter power injections to minimize a desirable objective while adhering to feeder and inverter ratings. Before particularizing the related optimization problem, we briefly review the *DistFlow* model [6]. This model applies to radial single-phase feeders, but can approximate radial three-phase primary networks under nearly balanced operation.

On a radial single-phase feeder, each bus n has a unique parent π_n and a set of children buses $\mathcal{C}_n$. The line connecting bus n to its parent bus π_n is indexed by n. For line n, the impedance is denoted by $r_n + jx_n$; the squared magnitude of its current by ℓ_n; and the complex power flow seen at π_n by $P_n + jQ_n$. The feeder topology is assumed known and remains fixed. According to *DistFlow*, the feeder is governed by the ensuing equations for all $n \in \mathcal{N}$ [6]:

$$p_n = \sum_{k \in \mathcal{C}_n} P_k - (P_n - r_n \ell_n) \quad (2a)$$

$$q_n = \sum_{k \in \mathcal{C}_n} Q_k - (Q_n - x_n \ell_n) \quad (2b)$$

$$v_n = v_{\pi_n} + (r_n^2 + x_n^2)\ell_n - 2(r_n P_n + x_n Q_n) \quad (2c)$$

$$\ell_n = \frac{P_n^2 + Q_n^2}{v_{\pi_n}}. \quad (2d)$$

Given the aforesaid model, we next pose a possible rendition of the optimal inverter control task. A utility could determine inverter setpoints by solving the OPF problem:

$$\min \quad \sum_{n=1}^{N} (p_n + r_n \ell_n) \tag{3a}$$

$$\text{over} \quad \mathbf{x} := \left\{ \{p_n^g, q_n^g\}_{n \in \mathcal{N}_g}, \{P_n, Q_n, v_n, \ell_n\}_{n=1}^{N}, v_0 \right\} \tag{3b}$$

$$\text{s.to} \quad (1), (2a) - (2c) \tag{3c}$$

$$\left\| \begin{bmatrix} 2P_n \\ 2Q_n \\ v_{\pi_n} - \ell_n \end{bmatrix} \right\| \leq \ell_n + v_{\pi_n}, \quad n \in \mathcal{N} \tag{3d}$$

$$\ell_n \leq \bar{\ell}_n, \quad n \in \mathcal{N} \tag{3e}$$

$$\underline{v}_n \leq v_n \leq \bar{v}_n, \quad n \in \mathcal{N} \tag{3f}$$

$$0 \leq p_n^g \leq \bar{p}_n^g, \quad n \in \mathcal{N}_g \tag{3g}$$

$$(p_n^g)^2 + (q_n^g)^2 \leq (\bar{s}_n^g)^2, \quad n \in \mathcal{N}_g \tag{3h}$$

over optimization variable $\mathbf{x}$. Problem (3) aims at minimizing the total active power flowing via the substation to the feeder including the ohmic losses along all lines. The second-order cone (SOC) constraint in (3d) constitutes the widely adopted convex relaxation of (2d) [32]. For several renditions of the OPF, this relaxation has been shown to be *exact* [5], that is (3d) holds with equality at optimality for all n. Constraint (3e) ensures currents remain below the prescribed line ampacities $\bar{\ell}_n$. Constraint (3f) maintains voltages within a given range $[\underline{v}_n, \bar{v}_n]$ for all $n \in \mathcal{N}$. Constraint (3g) limits the active power generated by inverter n to remain smaller than or equal to the maximum possible value $\bar{p}_n^g$, which varies across time as it depends on the currently experienced solar irradiance. Finally, constraint (3h) limits the apparent power of inverter n depending on its kVA rating $\bar{s}_n^g$. Although this is a convex quadratic constraint, we cast it in the SOC form $\| [p_n^g \ q_n^g]^\top \| \leq \bar{s}_n^g$ to unify the exposition. To keep the presentation uncluttered, it is assumed that each bus hosts at most one inverter.

The OPF in (3) constitutes a *parametric* optimization problem, which has to be solved every time load demands and solar generation change. Collect the OPF parameters in vector

$$\boldsymbol{\theta} := \left(\{p_n^\ell, q_n^\ell\}_{n=1}^N, \{\bar{p}_n^g\}_{n \in \mathcal{N}_g} \right). \tag{4}$$

The length of this parameter vector is $M := 2N + N_g$. We can now pose (3) as the parametric SOCP

$$\mathbf{x}_{\boldsymbol{\theta}} := \arg\min_{\mathbf{x}} \quad \mathbf{c}^\top \mathbf{x} \tag{5a}$$

$$\text{s.to} \quad \mathbf{A}_e \mathbf{x} = \mathbf{B}_e \boldsymbol{\theta} + \mathbf{f}_e \tag{5b}$$

$$\mathbf{A}_i \mathbf{x} \leq \mathbf{B}_i \boldsymbol{\theta} + \mathbf{f}_i \tag{5c}$$

$$\|\mathbf{A}_m \mathbf{x}\| \leq \mathbf{b}_m^\top \mathbf{x} + f_m, \ m = 1 : 2N. \tag{5d}$$

Constraint (5b) collects the equality constraints in (3c). Constraint (5c) collects the inequality constraints in (3e)–(3g). Constraint (5d) collects constraints (3d), (3h). The involved matrices $(\mathbf{A}_e, \mathbf{A}_i, \mathbf{A}_m, \mathbf{B}_e, \mathbf{B}_i)$, vectors $(\mathbf{c}, \mathbf{f}_e, \mathbf{f}_i, \mathbf{b}_m)$, and scalars f_m follow directly from (3). Let $\mathbf{x}_{\boldsymbol{\theta}}$ denote the minimizer of the OPF associated with parameter $\boldsymbol{\theta}$.

The goal of this work is to learn the *mapping* $\boldsymbol{\theta} \to \mathbf{x}_{\boldsymbol{\theta}}$ induced by the SOCP-based OPF in (5). We would like to train a machine learning model that once presented a $\boldsymbol{\theta}$, it predicts

the associated minimizer $\mathbf{x}_{\boldsymbol{\theta}}$. Such model is useful in different applications. For example, predictions of $\mathbf{x}_{\boldsymbol{\theta}}$ can be directly used as setpoints, thus accelerating the task of inverter control. Alternatively, they can be used to warm-start an OPF solver. OPF predictions can also be used in hosting capacity analyses where a system operator studies different levels of renewable integration and the approximate effect of inverter control. Depending on the application, quantifying the uncertainty for any given OPF prediction can be also important. Training such model involves three phases: *i)* Creating a labeled dataset by solving (5) for different $\boldsymbol{\theta}$'s to find the related $\mathbf{x}_{\boldsymbol{\theta}}$'s; *ii)* Training a learning model using the labeled dataset offline; and *iii)* Using the trained model in real-time to predict OPF solutions.

III. MODELING THE OPF MAPPING AS A GP

Reference [22] suggested modeling mapping $\boldsymbol{\theta} \to \mathbf{x}_{\boldsymbol{\theta}}$ as a GP to deal with a probabilistic OPF, i.e., to efficiently approximate empirical histograms of OPF solutions over randomly sampled grid conditions $\boldsymbol{\theta}$'s. Spurred by [22], we propose a GP-OPF model for expediting real-time near-optimal inverter control. This section reviews GP-OPF from [22] under the inverter control setup. Over the following sections we improve upon [22] towards: *1)* including OPF sensitivities to enhance data efficiency; *2)* implementing GP using random features to improve on computational complexity during training; and *3)* accomplishing sensitivity analysis of the SOCP-based OPF.

Towards learning the OPF mapping, the entries of $\mathbf{x}_{\boldsymbol{\theta}}$ are learned independently. We henceforth focus on a particular entry, say the injection q_n^g by inverter n. To simplify notation, we denote this entry as $y(\boldsymbol{\theta})$. By sampling T loading conditions $\{\boldsymbol{\theta}_t\}_{t=1}^T$, we solve (5) to optimality. We have thus constructed a labeled training dataset $\mathcal{T} := \{(\boldsymbol{\theta}_t, y(\boldsymbol{\theta}_t))\}_{t=1}^T$. Using this dataset, our goal is to learn the function $y : \mathbb{R}^M \to \mathbb{R}$ so we are able to predict $y(\boldsymbol{\theta})$ for unseen values of $\boldsymbol{\theta}$. The function $y(\boldsymbol{\theta})$ will be learned using GP regression, which is GP-OPF as briefly review next.

GP-OPF relies on a key property of the multivariate Gaussian probability density function (PDF). Consider a random vector $\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ drawn from a Gaussian PDF with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. Partition vector $\mathbf{y}$ into two subvectors as

$$\mathbf{y} = \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} \boldsymbol{\mu}_1 \\ \boldsymbol{\mu}_2 \end{bmatrix}, \begin{bmatrix} \boldsymbol{\Sigma}_{11} & \boldsymbol{\Sigma}_{21}^\top \\ \boldsymbol{\Sigma}_{21} & \boldsymbol{\Sigma}_{22} \end{bmatrix} \right) \tag{6}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ have been partitioned conformably. Because subvectors $\mathbf{y}_1$ and $\mathbf{y}_2$ are jointly Gaussian, the conditional PDF of $\mathbf{y}_2$ given $\mathbf{y}_1$ is also Gaussian with mean and covariance

$$\mathbb{E}[\mathbf{y}_2|\mathbf{y}_1] = \boldsymbol{\mu}_2 + \boldsymbol{\Sigma}_{21} \boldsymbol{\Sigma}_{11}^{-1} (\mathbf{y}_1 - \boldsymbol{\mu}_1) \tag{7a}$$

$$\text{Cov}[\mathbf{y}_2|\mathbf{y}_1] = \boldsymbol{\Sigma}_{22} - \boldsymbol{\Sigma}_{21} \boldsymbol{\Sigma}_{11}^{-1} \boldsymbol{\Sigma}_{21}^\top. \tag{7b}$$

The implication is that if $\mathbf{y}_1$ is known and $\mathbf{y}_2$ is not, then $\mathbb{E}[\mathbf{y}_2|\mathbf{y}_1]$ provides an estimate for $\mathbf{y}_2$. This is in fact the minimum mean squared error (MMSE) estimate of $\mathbf{y}_2$. In addition to the point estimate of (7a), the covariance $\text{Cov}[\mathbf{y}_2|\mathbf{y}_1]$ quantifies the uncertainty of this estimate, which can be used to provide confidence intervals [33, Ch. 2], [34, Ch. 5].

The OPF mapping can be learned by modeling $y(\boldsymbol{\theta})$ as a GP over $\boldsymbol{\theta}$ as suggested in [22]. A random process is a GP if a

collection of any number of samples forms a Gaussian random vector [33, Ch. 1]. In other words, function $y(\boldsymbol{\theta})$ is a GP if any vector $\mathbf{y}$ having entries $y(\boldsymbol{\theta}_i)$ over any collection of $\boldsymbol{\theta}_i$'s follows a Gaussian PDF as in (6). The idea is to let subvector $\mathbf{y}_1$ collect the already computed OPF solutions $\{y(\boldsymbol{\theta}_t)\}_{t=1}^T$ from dataset $\mathcal{T}$, and subvector $\mathbf{y}_2$ collect the solutions we would like to infer and correspond to parameter vectors $\mathcal{S} := \{\boldsymbol{\theta}_s\}_{s=1}^S$. Thanks to (7), we can use the training dataset $\mathcal{T}$ to predict OPF decisions over any testing dataset $\mathcal{S}$.

For (6)–(7) to be useful for any reasonable $\boldsymbol{\theta}$'s in $\mathcal{T}$ and $\mathcal{S}$, GP regression parameterizes the mean and covariance of $y(\boldsymbol{\theta})$ as a function of $\boldsymbol{\theta}$. The mean is typically modeled as zero, that is $\mu(\boldsymbol{\theta}_i) = 0$ for all $\boldsymbol{\theta}_i$. This is without loss of generality as explained in [33]. As for the covariance matrix $\boldsymbol{\Sigma}$, note that its (i,j)-th entry corresponds to the covariance $\mathbb{E}[y(\boldsymbol{\theta}_i)y(\boldsymbol{\theta}_j)]$. In GP-OPF, the latter is assumed to be described as $\mathbb{E}[y(\boldsymbol{\theta}_i)y(\boldsymbol{\theta}_j)] = k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$, where $k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ is a *kernel function* measuring the similarity between any two parameter vectors. A common choice is the Gaussian kernel

$$k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = \alpha e^{-\frac{\beta}{2}\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_j\|^2} \tag{8}$$

for positive α and β. The Gaussian kernel is a *shift-invariant* kernel as it measures the similarity between two vectors as a function of their distance alone as $k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = k(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j)$.

Although the kernel captures the covariance of the actual function, vector $\mathbf{y}_1$ entails observing functions under noise. In our learning-to-optimize setup, the OPF labels are not corrupted by measurement noise (unless the solver was terminated prematurely). However, they do come with modeling noise as the postulated GP model may not be able to match the OPF mapping perfectly. Modeling noise is assumed to be drawn independently from a zero-mean Gaussian PDF with variance $\gamma > 0$. Then, the (i,j)-th entry of $\boldsymbol{\Sigma}_{11}$ in (7) is given by

$$[\boldsymbol{\Sigma}_{11}]_{i,j} = k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) + \gamma\delta_{ij} \tag{9}$$

where δ_{ij} is the Kronecker delta function. Parameters (α, β, γ) can be found via maximum likelihood estimation (MLE) using dataset $\mathcal{T}$; see [33, Ch. 5]. This is possible since $\boldsymbol{\Sigma}_{11}$ and $\mathbf{y}_1 \sim \mathcal{N}(\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_{11})$ depend on (α, β, γ) per (8).

Remark 1. *The mapping $y(\boldsymbol{\theta})$ is deterministic: Given $\boldsymbol{\theta}$, the setpoint $y(\boldsymbol{\theta})$ can be found by solving (5) for the requested $\boldsymbol{\theta}$. Of course, if $\boldsymbol{\theta}$ is random, then $y(\boldsymbol{\theta})$ becomes random. Many problems in grid operations rely on approximating the PDF or estimating statistics (mean or covariance) of $y(\boldsymbol{\theta})$ given the PDF or samples of $\boldsymbol{\theta}$; see [6] and references therein. This work does not consider the aforesaid problem. Here $\boldsymbol{\theta}$ and $y(\boldsymbol{\theta})$ are both deterministic. What is modeled as a GP is our estimate $\hat{y}(\boldsymbol{\theta})$ of $y(\boldsymbol{\theta})$ given dataset $\mathcal{T}$. It is our model estimates for given and future data that are modeled as jointly Gaussian in (6). This agrees with the principle of Bayesian inference wherein an unknown quantity y is assigned a prior PDF even if y is deterministic. Upon collecting data related to y, one computes the posterior PDF of y. The final estimate $\hat{y}$ for y is exactly the mean of the posterior distribution. For example, GP modeling has been widely used in geostatistics where scientists would like to build a topographical map of an area using elevation readings collected at a finite number of locations. To be able to interpolate from the collected elevation samples to unobserved points, a GP model is postulated even though elevation is a deterministic process, not a random one. If $y(\boldsymbol{\theta})$ is relatively smooth, a kernel function such as the Gaussian one in (8) can capture the variation of y over $\boldsymbol{\theta}$'s. The analogy carries over to the OPF problem at hand.*

Remark 2. *We decided to build a separate GP model for each entry of the OPF minimizer $\mathbf{x}_{\boldsymbol{\theta}}$, or at least for the entries of $\mathbf{x}_{\boldsymbol{\theta}}$ we are interested in. Alternatively, we could pursue training GP models jointly over all inverters. In this case, one should come up with meaningful kernels over $\boldsymbol{\theta}$'s and buses alike, i.e., $k((\boldsymbol{\theta}_i, n), (\boldsymbol{\theta}_j, m))$, to jointly train GP models for $x_n(\boldsymbol{\theta})$ and $x_m(\boldsymbol{\theta})$. To simplify the joint kernel design task, a product structure is oftentimes imposed according to which $k((\boldsymbol{\theta}_i, n), (\boldsymbol{\theta}_j, m)) = k_1(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) \cdot k_2(n, m)$; see e.g., [23], [35]. Whether such structure makes sense for GP-OPF and how to parameterize $k_2(n, m)$ in a physics-informed manner are non-trivial but relevant questions, which go beyond the scope of this work.*

IV. SENSITIVITY-INFORMED GP-OPF (SI-GP-OPF)

So far, the training dataset $\mathcal{T}$ consists of pairs of OPF parameters and solutions. This complies with the standard supervised learning setup. Nonetheless, when one aims at predicting solutions to a parametric optimization, there is more information to be exploited. Incorporating such rich information of OPF solutions can improve data efficiency in the sense that: *i)* A learner could infer $y(\boldsymbol{\theta})$ with the same estimation accuracy using a smaller training dataset $\mathcal{T}$. This is computationally advantageous as fewer instances of (3) have to be solved; or *ii)* A learner could achieve higher estimation accuracy for the same $\mathcal{T}$. We next elaborate on one of the additional information on $y(\boldsymbol{\theta})$ the learner can exploit.

When it comes to a parametric problem [cf. (5)], one can compute the partial derivatives of the minimizer $\mathbf{x}_{\boldsymbol{\theta}}$ with respect to parameters $\boldsymbol{\theta}$ using sensitivity analysis; see [36], [30]. Given the primal/dual solution of an optimization, computing the Jacobian matrix $\nabla_{\boldsymbol{\theta}}\mathbf{x}_{\boldsymbol{\theta}}$ carrying the sensitivities of $\mathbf{x}_{\boldsymbol{\theta}}$ with respect to $\boldsymbol{\theta}$ is as easy as solving a system of linear equations as long as the problem involves continuously differentiable objective and constraint functions. We defer the sensitivity analysis of (5) to Sec. VI. For now, let us suppose that along with the solution $y(\boldsymbol{\theta}_t)$ for each t, the learner has also computed the M-length gradient $\nabla_{\boldsymbol{\theta}}y(\boldsymbol{\theta}_t)$, denoted by $\dot{\mathbf{y}}(\boldsymbol{\theta}_t)$ for short. We thus augment the training dataset as

$$\mathcal{T} = \{(\boldsymbol{\theta}_t, y(\boldsymbol{\theta}_t))\}_{t=1}^T \rightarrow \bar{\mathcal{T}} = \{(\boldsymbol{\theta}_t, y(\boldsymbol{\theta}_t), \dot{\mathbf{y}}(\boldsymbol{\theta}_t))\}_{t=1}^T. \tag{10}$$

The new dataset $\bar{\mathcal{T}}$ carries $(M+1)$ pieces of information per OPF example rather than just one as in the original $\mathcal{T}$. In other words, for each sampled $\boldsymbol{\theta}_t$, we now know not only the value of the OPF mapping $y(\boldsymbol{\theta})$, but also its gradient $\dot{y}(\boldsymbol{\theta})$. The gradient information $\dot{y}(\boldsymbol{\theta})$ is important as along with $y(\boldsymbol{\theta})$, it approximates the mapping $y(\boldsymbol{\theta})$ in a neighborhood around $\boldsymbol{\theta}_t$ through a first-order Taylor's series expansion. Moreover, knowing the gradients, we can train a machine learning that given a new $\boldsymbol{\theta}$, predicts both $y(\boldsymbol{\theta})$ and its gradient.

The pertinent question now is whether the extra information of gradients can be incorporated into GP-OPF. Can the additional data $\{\dot{\mathbf{y}}(\boldsymbol{\theta}_t)\}_{t=1}^T$ be included in the GP model as part of vector $\mathbf{y}_1$ in (6)? The requirement for achieving this is that gradient data can also be modeled as GPs, and that their covariances (appearing as blocks of $\boldsymbol{\Sigma}_{11}$ and $\boldsymbol{\Sigma}_{12}$ in (6)) can be described by a parametric model. Fortunately, an appealing property of GPs is that the derivative of a GP with respect to its independent variable ($\boldsymbol{\theta}$ in our case) is a GP itself. In particular, if $y(\boldsymbol{\theta})$ is a zero-mean GP, the gradient $\dot{\mathbf{y}}(\boldsymbol{\theta})$ is a zero-mean GP as well. Additionally, the covariance between $y(\boldsymbol{\theta})$ and $\dot{\mathbf{y}}(\boldsymbol{\theta})$ can be derived as

$$
\begin{aligned}
\mathbb{E}[y(\boldsymbol{\theta}_i)\dot{\mathbf{y}}(\boldsymbol{\theta}_j)] &= \mathbb{E}[y(\boldsymbol{\theta}_i)\nabla_{\boldsymbol{\theta}_j} y(\boldsymbol{\theta}_j)] \\
&= \nabla_{\boldsymbol{\theta}_j}\mathbb{E}[y(\boldsymbol{\theta}_i)y(\boldsymbol{\theta}_j)] \\
&= \nabla_{\boldsymbol{\theta}_j} k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)
\end{aligned}
\tag{11}
$$

where the second equality follows by exchanging the order of expectation and differentiation, and the third equality is by definition of the kernel function. The covariance between gradient vectors can be derived similarly as

$$
\mathbb{E}[\dot{\mathbf{y}}(\boldsymbol{\theta}_i)\dot{\mathbf{y}}^\top(\boldsymbol{\theta}_j)] = \nabla^2_{\boldsymbol{\theta}_i\boldsymbol{\theta}_j}\mathbb{E}[y(\boldsymbol{\theta}_i)y(\boldsymbol{\theta}_j)] = \nabla^2_{\boldsymbol{\theta}_i\boldsymbol{\theta}_j} k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j).
$$

For the Gaussian kernel in (8), the aforesaid quantities can be readily computed as

$$
\nabla_{\boldsymbol{\theta}_j} k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = \beta k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j)
$$
$$
\nabla^2_{\boldsymbol{\theta}_i\boldsymbol{\theta}_j} k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = \beta k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)[\mathbf{I}_M - \beta(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j)(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j)^\top]
$$

where $\mathbf{I}_M$ is the identity matrix of size M. As in (9), gradient labels are observed under modeling noise so for some $\epsilon > 0$:

$$
\mathbb{E}[\dot{\mathbf{y}}(\boldsymbol{\theta}_i)\dot{\mathbf{y}}(\boldsymbol{\theta}_j)^\top] = \nabla^2_{\boldsymbol{\theta}_i\boldsymbol{\theta}_j} k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) + \epsilon\delta_{ij}\mathbf{I}_M.
$$

Since gradients comply with the GP model, they can be included in the GP framework presented earlier. Stacking this extra information modifies $\mathbf{y}_1$ of (6) from a T-length vector carrying only $y(\boldsymbol{\theta}_t)$'s to a vector of dimension $\bar{T} := T(M+1)$:

$$
\bar{\mathbf{y}}_1^\top := \begin{bmatrix} \mathbf{y}_1^\top & \dot{\mathbf{y}}^\top(\boldsymbol{\theta}_1) & \cdots & \dot{\mathbf{y}}^\top(\boldsymbol{\theta}_T) \end{bmatrix}.
$$

Vector $\bar{\mathbf{y}}_1$ replaces $\mathbf{y}_1$ in (6) and (7). Mean vectors remain zero and the covariances can be computed as explained earlier. It is worth noting that incorporating sensitivities into GP models is quite standard [33, Sec. 9.4]. The novelty here is that the sensitivities are indeed available for optimization data and can be obtained with minimal computational overhead. Deferring the computation of $\dot{\mathbf{y}}(\boldsymbol{\theta})$ to Section VI, we next address the computational issues arising when expanding data by $(M+1)$ times while migrating from $\mathbf{y}_1$ to $\bar{\mathbf{y}}_1$.

V. RANDOM FEATURE-BASED SI-GP-OPF

GP-OPF suffers from the curse of dimensionality [33]. If the size of the training dataset is T, inverting matrix $\boldsymbol{\Sigma}_{11}$ in (7) takes $\mathcal{O}(T^3)$ operations during training. With $\boldsymbol{\Sigma}_{11}^{-1}$ and $\boldsymbol{\Sigma}_{11}^{-1}\mathbf{y}_1$ computed, prediction takes $\mathcal{O}(T)$ for the mean in (7a), and $\mathcal{O}(T^2)$ for the variance in (7b) per new test case. To render GP learning scalable, existing solutions include low-rank [37], structured approximants of $\boldsymbol{\Sigma}_{11}$ [38], and random features [39]. The scaling issue of GPs is exacerbated with

SI-GPs. Albeit gradient labels are introduced to reduce the number of OPF instances T that need to be solved, they increase the size of the augmented dataset as $\bar{T} = (M+1)T$. This section extends the concept of *random features (RFs)* to gradient data to enable scalable learning of the OPF mapping.

We first present the plain RF-based GP-OPF (RF-GP-OPF). The crux in GP-OPF is inverting $\boldsymbol{\Sigma}_{11}$. The idea of random features is to approximate the kernel function of (8) as the inner product between two D-length vectors [40]

$$
k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) \simeq \alpha\mathbf{z}^\top(\boldsymbol{\theta}_i)\mathbf{z}(\boldsymbol{\theta}_j)
\tag{12}
$$

where vector $\mathbf{z}(\boldsymbol{\theta})$ is a randomized nonlinear (sinusoidal) transformation of $\boldsymbol{\theta}$. Its d-th entry is defined as

$$
z_d(\boldsymbol{\theta}) := \sqrt{\tfrac{2}{D}}\cos\left(\mathbf{v}_d^\top\boldsymbol{\theta} + \phi_d\right), \quad d = 1, \ldots, D.
\tag{13}
$$

Here $\{\mathbf{v}_d\}_{d=1}^D$ are random vectors drawn independently from $\mathcal{N}(\mathbf{0}, \beta\mathbf{I}_M)$, and $\{\phi_d\}_{d=1}^D$ are random scalars drawn uniformly from $[0, 2\pi]$. Recall β is one of the parameters of the Gaussian kernel in (8) and M is the length of $\boldsymbol{\theta}$. Vector $\mathbf{z}(\boldsymbol{\theta}_i)$ constitutes the vector of *random features* that transforms datum $\boldsymbol{\theta}_i \in \mathbb{R}^M$ to $\mathbf{z}(\boldsymbol{\theta}_i) \in \mathbb{R}^D$. For an explanation of why (12) is a valid approximation and how D affects accuracy, the interested reader is referred to Appendix A. Reference [40] shows that (12) approximates $k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ within ϵ uniformly over $(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ if D is selected as $\mathcal{O}(M\epsilon^{-2}\log\epsilon^{-2})$, though excellent regression results are empirically observed with smaller D.

Let us collect the RF vectors for all training data $\{\mathbf{z}(\boldsymbol{\theta}_t)\}_{t=1}^T$ as rows of a $T\times D$ matrix $\mathbf{Z}_1$. Due to (12), we can approximate the covariance $\boldsymbol{\Sigma}_{11}$ from (8)–(9) as

$$
\hat{\boldsymbol{\Sigma}}_{11} = \alpha\mathbf{Z}_1\mathbf{Z}_1^\top + \gamma\mathbf{I}_T.
\tag{14}
$$

Similarly to $\mathbf{Z}_1$, let the $S \times D$ matrix $\mathbf{Z}_2$ collect the RF vectors for all data of the validation dataset $\mathcal{S}$. Then, the cross-covariance $\boldsymbol{\Sigma}_{21}$ can be approximated as

$$
\hat{\boldsymbol{\Sigma}}_{21} = \alpha\mathbf{Z}_2\mathbf{Z}_1^\top.
\tag{15}
$$

From (14)–(15), we can approximate $\boldsymbol{\Sigma}_{21}\boldsymbol{\Sigma}_{11}^{-1}$ as

$$
\hat{\boldsymbol{\Sigma}}_{21}\hat{\boldsymbol{\Sigma}}_{11}^{-1} = \alpha\mathbf{Z}_2\mathbf{Z}_1^\top\left(\alpha\mathbf{Z}_1\mathbf{Z}_1^\top + \gamma\mathbf{I}_T\right)^{-1}
\tag{16a}
$$
$$
= \alpha\mathbf{Z}_2\left(\alpha\mathbf{Z}_1^\top\mathbf{Z}_1 + \gamma\mathbf{I}_D\right)^{-1}\mathbf{Z}_1^\top
\tag{16b}
$$

where the second equality follows from the matrix inversion lemma. The key point is that (16b) involves inverting a $D \times D$ rather than a $T \times T$ matrix as in (16a). Leveraging (16), Table I reports the steps of RF-GP-OPF and their complexity. If $T > D > M$, the training phase takes $\mathcal{O}(D^2 T)$, while the prediction phase costs $\mathcal{O}(DM)$ for the mean and $\mathcal{O}(D^2)$ for the variance per new datum. If $T < M$, RF-GP-OPF has no advantage over the plain GP-OPF.

An RF-based implementation is really helpful when migrating from GP-OPF to SI-GP-OPF as now the size of the training dataset explodes from T to $\bar{T} = T(M+1)$. If one implements SI-GP-OPF using RFs (hereafter termed RF-SI-GP-OPF) shortsightedly as per Table I, the complexity over training would be $\mathcal{O}(D^2 TM)$. A smarter implementation exploiting the problem structure can reduce the complexity further to $\mathcal{O}(DTM + D^3 + D^2(T + M))$ as explained next.

TABLE I
TRAINING/PREDICTION WITH RF-GP-OPF

Training Phase	$\mathcal{O}(D^2T)$
T1) Draw $\{\mathbf{v}_d, \phi_d\}_{d=1}^D$ and find $\mathbf{Z}_1$ from (13)	$\mathcal{O}(DTM)$
T2) Compute $\mathbf{Z}_1^\top \mathbf{Z}_1$	$\mathcal{O}(D^2T)$
T3) Compute $\mathbf{Z}_1^\top \mathbf{y}_1$	$\mathcal{O}(DT)$
T4) Invert $\alpha\mathbf{Z}_1^\top \mathbf{Z}_1 + \gamma\mathbf{I}_D$	$\mathcal{O}(D^3)$
T5) Compute $(\alpha\mathbf{Z}_1^\top \mathbf{Z}_1 + \gamma\mathbf{I}_D)^{-1}(\mathbf{Z}_1^\top \mathbf{y}_1)$	$\mathcal{O}(D^2)$
T6) Compute $(\alpha\mathbf{Z}_1^\top \mathbf{Z}_1 + \gamma\mathbf{I}_D)^{-1}(\mathbf{Z}_1^\top \mathbf{Z}_1)$	$\mathcal{O}(D^3)$
Prediction Phase (per new OPF instance)	
P1) Compute $\mathbf{Z}_2 \in \mathbb{R}^{1 \times D}$ from (13)	$\mathcal{O}(DM)$
P2) Premultiply the result of *T5)* by $\alpha\mathbf{Z}_2$ to find predictive mean (7a)	$\mathcal{O}(D)$
P3) Pre/post-multiply the result of *T6)* by $\alpha\mathbf{Z}_2$ $(\alpha\mathbf{Z}_2^\top)$ to find predictive variance (7b)	$\mathcal{O}(D^2)$

TABLE II
TRAINING/PREDICTION WITH RF-SI-GP-OPF

Training $\mathcal{O}(DTM + D^2(D+T+M))$	
T1) Draw $\{\mathbf{v}_d, \phi_d\}_{d=1}^D$ and find $\bar{\mathbf{Z}}_1$	$\mathcal{O}(DTM)$
T2) Find $\bar{\mathbf{Z}}_1^\top \mathbf{D}^{-1} \bar{\mathbf{Z}}_1$	$\mathcal{O}(D^2(T+M))$
T3) Find $\bar{\mathbf{Z}}_1^\top \mathbf{D}^{-1} \bar{\mathbf{y}}_1$	$\mathcal{O}(DTM)$
T4) Invert $\bar{\mathbf{Z}}_1^\top \mathbf{D}^{-1} \bar{\mathbf{Z}}_1 + \mathbf{I}_D$	$\mathcal{O}(D^3)$
T5) Find $(\bar{\mathbf{Z}}_1^\top \mathbf{D}^{-1} \bar{\mathbf{Z}}_1 + \mathbf{I}_D)^{-1}(\bar{\mathbf{Z}}_1^\top \mathbf{D}^{-1} \bar{\mathbf{y}}_1)$	$\mathcal{O}(D^2)$
T6) Find $(\bar{\mathbf{Z}}_1^\top \mathbf{D}^{-1} \bar{\mathbf{Z}}_1 + \mathbf{I}_D)^{-1}(\bar{\mathbf{Z}}_1^\top \mathbf{D}^{-1} \bar{\mathbf{Z}}_1)$	$\mathcal{O}(D^3)$
Prediction (as in Table I)	

We are interested in finding unbiased estimates of $\mathbb{E}[y(\boldsymbol{\theta}_i)\dot{\mathbf{y}}(\boldsymbol{\theta}_j)]$ and $\mathbb{E}[\dot{\mathbf{y}}(\boldsymbol{\theta}_i)\dot{\mathbf{y}}^\top(\boldsymbol{\theta}_j)]$. We rely on (11). If (12) is an estimate of $k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$, then $\nabla_{\boldsymbol{\theta}_j} k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ and $\nabla^2_{\boldsymbol{\theta}_i \boldsymbol{\theta}_j} k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ can be approximated by inner products as

$$\nabla_{\boldsymbol{\theta}_j} k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) \simeq \alpha \left(\nabla_{\boldsymbol{\theta}_j} \mathbf{z}(\boldsymbol{\theta}_j) \right)^\top \mathbf{z}(\boldsymbol{\theta}_i) \tag{17a}$$

$$\nabla^2_{\boldsymbol{\theta}_i \boldsymbol{\theta}_j} k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) \simeq \alpha \left(\nabla_{\boldsymbol{\theta}_i} \mathbf{z}(\boldsymbol{\theta}_i) \right)^\top \nabla_{\boldsymbol{\theta}_j} \mathbf{z}(\boldsymbol{\theta}_j) \tag{17b}$$

where $\nabla_{\boldsymbol{\theta}_j} \mathbf{z}(\boldsymbol{\theta}_j)$ is a $D \times M$ Jacobian matrix. The d-th row of this Jacobian can be computed by differentiating (13) to get

$$\nabla_{\boldsymbol{\theta}_j} z_d(\boldsymbol{\theta}_j) = -\sqrt{\tfrac{2}{D}} \sin(\mathbf{v}_d^\top \boldsymbol{\theta}_j + \phi_d)\mathbf{v}_d.$$

For the computational advantage shown in (16) to carry over to SI-GPs, we should be able to express the covariance $\mathbb{E}[\bar{\mathbf{y}}_1 \bar{\mathbf{y}}_1^\top]$ as a rank–D plus a scaled identity matrix. To this end, define the quantities

$$s_d(\boldsymbol{\theta}) := -\sqrt{\tfrac{2}{D}} \sin(\mathbf{v}_d^\top \boldsymbol{\theta} + \phi_d), \quad d = 1, \ldots, D$$

and stack them in vector $\mathbf{s}(\boldsymbol{\theta}_j) := [s_1(\boldsymbol{\theta}_j) \ \cdots \ s_D(\boldsymbol{\theta}_j)]^\top$. It is not hard to verify that the Jacobian matrix $\nabla_{\boldsymbol{\theta}_j} \mathbf{z}(\boldsymbol{\theta}_j)$ can be expressed as the Khatri-Rao product

$$\nabla_{\boldsymbol{\theta}_j} \mathbf{z}(\boldsymbol{\theta}_j) = \left(\mathbf{s}^\top(\boldsymbol{\theta}_j) * \mathbf{V} \right)^\top$$

where $\mathbf{V} := [\mathbf{v}_1 \ \cdots \ \mathbf{v}_D]$ is a $M \times D$ matrix. If we place vectors $\{\mathbf{s}(\boldsymbol{\theta}_t)\}_{t=1}^T$ as rows of matrix $\mathbf{S}_1$, we can approximate

$$\mathbb{E}[\bar{\mathbf{y}}_1 \bar{\mathbf{y}}_1^\top] \simeq \alpha \bar{\mathbf{Z}}_1 \bar{\mathbf{Z}}_1^\top + \mathbf{D} \tag{18}$$

where $\quad \bar{\mathbf{Z}}_1 := \begin{bmatrix} \mathbf{Z}_1 \\ \mathbf{S}_1 * \mathbf{V} \end{bmatrix} \quad$ and $\quad \mathbf{D} := \begin{bmatrix} \gamma\mathbf{I}_T & \mathbf{0} \\ \mathbf{0} & \epsilon\mathbf{I}_{MT} \end{bmatrix}$.

Since $\mathbf{Z}_1$ and $\mathbf{S}_1$ are $T \times D$ and $\mathbf{V}$ is $M \times D$, matrix $\bar{\mathbf{Z}}_1$ is $T(M+1) \times D$. Thanks to (18), the computational advantage of (16) carries over to SI-GPs as now $\boldsymbol{\Sigma}_{11} = \mathbb{E}[\bar{\mathbf{y}}_1 \bar{\mathbf{y}}_1^\top]$ and

$$\begin{aligned} \hat{\boldsymbol{\Sigma}}_{21}\hat{\boldsymbol{\Sigma}}_{11}^{-1} &= \alpha\mathbf{Z}_2 \bar{\mathbf{Z}}_1^\top \left(\alpha\bar{\mathbf{Z}}_1\bar{\mathbf{Z}}_1^\top + \mathbf{D} \right)^{-1} \\ &= \alpha\mathbf{Z}_2 \left(\alpha\bar{\mathbf{Z}}_1^\top \mathbf{D}^{-1} \bar{\mathbf{Z}}_1 + \mathbf{I}_D \right)^{-1} \bar{\mathbf{Z}}_1^\top \mathbf{D}^{-1}. \end{aligned}$$

Again, we need to invert a $D \times D$ instead of a $\bar{T} \times \bar{T}$ matrix. Table II details the computational complexity per step of RF-SI-GP-OPF. Interestingly, steps *T1)–T2)* of Table II maintain the complexity of steps *T1)–T2)* of Table I despite T has been replaced by $\bar{T}$: Regarding *T1)*, a careful yet mundane analysis on (18) shows that $\bar{\mathbf{Z}}_1$ can indeed be computed in $\mathcal{O}(DTM)$, and not $\mathcal{O}(D\bar{T}M)$. Step *T2)* takes $\mathcal{O}(D^2T)$ as the properties of the Khatri-Rao product yield

$$\begin{aligned} \bar{\mathbf{Z}}_1^\top \mathbf{D}^{-1} \bar{\mathbf{Z}}_1 &= \gamma^{-1}\mathbf{Z}_1^\top \mathbf{Z}_1 + \epsilon^{-1}(\mathbf{S}_1 * \mathbf{V})^\top (\mathbf{S}_1 * \mathbf{V}) \\ &= \gamma^{-1}\mathbf{Z}_1^\top \mathbf{Z}_1 + \epsilon^{-1}(\mathbf{S}_1^\top \mathbf{S}_1) \circ (\mathbf{V}^\top \mathbf{V}) \end{aligned}$$

where $\circ$ denotes the Hadamard (entry-wise) matrix multiplication. Evidently, the complexity of RF-SI-GP-OPF is of the same order as that of RF-GP-OPF.

In summary, the suggested RF-SI-GP-OPF is implemented as described next. Data generation step *s1)* is shared across all inverters. This involves solving T OPF instances and computing sensitivities. Learning step *s2)* is implemented separately per inverter. It includes finding kernel hyperparameters via MLE and following the steps in Table II. Both *s1)* and *s2)* occur offline. In real time, setpoints are computed per inverter according to the prediction phase of Table VII-B. This phase entails only $\mathcal{O}(DM)$ or $\mathcal{O}(D^2)$ calculations depending on whether predictive variance is needed or not. Because $D > T > M$ in our tests, the computational time during real-time operation is independent of M and scales linearly with the number of inverters. If we assume that M and D scale linearly with N, and that $N_g = N$, the prediction phase has a worst-case complexity of $\mathcal{O}(N^3)$. This still outperforms interior point-based OPF solvers (LP, QP, SOCP) whose complexities are $\mathcal{O}(N^\kappa)$ with $\kappa > 3.5$.

VI. SENSITIVITY ANALYSIS OF SOCP-BASED OPF

Section III expanded dataset $\mathcal{T}$ to $\bar{\mathcal{T}}$ by including the gradient vector $\nabla_{\boldsymbol{\theta}} y(\boldsymbol{\theta})$ and recall $y(\boldsymbol{\theta})$ denotes one of the entries of the minimizer $\mathbf{x}(\boldsymbol{\theta})$ of (5). This section performs sensitivity analysis of (5) to compute the Jacobian matrix of the entire vector $\mathbf{x}(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$. Once presented with a $\boldsymbol{\theta}$, an off-the-shelf SOCP solver can be used to obtain the optimal primal/dual variables of (5). Dropping the subscript $\boldsymbol{\theta}$, let us denote the minimizer as $\mathbf{x}$, and the optimal dual variables for (5b)–(5d) as $\boldsymbol{\lambda}$, $\boldsymbol{\mu}$, and $\{\bar{\boldsymbol{\nu}}_m\}_{m=1}^M$, respectively. The lengths of $\boldsymbol{\lambda}$ and $\boldsymbol{\mu}$ coincide with the number of equality and inequality constraints, while $\bar{\boldsymbol{\nu}}_m$'s are the dual conic variables with lengths one more than the number of rows in $\mathbf{A}_m$. The dual variables obtained from most SOCP solvers correspond to the conic approach-based Lagrangian function

$$\mathbf{c}^\top \mathbf{x} + \boldsymbol{\lambda}^\top (\mathbf{A}_e \mathbf{x} - \mathbf{B}_e \boldsymbol{\theta} - \mathbf{f}_e) + \boldsymbol{\mu}^\top (\mathbf{A}_i \mathbf{x} - \mathbf{B}_i \boldsymbol{\theta} - \mathbf{f}_i)$$

$$+ \sum_{m=1}^{2N} \bar{\boldsymbol{\nu}}_m^\top \begin{bmatrix} -(\mathbf{b}_m^\top \mathbf{x} + f_m) \\ \mathbf{A}_m \mathbf{x} \end{bmatrix}.$$

Deviating from the conic-approach, for sensitivity computation we invoke the direct approach-based Lagrangian where the SOCs in (5d) are treated as scalar constraints

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu}; \boldsymbol{\theta}) = \mathbf{c}^\top \mathbf{x} + \boldsymbol{\lambda}^\top (\mathbf{A}_e \mathbf{x} - \mathbf{B}_e \boldsymbol{\theta} - \mathbf{f}_e)$$
$$+ \boldsymbol{\mu}^\top (\mathbf{A}_i \mathbf{x} - \mathbf{B}_i \boldsymbol{\theta} - \mathbf{f}_i)$$
$$+ \sum_{m=1}^{2N} \nu_m \left(\|\mathbf{A}_m \mathbf{x}\| - \mathbf{b}_m^\top \mathbf{x} - f_m \right).$$

It can be shown that the optimal dual variables obtained from the conic approach satisfy the Karush–Kuhn–Tucker (KKT) conditions for the direct approach with ν_m being equal to the first entry of $\bar{\boldsymbol{\nu}}_m$ [41]. This allows us to obtain the optimal dual variables from a conic solver and proceed with sensitivity analysis using the KKT conditions for the direct approach.

Sensitivity analysis aims at finding infinitesimal changes $(\mathrm{d}\mathbf{x}, \mathrm{d}\boldsymbol{\lambda}, \mathrm{d}\boldsymbol{\mu}, \mathrm{d}\boldsymbol{\nu})$, so that the perturbed point $(\mathbf{x} + \mathrm{d}\mathbf{x}, \boldsymbol{\lambda} + \mathrm{d}\boldsymbol{\lambda}, \boldsymbol{\mu} + \mathrm{d}\boldsymbol{\mu}, \boldsymbol{\nu} + \mathrm{d}\boldsymbol{\nu})$ satisfies the first-order KKT conditions when the problem parameters change from $\boldsymbol{\theta}$ to $\boldsymbol{\theta} + \mathrm{d}\boldsymbol{\theta}$ [36]. To this end, we first review optimality conditions and then apply implicit differentiation to compute the sought sensitivities. Starting with Lagrangian optimality condition $\nabla_{\mathbf{x}} \mathcal{L} = \mathbf{0}$:

$$\mathbf{c} + \mathbf{A}_e^\top \boldsymbol{\lambda} + \mathbf{A}_i^\top \boldsymbol{\mu} + \sum_{m=1}^{2N} \nu_m \left(\frac{\mathbf{A}_m^\top \mathbf{A}_m \mathbf{x}}{\|\mathbf{A}_m \mathbf{x}\|} - \mathbf{b}_m \right) = \mathbf{0} \quad (20)$$

The first-order optimality conditions further include primal feasibility (5b)–(5d); dual feasibility $\boldsymbol{\mu} \geq \mathbf{0}$ and $\boldsymbol{\nu} \geq \mathbf{0}$; and complementary slackness

$$\mathrm{dg}(\boldsymbol{\mu}) (\mathbf{A}_i \mathbf{x} - \mathbf{B}_i \boldsymbol{\theta} - \mathbf{f}_i) = \mathbf{0} \qquad (21a)$$
$$\nu_m \left(\|\mathbf{A}_m \mathbf{x}\| - \mathbf{b}_m^\top \mathbf{x} - f_m \right) = 0, \quad \forall m. \qquad (21b)$$

We next compute the total differentials over $(\mathrm{d}\mathbf{x}, \mathrm{d}\boldsymbol{\lambda}, \mathrm{d}\boldsymbol{\mu}, \mathrm{d}\boldsymbol{\nu}, \mathrm{d}\boldsymbol{\theta})$ for the first-order optimality conditions involving equalities, namely conditions (5b), (20), and (21):

$$\mathbf{A}_e \, \mathrm{d}\mathbf{x} - \mathbf{B}_e \, \mathrm{d}\boldsymbol{\theta} = \mathbf{0} \qquad (22a)$$

$$\mathbf{A}_e^\top \, \mathrm{d}\boldsymbol{\lambda} + \mathbf{A}_i^\top \, \mathrm{d}\boldsymbol{\mu} + \sum_{m=1}^{2N} \left(\frac{\mathbf{A}_m^\top \mathbf{A}_m \mathbf{x}}{\|\mathbf{A}_m \mathbf{x}\|} - \mathbf{b}_m \right) \mathrm{d}\nu_m$$
$$+ \sum_{m=1}^{2N} \nu_m \left(\frac{\mathbf{A}_m^\top \mathbf{A}_m}{\|\mathbf{A}_m \mathbf{x}\|} - \frac{\mathbf{A}_m^\top \mathbf{A}_m \mathbf{x} \mathbf{x}^\top \mathbf{A}_m^\top \mathbf{A}_m}{\|\mathbf{A}_m \mathbf{x}\|^3} \right) \mathrm{d}\mathbf{x} = \mathbf{0}$$
$$(22b)$$

$$\mathrm{dg}(\boldsymbol{\mu}) (\mathbf{A}_i \, \mathrm{d}\mathbf{x} - \mathbf{B}_i \, \mathrm{d}\boldsymbol{\theta}) + \mathrm{dg} (\mathbf{A}_i \mathbf{x} - \mathbf{B}_i \boldsymbol{\theta} - \mathbf{f}_i) \, \mathrm{d}\boldsymbol{\mu} = \mathbf{0}$$
$$(22c)$$

$$\nu_m \left(\frac{\mathbf{x}^\top \mathbf{A}_m^\top \mathbf{A}_m}{\|\mathbf{A}_m \mathbf{x}\|} - \mathbf{b}_m^\top \right) \mathrm{d}\mathbf{x}$$
$$+ \left(\|\mathbf{A}_m \mathbf{x}\| - \mathbf{b}_m^\top \mathbf{x} - f_m \right) \mathrm{d}\nu_m = 0, \ \forall m. \quad (22d)$$

Given an optimal primal/dual solution $(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu})$ of (5), a perturbed point $(\mathbf{x} + \mathrm{d}\mathbf{x}, \boldsymbol{\lambda} + \mathrm{d}\boldsymbol{\lambda}, \boldsymbol{\mu} + \mathrm{d}\boldsymbol{\mu}, \boldsymbol{\nu} + \mathrm{d}\boldsymbol{\nu})$ satisfies the KKT equality conditions (5b), (20), and (21) for $\boldsymbol{\theta} + \mathrm{d}\boldsymbol{\theta}$, if the conditions in (22) are satisfied. Interestingly, it can be shown that under *strict complementary slackness*, satisfying (22) ensures that the perturbed point satisfies the inequality

conditions (5c)–(5d), and the dual feasibility conditions as well [18]. In other words, although the perturbed point was constructed by taking into account only the optimality conditions expressed as equalities, it also satisfies the optimality conditions expressed as inequalities. Hence, the perturbed point constitutes an optimal primal/dual solution of (5) for parameter $\boldsymbol{\theta} + \mathrm{d}\boldsymbol{\theta}$.[1] Defining $\boldsymbol{\delta} := [\mathbf{x}^\top \ \boldsymbol{\lambda}^\top \ \boldsymbol{\mu}^\top \ \boldsymbol{\nu}^\top]^\top$, the system of equations in (22) can be compactly represented as

$$\mathbf{S} \, \mathrm{d}\boldsymbol{\delta} = \mathbf{U} \, \mathrm{d}\boldsymbol{\theta} \qquad (23)$$

where $(\mathbf{S}, \mathbf{U})$ follow from (22) and depend on $(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu})$ as detailed in Appendix B.

If $\mathbf{S}$ is invertible, the desired sensitivities $\nabla_{\boldsymbol{\theta}} \mathbf{x}$ can be extracted as the top rows of $\mathbf{S}^{-1} \mathbf{U}$ corresponding to $\mathbf{x}$. However, OPF instances leading to singular $\mathbf{S}$ have been reported for transmission and distribution systems [17], [42]. Investigating further into such scenarios, reference [18] argues that the invertibility of $\mathbf{S}$ is not necessarily needed to compute $\nabla_{\boldsymbol{\theta}} \mathbf{x}$ using (23). Rather, given a vector $\mathrm{d}\boldsymbol{\theta}$, if (23) has a unique solution for $\mathrm{d}\mathbf{x}$, the sensitivities $\nabla_{\boldsymbol{\theta}} \mathbf{x}$ do exist. Uniqueness in $\mathrm{d}\mathbf{x}$ is guaranteed if for any vector in $\mathrm{null}(\mathbf{S})$, its top entries corresponding to $\mathbf{x}$ are zero. Therefore, if the basis of $\mathrm{null}(\mathbf{S})$ exhibits the desired sparsity, then $\nabla_{\boldsymbol{\theta}} \mathbf{x}$ exists. If it exists, matrix $\nabla_{\boldsymbol{\theta}} \mathbf{x}$ can be extracted as the top rows of $\mathbf{S}^\dagger \mathbf{U}$ corresponding to $\mathbf{x}$, where $\mathbf{S}^\dagger$ is the pseudoinverse of $\mathbf{S}$. Ultimately, existence and computation of $\nabla_{\boldsymbol{\theta}} \mathbf{x}$ are ensured by two technical assumptions: strict complementary slackness and a second-order optimality condition; see [18] for details. In this work, the steps followed to augment the training set $\mathcal{T}$ with sensitivities $\nabla_{\boldsymbol{\theta}} \mathbf{x}$ are: *i)* Use the optimal solution $(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}, \boldsymbol{\nu})$ obtained from an SOC solver to construct the system of linear equations in (23); *ii)* Numerically verify the existence of $\nabla_{\boldsymbol{\theta}} \mathbf{x}$ by checking the sparsity pattern for the basis of $\mathrm{null}(\mathbf{S})$; and *iii)* Extract $\nabla_{\boldsymbol{\theta}} \mathbf{x}$ as the appropriate rows of $\mathbf{S}^\dagger \mathbf{U}$.

Remark 3. *Our analysis presumed the cost and constraint functions of (5) are differentiable with respect to $\mathbf{x}$ and $\boldsymbol{\theta}$. This is true with the exception of the SOCs in (5d). The constraint function $\|\mathbf{A}_m \mathbf{x}\| - \mathbf{b}_m^\top \mathbf{x} - f_m$ becomes non-differentiable if $\mathbf{A}_m \mathbf{x} = \mathbf{0}$ at optimality. For the SOCs corresponding to the convex relaxation in (3d), this problematic scenario cannot occur because if (5) is feasible and the relaxation is exact, then $\|\mathbf{A}_m \mathbf{x}\| = v_{\pi_n} + \ell_n \geq \underline{v}_{\pi_n} > 0$.*

The apparent power constraints in (3h) call for a more careful treatment. Suppose the m-th apparent power constraint corresponds to inverter n. When posed as the SOC $\|\mathbf{A}_m \mathbf{x}\| \leq f_m$ with $f_m = \bar{s}_n^g > 0$, the constraint function becomes non-differentiable if $\|\mathbf{A}_m \mathbf{x}\| = 0$, or equivalently, $p_n^g = q_n^g = 0$ at optimality. Complementary slackness yields also $\nu_m = 0$. To deal with any non-differentiable constraint, we will perform sensitivity analysis assuming all troublesome constraints have been replaced by their original convex quadratic form in (3h). That form is differentiable and thus amenable to sensitivity
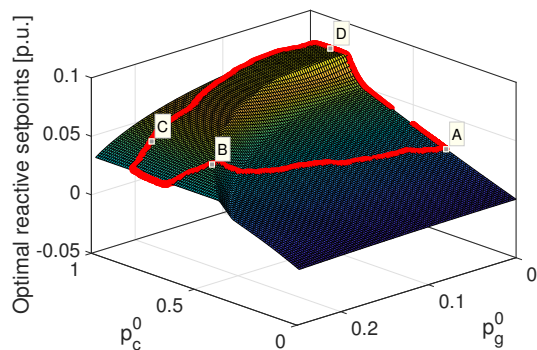

Fig. 1. Optimal q_5^g at bus 5 of the 13-bus benchmark as a function of the aggregate demand p_c^0 and solar generation p_g^0. Red circles mark the optimal reactive setpoints corresponding to real-world data.
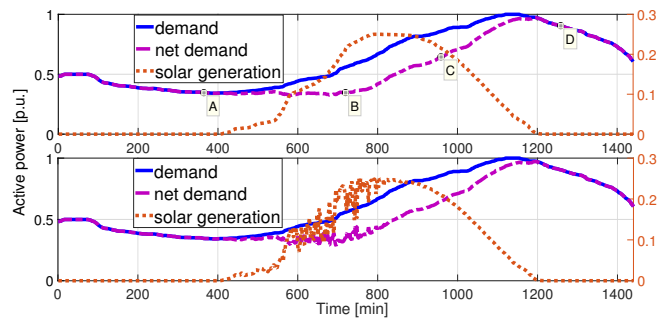

Fig. 2. Active demand (p_c^0), available active solar generation power (p_g^0), and net demand ($p_c^0 - p_g^0$) over a day under sunny (top panel) and cloudy (bottom panel) conditions. The right axis measures the magnitude of p_g^0.

analysis. Let ν_m' be the optimal dual for the differentiable form of the constraint. The primal/dual solutions of the two OPF formulations coincide and $\nu_m' = \nu_m = 0$. Standard results from sensitivity analysis show that if a constraint is non-binding under $\boldsymbol{\theta}$, it remains non-binding under any $\boldsymbol{\theta} + \mathrm{d}\boldsymbol{\theta}$ and so $\nu_m' = \mathrm{d}\nu_m' = 0$; see [30]. Because of this, the troublesome constraint can be ignored when forming (23). While the aforementioned discussion serves well for completeness, our numerical tests did not encounter instances of $\|\mathbf{A}_m \mathbf{x}\| = 0$.

VII. NUMERICAL TESTS

GP-OPF was tested on the IEEE 13- and 123-bus benchmarks converted to single-phase [8]. Real-world minute-based active load and solar generation data from 446 households were obtained from the Smart* Project collected on July 1 of 2011 and 2015, respectively [43]. Load demand per bus was simulated by adding up 20 randomly sampled household demands, and scaling this aggregate demand so its maximum matches the benchmark value for that bus. Sequences of solar generation were obtained upon aggregation and scaling similarly. Solar sequences were scaled on a per feeder and bus basis to simulate different solar penetration levels as described later. Lacking reactive power data, we simulated lagging power factors uniformly and independently drawn from $[0.9, 1.0]$ across buses, which were kept fixed across time. To allow for reactive power injection at maximum solar irradiance, we oversized inverters by 10%. All tests were ran on a 2.9 GHz AMD 7-core processor laptop computer with 16 GB RAM.

The labeled dataset was generated by solving the OPF in (3) using the YALMIP toolbox and the SOCP solver SDPT3. We used the GPML toolbox for estimating (α, β, γ). Having estimated these parameters, we subsequently estimated ϵ using MATLAB's `fitrgp` function. This toolbox accepts custom covariance functions, which is needed for introducing the covariance of sensitivities and finding ϵ. Covariance matrices were formed using the learned hyper-parameters per (11), and Tables I and II. The prediction accuracy of a GP-OPF estimate $\hat{x}_n$ of x_n was measured using the relative percent error (RPE) which is useful when x_n can take zero values $\text{RPE}_n := |\hat{x}_n - x_n| / \left(\frac{1}{N} \sum_{m=1}^{N} x_m \right)$.

A. Two-parameter OPF on IEEE 13-bus Feeder

For illustrative purposes and to draw intuition, we first evaluated GP-OPF on a simple OPF setup that depended on only two parameters (p_c^0, p_g^0) for the IEEE 13-bus system. We synthesized active loads and solar generation by scaling benchmark values by p_c^0 and p_g^0 respectively at all buses. Reactive loads were synthesized according to random power factors as described earlier. Thus, all OPF parameters were linear functions of (p_c^0, p_g^0). To obtain the OPF mapping, we uniformly sampled 100 values of p_c^0 and p_g^0 from $[0, 1]$ and $[0, 0.25]$, respectively; and then solved the related $10,000$ OPFs. Figure 1 illustrates the OPF mapping between (p_c^0, p_g^0) and the optimal reactive setpoint for bus 5. According to Fig. 1, the optimal q_5^g changes linearly with load p_c^0 for smaller values of p_c^0. This is because at low loading, voltage constraints are inactive and reactive power compensation from inverters scales roughly linearly to minimize losses. As load increases, the apparent power constraint at bus 2 becomes active, thus causing an abrupt increase in q_5^g. The reactive setpoint increases until the apparent power constraint at bus 5 becomes active, creating a nonlinear relation with solar p_g^0. Further, solar generation and reactive capacity are inversely related. Hence, apparent power constraints become active at higher loads for decreasing solar.

However, not all points of the OPF surface in Figure 1 may occur in practice. For example, as load peaks in the evening but solar during daytime, the pair $(p_c^0, p_g^0) = (1, 0.25)$ is unlikely to occur. To capture more realistic grid conditions, we produced (p_c^0, p_g^0) pairs using the Smart* project data by summing up loads and solar generation across all buses. The two obtained time series were smoothed using a median filter of order 150 with MATLAB's `medfilt1`, and then scaled so their peaks matched the maximum values shown in the top panel of Fig. 2. Solving the OPF for these values yielded the red dots shown in Fig. 1. Points (A, B, C, D) noted on Fig. 1 and Fig. 2 (top) trace the path of optimal q_5^g across time.

In a nutshell, it may be more practical to learn the OPF mapping across the red-dot path rather than the entire 2-D grid of Figure 1. Hence, we estimated the OPF mapping along this path using GP-OPF and SI-GP-OPF. We collected $T = 48$ training OPF instances by sampling the OPF path of Figure 1 every 40 minutes between 6 AM till midnight. The estimated
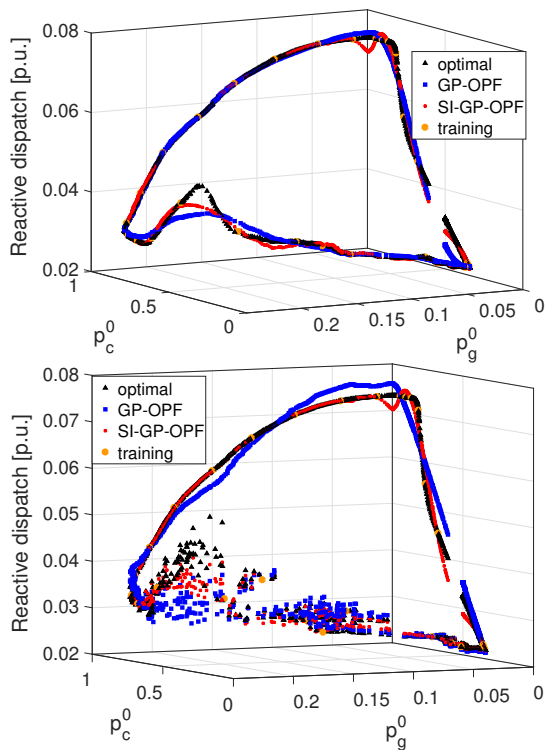
Fig. 3. Optimal reactive dispatch for bus 5 of the IEEE 13-bus benchmark and GP-OPF predictions under sunny (top) and cloudy (bottom) conditions.

Fig. 4. Optimal and predicted voltages at buses 12 (top) and 4 (bottom) of the IEEE 13-bus feeder under cloudy conditions.

minimizers illustrated in Figure 3 (top panel) show that both approaches succeeded at inferring the OPF mapping. Notably, the standard deviation for the estimated q_5^g across time dropped from 2.6×10^{-3} with GP-OPF to 0.03×10^{-3} with SI-GP-OPF. The previous test simulated relatively smooth solar generation p_g^0. To account for solar variability, we repeated the test for the solar p_g^0 plotted in the bottom panel of Fig. 2. This signal was generated by summing solar generations across all buses but without smoothing them with a median filter. As seen in Figure 3 (bottom panel), the SI-GP-OPF outperforms GP-OPF in terms of prediction accuracy, especially in areas of larger fluctuations. In such areas, the training datapoints may not be sufficiently many for GP-OPF; as the now non-smooth solar/load trajectories were sampled every 30 minutes again. By matching the OPF mapping gradients at training points, SI-GP-OPF was able to provide more accurate predictions.

GP models can be trained for any entry of the OPF minimizer $\mathbf{x}$ or other quantity of interest as long as its labels and sensitivities can be computed; they are not restricted to inverter setpoints. As an example, the previous test was repeated for inferring nodal voltages. For this test, the training data were the optimal voltages downsampled every 30 minutes. Figure 4 shows the voltage prediction results at buses 12 (top panel) and 4 (bottom panel) for a cloudy setup. The results confirm the effectiveness of GP-OPF and SI-GP-OPF for learning voltages across a feeder.

B. Tests on IEEE 123-bus Feeder

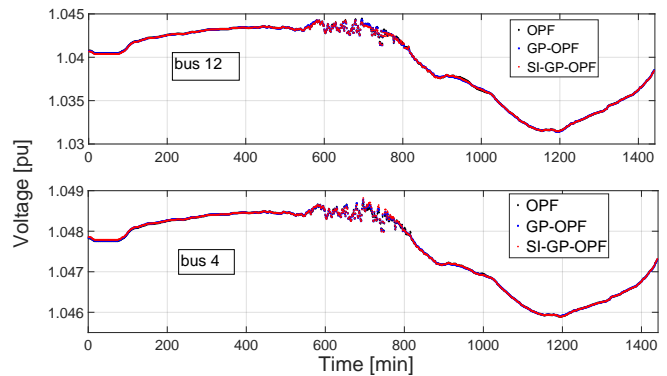GP-OPFs were also tested on the IEEE 123-bus benchmark. Here, solar and loads at each bus varied independently based

on real-world data from the Smart* Project. Buses with indices that are multiplies of 5 hosted DERs yielding a total of 17 inverters. Solar generation data were normalized so their peak values matched 50% of their nominal load value at that bus. Considering the period of 7:00 AM to 8:00 PM yielded 781 OPF instances related to 1-min intervals. Downsampling every 30 minutes gave $T = 27$ training datapoints. The optimal setpoints for all 17 inverters were learned using the three methods: GP-OPF, SI-GP-OPF, and RF-SI-GP-OPF. For the last one, the number of RFs was set to $D = 1,600$ striking a good trade-off between learning time and accuracy.

Figure 5 shows the boxplots for RPEs of optimal (re)active setpoints for all inverters and time instances. It also shows their standard deviations (STDs) as predicted by the GP models. Overall, SI-GP-OPF outperforms GP-OPF in terms of error and uncertainty, which confirms the advantage of incorporating sensitivities into learning. SI-GP-OPF is significantly better than GP-OPF primarily for active power setpoints. It is also evident that the RF-based approximation entails some performance degradation in exchange of computational speed-up to be detailed later.

While RPEs provide relative information on the estimation accuracy, insight is needed on how closely the estimated values follow the optimal setpoints. Figure 6 depicts the optimal setpoints and their predictions across all inverters and instances of the testing dataset. For clarity of presentation, the points have been sorted in increasing order with respect to the predicted value. The learned setpoints were obtained using RF-SI-GP-OPF. The shaded areas demonstrate the $\pm 3\sigma$ confidence interval obtained by taking the square root of the diagonal entries of the covariance matrix in (7b). These results visualize the improvement achieved by using sensitivities. Figure 6 shows high uncertainty intervals and low accuracy near negative reactive power setpoints. This is due to insufficient training data for over-voltage conditions. This can be solved by adding more labeled data in such instances. Figure 12 will later show how adding pertinent training data can solve such issues.

One may argue that nearly optimal inverter setpoints can be obtained by a linearized OPF, which enjoys improved computational complexity over the exact SOCP formulation of
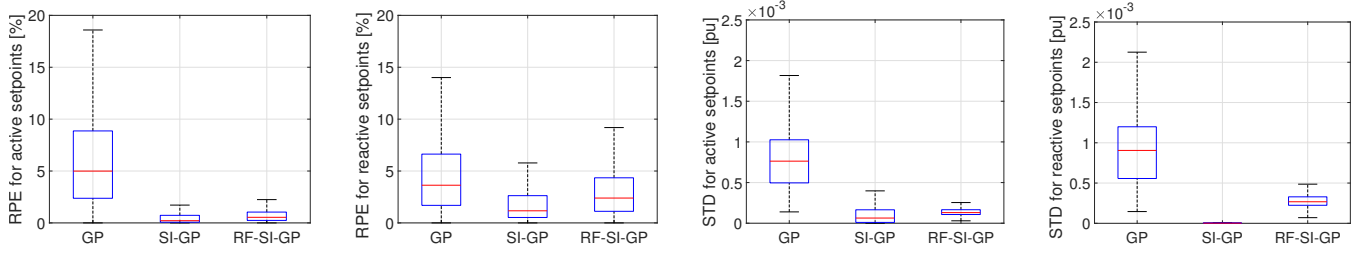
Fig. 5. Boxplots for RPEs and standard deviations (STD) while estimating optimal (re)active setpoints.
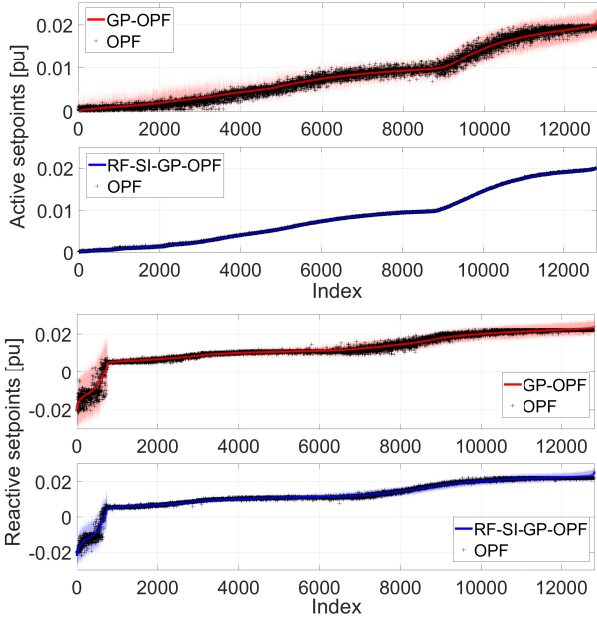

Fig. 6. Sorted optimal and estimated active (top) and reactive (bottom) setpoints over inverters and instances using GP-OPF and RF-SI-GP-OPF.
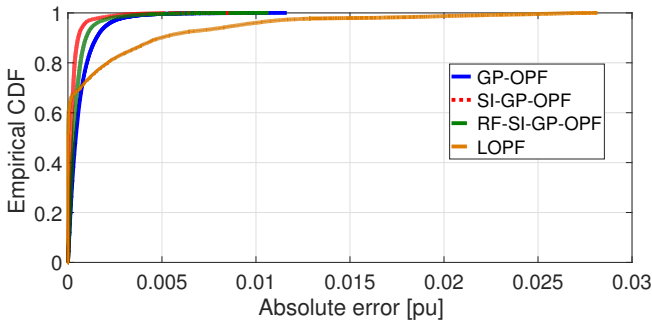

Fig. 7. Empirical CDF of error for predicting reactive setpoints.
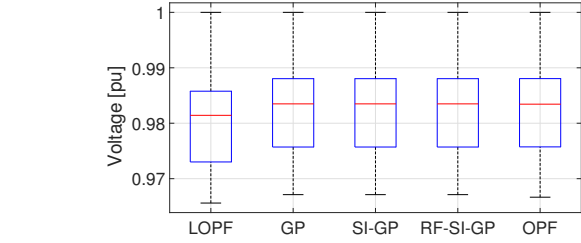

Fig. 8. Voltage across buses and testing instances obtained from solving the PF problem using the inverter dispatches obtained using different approaches.
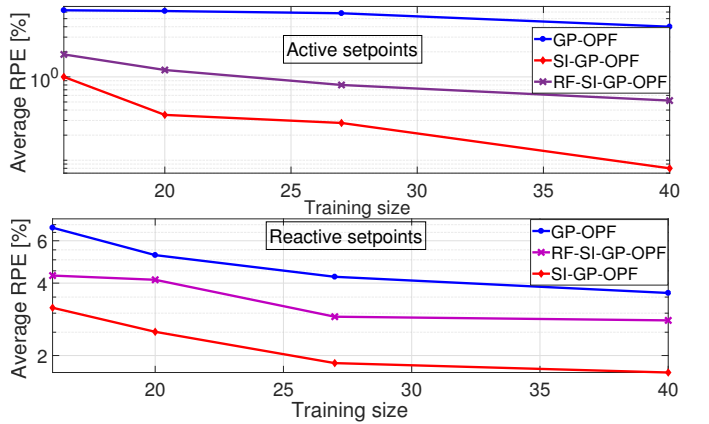

Fig. 9. Average RPE while estimating active (top) and reactive setpoints (bottom) using training datasets of different size T.

the OPF. To this end, we compared GP-OPF with the linearized OPF (LOPF) detailed in Appendix C. The LOPF in (27) approximates the AC-OPF of (3) by a quadratic program. LOPF does not model line currents explicitly, and hence, the current limits of (3e) were dropped. This does not harm the comparison since line limits of (3e) were not binding in our tests. We solved LOPF under the same grid conditions as in the previous tests. The average running time per LOPF instance was 0.292 seconds, which is over 20 times the prediction time of RF-SI-GP-OPF. Figure 7 shows the empirical

cumulative distribution function (CDF) of the absolute error of reactive setpoints obtained from LOPF, GP-OPF, SI-GP-OPF, and RF-SI-GP-OPF. The GP-based approaches were trained using $T = 27$. Figure 7 confirms the superior accuracy of a well-trained GP-OPF model over LOPF. This behavior was expected because the GP-based schemes are trained to follow the optimal setpoints, whereas the LOPF is solving an approximate optimization problem.

To verify feasibility of the setpoints predicted by GP-OPF and the setpoints computed by LOPF, we plugged these setpoints into the AC power flow equations and computed the induced grid voltages. To ease comparison, the substation voltage was set to $v_0 = 1$ pu. Figure 8 shows boxplots of voltages across buses and testing instances. The results confirm that the error of the learned setpoints propagated through PF equations, does not render any infeasibility. Further, the GP-based methods yield lower voltage deviations than the LOPF.

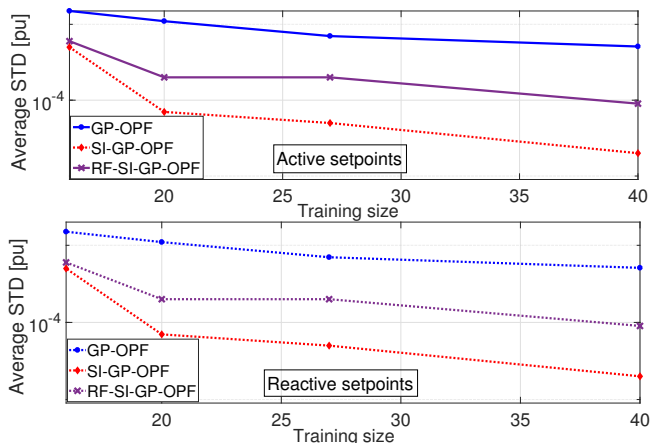To study the effect of training size T, the inverter dispatches

Fig. 10. Average standard deviation (STD) while estimating active (top) and reactive setpoints (bottom) using training datasets of different size T.

TABLE III
AVERAGE RUNNING TIMES [S] FOR PREDICTING OPTIMAL SETPOINTS FOR
ALL INVERTERS OF THE IEEE 123-BUS SYSTEM PER OPF INSTANCE.

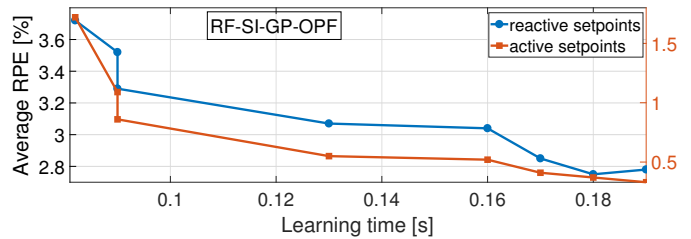|	Size of Training Dataset T			
	40	27	20	16
GP-OPF	0.004	0.002	0.002	0.001
SI-GP-OPF	0.853	0.388	0.228	0.141
RF-SI-GP-OPF	0.012	0.012	0.011	0.011

Fig. 11. Average RPE over prediction time for estimating active (right axis) and reactive (left axis) setpoints using RF-SI-GP-OPF. The number of random features D was increased (left to right) across the range $[600, 2000]$ with increments of 200. Larger D yield smaller errors but longer times.
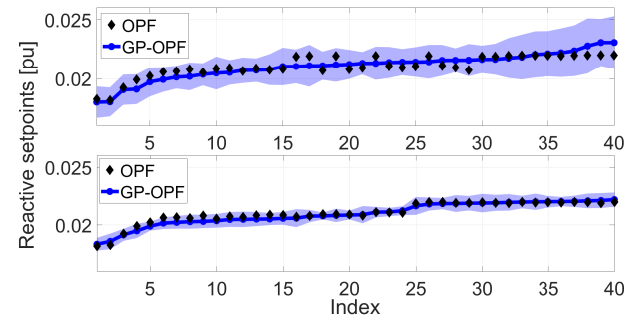

Fig. 12. Optimal and predicted reactive setpoints for inverter 10 over testing instances using GP-OPF. Instances have been sorted in increasing order based on predictions for presentation purposes. Shaded area corresponds to uncertainty interval of $\pm 2\sigma$ computed from (7b).

were learned for $T \in \{16, 20, 40\}$ by uniformly sampling the labeled data every $\{50, 40, 20\}$ minutes, respectively. To compare the estimation performance over T, we averaged RPEs and STDs over inverters and instances. Figures 9 and 10 show the results for predicting active (top) and reactive (bottom) setpoints. Table VII-B reports the average time of predicting setpoints across the feeder per OPF instance for each T. These times of course do not account for data generation. All learning methods are faster than solving an OPF, which takes 3.8 seconds per instance. The computation time for finding the sensitivities was 0.07 seconds, which is negligible compared to the OPF time. The tests further confirm that prediction accuracy improves with increasing T for all methods. Moreover, the sensitivity-informed GP-OPF featured lower RPE/STD than plain GP-OPF across T. For reactive setpoints, the RPE attained by GP-OPF using $T = 40$ is achieved by RF-SI-GP-OPF using less than $T = 25$.

Fixing $T = 27$, we studied the effect of D on running time and accuracy. The number of RFs varied across $[600, 2000]$ with increments of 200. Figure 11 shows the average RPE for inferring setpoints using RF-SI-GP-OPF. As anticipated, prediction time increases with D, whereas RPE is decreasing with D. Increasing D from $1,600$ to $2,000$ offers marginal prediction improvement. These results show the trade-off between accuracy and prediction time using RFs.

We subsequently explored the practical merit of uncertainty intervals. High predictive variances indicate that particular grid conditions $\boldsymbol{\theta}$'s were not well represented in the training set. Therefore, GP-OPF predictions with large variances identify areas in the parameter space $\boldsymbol{\theta}$ from which more labels $y(\boldsymbol{\theta})$

need to be sampled. To validate our hypothesis, we used K-means to cluster $\boldsymbol{\theta}$'s into 20 clusters. We trained a GP-OPF model using training data drawn only from 15 out of the 20 clusters. Testing data were selected by choosing 8 samples per each of the remaining 5 clusters. This sampling protocol ensured that testing data differed from training data. Figure 12 (top) depicts optimal and predicted reactive setpoints for inverter 10. For better presentation, setpoints have been sorted based on the predicted value. We repeated the test but now included training data from all 20 clusters. Figure 12 (bottom) shows the reactive setpoints for the same inverter and over the same OPF instances as those on the top panel. Of course, these testing instances were not part of the training dataset. As expected, the predictions of the bottom panel are closer to the optimal setpoints, and that is also verified by their smaller predictive variances.

To analyze the performance of the proposed GP-based schemes under more extreme grid conditions, the solar active power generation was further increased as follows: *a)* Inverters were placed at all non-zero-injection buses, thus increasing the total number of inverters to 85; and *b)* Load/solar injections were scaled to match the nominal benchmark values and twice the benchmark values, respectively. Under this setting, voltage deviations exceeded the $\pm 3\%$ per unit (pu) limit. Note that the ANSI-C.84.1 standard specifies a $\pm 5\%$ pu deviation limit for service voltages, i.e., voltages at the customer connection point. To account for the voltage drop between the service and distribution transformers, we adopted the common practice of aiming for a maximum of $\pm 3\%$ voltage deviation at distribution transformers; see e.g., [44]. For this test, we generated
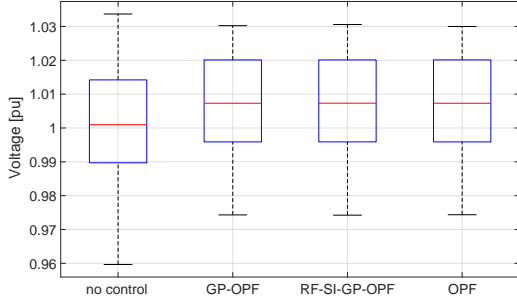
Fig. 13. Boxplots of AC voltages obtained under no inverter control, GP-OPF, RF-SI-GP-OPF, and AC-OPF.

more data by synthetically generating a set for a period of two days rather than one. This was accomplished by perturbing the data described in the first paragraph of Section VII with additive random Gaussian noise with variance of 0.001 pu. The training dataset was obtained by downsampling every 5 minutes of both days of data. The remaining instances from the second day were chosen as the testing dataset. The GPs were modeled using a Gaussian covariance with automatic relevance determination (ARD). The GP-OPF and RF-SI-GP-OPF yielded inverter setpoints that were subsequently used to solve the nonlinear PF equations. Figure 13 compares the obtained voltages with the voltages induced by an AC-OPF solution as well as with the voltages under no inverter control. The plots demonstrate that without inverter control, voltages exceed the $\pm 3\%$ deviation limits, whereas the GP-based schemes cause acceptable voltage deviations.

VIII. CONCLUSIONS

The GP-OPF is a fast and effective method for learning the OPF mapping for expedited inverter control. For large datasets, RF-GP-OPF features reduced computational complexity over GP-OPF. SI-GP-OPF improves estimation accuracy at the expense of increasing the training and operation speed, yet its RF-based counterpart reduces both training and prediction time. If a labeled training dataset of OPF solutions is available, GP-OPF predicts inverter setpoints faster than RF-SI-GP-OPF, though RF-SI-GP-OPF yields more accurate results. If a dataset is not available, then RF-SI-GP-OPF outperforms GP-OPF both in prediction accuracy and computational time. Both learning methods are considerably faster than solving the OPF.

Extensive numerical tests on the IEEE 13- and 123-bus feeders corroborate our findings. In particular, GP-OPF predicted near-optimal setpoints of the 123-bus system within 0.005 seconds, while solving the OPF took 3.8 seconds. SI-GP-OPF achieved the same prediction accuracy as GP-OPF by using only $1/4$ of the OPF instances (training labels). Computing the required sensitivities was posed as solving a set of linear equations, which took 0.07 seconds. Finally, RFs accelerated computations for SI-GP-OPF by 70 times while maintaining superior performance to the GP-OPF. The adopted Bayesian approach provided uncertainties that could flag unreliable learned setpoints.

This work sets the foundations for several exciting research directions. An interesting direction is to employ active learning to select the training data at locations with high uncertainty. This method can help reduce the training size while achieving high learning accuracy. Another practically relevant direction is to leverage spatiotemporal covariance between inverter dispatches and avoid learning the setpoints per inverter. SI-RF-GP-OPF models can also be used as digital twins or surrogates of the OPF in bilevel programming settings as they provide reasonable predictions for minimizers and their gradients alike.

APPENDIX

A. Random Feature Approximation

This appendix justifies the approximation in (12). According to Bochner's theorem [40], every continuous, shift-invariant kernel $k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ is the Fourier transform of a pdf $p(\mathbf{v})$ as

$$k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = \int_{-\infty}^{+\infty} e^{j\mathbf{v}^\top(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j)} p(\mathbf{v})\, \mathrm{d}\mathbf{v} = \mathbb{E}_{\mathbf{v}}[e^{j\mathbf{v}^\top(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j)}].$$

The Fourier transform of the Gaussian kernel in (8) is known to be also a Gaussian yet of the inverse variance. Therefore, the pdf associated with the Gaussian kernel is $p(\mathbf{v}) = \mathcal{N}(\mathbf{0}, \beta \mathbf{I}_M)$. This is up to some scaling constants that can be absorbed into parameter α. Leveraging this equivalence, one can draw D samples $\{\mathbf{v}_d\}_{d=1}^D$ from $\mathcal{N}(\mathbf{0}, \beta \mathbf{I}_M)$, and compute a sample estimate of the earlier expectation as

$$\hat{k}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) := \frac{1}{D} \sum_{d=1}^D e^{j\mathbf{v}_d^\top \boldsymbol{\theta}_i} e^{-j\mathbf{v}_d^\top \boldsymbol{\theta}_j} = \boldsymbol{\zeta}^\top(\boldsymbol{\theta}_j) \boldsymbol{\zeta}^*(\boldsymbol{\theta}_i) \quad (24)$$

where * denotes complex conjugation and $\boldsymbol{\zeta}(\boldsymbol{\theta})$ is a D-length vector whose d-th entry is defined as $\zeta_d(\boldsymbol{\theta}) := e^{j\mathbf{v}_d^\top \boldsymbol{\theta}}/\sqrt{D}$. Equation (24) provides an unbiased estimate of $k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$ and its variance decreases as $1/D^2$. To avoid working with complex-valued feature vectors, note that

$$k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = \mathbb{E}_{\mathbf{v}}[e^{j\mathbf{v}^\top(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j)}] = \mathbb{E}_{\mathbf{v}}[\cos(\mathbf{v}^\top(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j))] \quad (25)$$

This follows from Euler's identity and upon noting that $\mathbb{E}_{\mathbf{v}}\left[\sin\left(\mathbf{v}^\top(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j)\right)\right] = 0$ because $\sin$ is an odd function and is applied to the zero-mean random variable $\mathbf{v}^\top(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j)$. The quantity $\cos(\mathbf{v}^\top(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j))$ can be expressed as the inner product between two $2D$-length vectors [39]. To express this quantity as the inner product between D-length vectors as in (24), introduce an auxiliary random variable ϕ that is drawn uniformly from $[0, 2\pi]$, so that (25) can be expressed as [40]

$$k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = \mathbb{E}_{\mathbf{v}, \phi}\left[2\cos\left(\mathbf{v}^\top\boldsymbol{\theta}_i + \phi\right)\cos\left(\mathbf{v}^\top\boldsymbol{\theta}_j + \phi\right)\right]. \quad (26)$$

To see that (26) is equivalent to (25), use trigonometric identities and observe $\mathbb{E}_{\phi}[\cos(\mathbf{v}^\top(\boldsymbol{\theta}_i + \boldsymbol{\theta}_j) + 2\phi)] = 0$ for all $\mathbf{v}$. The approximation in (12) is a sample estimate of (26) obtained upon drawing D samples of $(\mathbf{v}_d, \phi_d)$.

B. Building the Linear System of (23)

Matrices $\mathbf{S}$ and $\mathbf{U}$ in (23) can be constructed from (22) as

$$\mathbf{S} = \begin{bmatrix} \mathbf{A}_e & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{S}_{21} & \mathbf{A}_e^\top & \mathbf{A}_i^\top & \mathbf{S}_{24} \\ \mathrm{dg}(\boldsymbol{\mu})\mathbf{A}_i & \mathbf{0} & \mathbf{S}_{33} & \mathbf{0} \\ \mathbf{S}_{41} & \mathbf{0} & \mathbf{0} & \mathbf{S}_{44} \end{bmatrix}, \quad \mathbf{U} = \begin{bmatrix} \mathbf{B}_e \\ \mathbf{0} \\ \mathrm{dg}(\boldsymbol{\mu})\mathbf{B}_i \\ \mathbf{0} \end{bmatrix}$$

where the m-th column of $\mathbf{S}_{24}$ is $\frac{\mathbf{A}_m^\top \mathbf{A}_m \mathbf{x}}{\|\mathbf{A}_m \mathbf{x}\|} - \mathbf{b}_m \; \forall m$, and

$$\mathbf{S}_{21} = \sum_{m=1}^{2N} \nu_m \left(\frac{\mathbf{A}_m^\top \mathbf{A}_m}{\|\mathbf{A}_m \mathbf{x}\|} - \frac{\mathbf{A}_m^\top \mathbf{A}_m \mathbf{x} \mathbf{x}^\top \mathbf{A}_m^\top \mathbf{A}_m}{\|\mathbf{A}_m \mathbf{x}\|^3} \right)$$

$$\mathbf{S}_{33} = \mathrm{dg}(\mathbf{A}_i \mathbf{x} - \mathbf{B}_i \boldsymbol{\theta} - \mathbf{f}_i)$$

$$\mathbf{S}_{41} = \mathrm{dg}(\boldsymbol{\nu}) \mathbf{S}_{24}^\top, \quad \mathbf{S}_{44} = \mathrm{dg}(\{\|\mathbf{A}_m \mathbf{x}\| - \mathbf{b}_m^\top \mathbf{x} - f_m\}_{m=1}^{2N}).$$

C. Linearized OPF (LOPF)

We cast an approximate OPF relying on a linearization of the power flow equations [cf. (27c)]. This LOPF can be expressed as the quadratic program

$$\min \quad \mathbf{p}^\top \mathbf{1} + \mathbf{p}^\top \mathbf{R} \mathbf{p} + \mathbf{q}^\top \mathbf{R} \mathbf{q} \tag{27a}$$

$$\text{over} \quad \{p_n^g, q_n^g\}_{n \in \mathcal{N}_g}, \{v_n\}_{n=1}^N, v_0 \tag{27b}$$

$$\text{s.to} \quad \mathbf{v} = \mathbf{R}\mathbf{p} + \mathbf{X}\mathbf{q} + v_0 \mathbf{1} \tag{27c}$$

$$(3f), (3g) \tag{27d}$$

$$\left| p_n^g \cos\left(\tfrac{k\pi}{16}\right) + q_n^g \sin\left(\tfrac{k\pi}{16}\right) \right| \le \bar{s}_n^g, \; k = 1 : 16. \tag{27e}$$

where vectors $(\mathbf{p}, \mathbf{q})$ collect the net power injections defined in (1), and vector $\mathbf{v}$ approximates nodal voltages as an affine function of injections with positive definite matrices $(\mathbf{R}, \mathbf{X})$; see e.g., [6]. The term $\mathbf{p}^\top \mathbf{R} \mathbf{p} + \mathbf{q}^\top \mathbf{R} \mathbf{q}$ is an approximation (second-order Taylor's series expansion in fact) of ohmic line losses [6, Prop. 1]. Constraint (27e) approximates the quadratic constraint in (3h) using a 32-vertex polytope [4].

REFERENCES

[1] K. Turitsyn, P. Sulc, S. Backhaus, and M. Chertkov, "Options for control of reactive power by distributed photovoltaic generators," *Proc. IEEE*, vol. 99, no. 6, pp. 1063–1073, Jun. 2011.

[2] X. Zhou, M. Farivar, Z. Liu, L. Chen, and S. H. Low, "Reverse and forward engineering of local voltage control in distribution networks," *IEEE Trans. Autom. Contr.*, vol. 66, no. 3, pp. 1116–1128, Mar. 2021.

[3] V. Kekatos, L. Zhang, G. B. Giannakis, and R. Baldick, "Voltage regulation algorithms for multiphase power distribution grids," *IEEE Trans. Power Syst.*, vol. 31, no. 5, pp. 3913–3923, Sep. 2016.

[4] R. A. Jabr, "Linear decision rules for control of reactive power by distributed photovoltaic generators," *IEEE Trans. Power Syst.*, vol. 33, no. 2, pp. 2165–2174, Mar. 2018.

[5] S. Low, "Convex relaxation of optimal power flow — Part II: Exactness," *IEEE Trans. Control of Network Systems*, vol. 1, no. 2, pp. 177–189, Jun. 2014.

[6] S. Taheri, M. Jalali, V. Kekatos, and L. Tong, "Fast probabilistic hosting capacity analysis for active distribution systems," *IEEE Trans. Smart Grid*, no. 3, pp. 2000–2012, May 2021.

[7] S. Karagiannopoulos, P. Aristidou, and G. Hug, "Data-driven local control design for active distribution grids using off-line optimal power flow and machine learning techniques," *IEEE Trans. Smart Grid*, vol. PP, no. 99, pp. 1–1, 2019.

[8] M. Jalali, V. Kekatos, N. Gatsis, and D. Deka, "Designing reactive power control rules for smart inverters using support vector machines," *IEEE Trans. Smart Grid*, vol. 11, no. 2, pp. 1759–1770, Mar. 2020.

[9] X. Pan, T. Zhao, and M. Chen, "DeepOPF: Deep neural network for DC optimal power flow," in *Proc. IEEE Intl. Conf. on Smart Grid Commun.*, Beijing, China, Oct. 2019, pp. 1–6.

[10] A. Velloso and V. Pascal, "Combining deep learning and optimization for preventive security-constrained DC optimal power flow," *IEEE Trans. Power Syst.*, vol. 36, no. 4, pp. 3618–3628, Jul. 2021.

[11] A. Zamzam and K. Baker, "Learning optimal solutions for extremely fast AC optimal power flow," in *Proc. IEEE Intl. Conf. on Smart Grid Commun.*, Tempe, AZ, Nov. 2020, pp. 1–6.

[12] N. Guha, Z. Wang, M. Wytock, and A. Majumdar, "Machine learning for AC optimal power flow," in *Climate Change Workshop at ICML*, Long Beach, CA, Jun. 2019, pp. 1–4.

[13] W. Wang, N. Yu, Y. Gao, and J. Shi, "Safe off-policy deep reinforcement learning algorithm for Volt-VAR control in power distribution systems," *IEEE Trans. Smart Grid*, vol. 11, no. 4, pp. 3008–3018, Jul. 2020.

[14] Q. Zhang, K. Dehghanpour, Z. Wang, F. Qiu, and D. Zhao, "Multi-agent safe policy learning for power management of networked microgrids," *IEEE Trans. Smart Grid*, vol. 12, no. 2, pp. 1048–1062, Mar. 2021.

[15] S. Gupta, V. Kekatos, and M. Jin, "Controlling smart inverters using proxies: A chance-constrained DNN-based approach," *IEEE Trans. Smart Grid*, vol. 13, no. 2, pp. 1310–1321, Mar. 2022.

[16] ——, "Deep learning for reactive power control of smart inverters under communication constraints," in *Proc. IEEE Intl. Conf. on Smart Grid Commun.*, Tempe, AZ, 2020, pp. 1–6.

[17] M. K. Singh, S. Gupta, V. Kekatos, G. Cavraro, and A. Bernstein, "Learning to optimize power distribution grids using sensitivity-informed deep neural networks," in *Proc. IEEE Intl. Conf. on Smart Grid Commun.*, Tempe, AZ, Nov. 2020, pp. 1–6.

[18] M. K. Singh, V. Kekatos, and G. B. Giannakis, "Learning to solve the AC-OPF using sensitivity-informed deep neural networks," *IEEE Trans. Power Syst.*, vol. 37, no. 4, pp. 2833–2846, Jul. 2022.

[19] F. Fioretto, T. W. Mak, and P. V. Hentenryck, "Predicting AC optimal power flows: Combining deep learning and Lagrangian dual methods," in *AAAI Conf. on Artificial Intelligence*, New York, NY, Feb. 2020.

[20] R. Nellikkath and S. Chatzivasileiadis, "Physics-informed neural networks for AC optimal power flow." [Online]. Available: https://arxiv.org/abs/2110.02672

[21] X. Lei, Z. Yang, J. Yu, J. Zhao, Q. Gao, and H. Yu, "Data-driven optimal power flow: A physics-informed machine learning approach," *IEEE Trans. Power Syst.*, vol. 36, no. 1, pp. 346–354, Jun. 2021.

[22] P. Pareek and H. D. Nguyen, "Gaussian process learning-based probabilistic optimal power flow," *IEEE Trans. Power Syst.*, vol. 36, no. 1, pp. 541–544, Jan. 2021.

[23] M. Jalali, V. Kekatos, S. Bhela, H. Zhu, and V. Centeno, "Inferring power system dynamics from synchrophasor data using Gaussian processes," *IEEE Trans. Power Syst.*, Nov. 2021, (early access).

[24] M. Jalali, V. Kekatos, S. Bhela, and H. Zhu, "Inferring power system frequency oscillations using Gaussian processes," in *Proc. IEEE Conf. on Decision and Control*, Austin, TX, Dec. 2021.

[25] P. Pareek and H. D. Nguyen, "A framework for analytical power flow solution using Gaussian process learning," *IEEE Trans. Sustain. Energy*, vol. 13, no. 1, pp. 452–463, 2022.

[26] P. Pareek, W. Yu, and H. D. Nguyen, "Optimal steady-state voltage control using Gaussian process learning," *IEEE Trans. Ind. Inform.*, vol. 17, no. 10, pp. 7017–7027, Oct. 2021.

[27] K. Almeida, F. Galiana, and S. Soares, "A general parametric optimal power flow," *IEEE Trans. Power Syst.*, vol. 9, no. 1, pp. 540–547, Feb. 1994.

[28] V. Ajjarapu and N. Jain, "Optimal continuation power flow," *Electric Power Systems Research*, vol. 35, no. 1, pp. 17–24, Oct. 1995.

[29] J. F. Bonnans and A. Shapiro, *Perturbation Analysis of Optimization Problems.* New York, NY: Springer Science & Business Media, 2000.

[30] A. J. Conejo, E. Castillo, R. Minguez, and R. Garcia-Bertrand, *Decomposition techniques in mathematical programming.* Springer, 2006.

[31] A. Agrawal, S. Barratt, S. Boyd, E. Busseti, and W. M. Moursi, "Differentiating through a cone program," *Journal of Applied and Numerical Optimization*, vol. 1, no. 2, pp. 107–15, 2019.

[32] M. Farivar and S. Low, "Branch flow model: Relaxations and convexification — Part I," *IEEE Trans. Power Syst.*, vol. 28, no. 3, pp. 2554–2564, Aug. 2013.

[33] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning.* Cambridge, MA: MIT Press, 2006.

[34] C. M. Bishop, *Pattern Recognition and Machine Learning.* New York, NY: Springer, 2006.

[35] V. Kekatos, Y. Zhang, and G. B. Giannakis, "Electricity market forecasting via low-rank multi-kernel learning," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 6, pp. 1182–1193, Dec. 2014.

[36] A. V. Fiacco, "Sensitivity analysis for nonlinear programming using penalty methods," *Mathematical Programming*, vol. 10, no. 1, pp. 287–311, Dec. 1976.

[37] E. Snelson and Z. Ghahramani, "Sparse Gaussian processes using pseudo-inputs," in *Intl. Conf. on Neural Information Processing Systems*, BC, Canada, Dec. 2005, p. 1257–1264.

[38] D. Achlioptas, F. McSherry, and B. Schölkopf, "Sampling techniques for kernel methods," in *Advances in Neural Information Processing Systems*, BC, Canada, Sep. 2002, pp. 335–342.

[39] Y. Shen, T. Chen, and G. B. Giannakis, "Random feature-based online multi-kernel learning in environments with unknown dynamics," *J. of Machine Learning Research*, vol. 20, no. 1, p. 773–808, Jan. 2019.

[40] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Advances in Neural Information Processing Systems*, vol. 20, Dec. 2008, pp. 1177–1184.

[41] M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret, "Applications of second-order cone programming," *Linear Algebra and its Applications*, vol. 284, no. 1, pp. 193–228, 1998.

[42] A. Hauswirth, S. Bolognani, G. Hug, and F. Dorfler, "Generic existence of unique Lagrange multipliers in AC optimal power flow," *IEEE Contr. Syst. Lett.*, vol. 2, no. 4, pp. 791–796, Oct. 2018.

[43] S. Barker, A. Mishra, D. Irwin, E. Cecchet, P. Shenoy, and J. Albrecht, "An open data set and tools for enabling research in sustainable homes," in *Workshop on Data Mining Applications in Sustainability (SustKDD)*, Beijing, China, Aug. 2012, pp. 1–6.

[44] W. H. Kersting, *Distribution System Modeling and Analysis*. New York, NY: CRC Press, 2018.

Georgios B. Giannakis (Fellow, IEEE) received the Diploma in electrical engineering from the National Technical University of Athens, Athens, Greece, in 1981, the M.Sc. degree in electrical engineering, the M.Sc. degree in mathematics, and the Ph.D. degree in electrical engineering from the University of the Southern California, Los Angeles, CA, USA, in 1983, 1986, and 1986, respectively. He was a faculty member with the University of Virginia, Charlottesville, VA, USA, from 1987 to 1998, and since 1999, he has been a Professor with the University of Minnesota, Minneapolis, MN, USA, where he held an ADC Endowed Chair of telecommunications, was the Director of the Digital Technology Center from 2008 to 2021, and since 2016, he is has been a University of Minnesota McKnight Presidential Chair in ECE. He has authored or coauthored more than 480 journal papers, 780 conference papers, 25 book chapters, two edited books, and two research monographs in his research field, which include statistical learning, signal processing, communications, and networking. His current research include data science, and network science with applications to the Internet of Things, and power networks with renewables. He is the co-inventor of 34 issued patents. He was the co-recipient of the ten Best Journal Paper Awards from the IEEE Signal Processing (SP) and Communications Societies, including the G. Marconi Prize Paper Award in Wireless Communications. He was also recipient of the IEEE-SPS Norbert Wiener Society Award (2019), the EURASIP's A. Papoulis Society Award (2020), Technical Achievement Awards from the IEEE-SPS (2000) and from EURASIP (2005), the IEEE ComSoc Education Award (2019), and the IEEE Fourier Technical Field Award (2015). He is a Foreign Member of the Academia Europaea, and a Fellow of the National Academy of Inventors, the European Academy of Sciences, and EURASIP. He has served the IEEE in a number of posts, including a Distinguished Lecturer for the IEEE-SPS.

Mana Jalali received the B.S. degree from the University of Tehran, Tehran, Iran, in 2017; and the M.Sc. and Ph.D. degrees from Virginia Tech, Blacksburg, VA, USA, in 2019 and 2022, respectively; both in electrical engineering. She is now a power systems researcher at Invenia Labs, Cambridge, UK. During the summer of 2021, she interned at Engineering Analytics and Modeling group of Dominion Energy, VA, USA. Her research interests are on the application of optimization and machine learning on smart grids.

Manish K. Singh received the B.Tech. degree from the Indian Institute of Technology (BHU), Varanasi, India, in 2013; and the M.S. and Ph.D. degrees in electrical engineering from Virginia Tech, Blacksburg, VA, USA, in 2018 and 2021, respectively. He is currently a postdoctoral researcher with the University of Minnesota, Minneapolis, MN. During 2013-2016, he worked as an Engineer in the Smart Grid Dept. of POWERGRID, the central transmission utility of India. His research interests are focused on optimization, control, and learning techniques to develop algorithmic solutions for the operation and analysis of electric power systems as well as water and natural gas networks.

Chen-Ching Liu (S'80–M'83–SM'90– F'94–LF) is American Electric Power Professor and Director of the Power and Energy Center at Virginia Tech, VA, USA. He is also an Adjunct Full Professor of University College Dublin, Ireland. Prof. Liu was a recipient of the IEEE PES Outstanding Power Engineering Educator Award in 2004. He served as the Chair of the IEEE PES Technical Committee on Power System Analysis, Computing, and Economics from 2005 to 2006. He is the U.S. Member of CIGRE Study Committee D2, Information Systems and Telecommunication. Dr. Liu is a Member of the U.S. National Academy of Engineering and Life Fellow of the IEEE.

Vassilis Kekatos (SM'16) is an Associate Professor with the Bradley Dept. of ECE at Virginia Tech. He obtained his Diploma, M.Sc., and Ph.D. in computer science and engineering from the Univ. of Patras, Greece, in 2001, 2003, and 2007, respectively. He is a recipient of the NSF Career Award in 2018 and the Marie Curie Fellowship. He has been a research associate with the ECE Dept. at the Univ. of Minnesota, where he received the postdoctoral career development award (honorable mention). During 2014, he stayed with the Univ. of Texas at Austin and the Ohio State Univ. as a visiting researcher. His research focus is on optimization and learning for future energy systems. He is currently serving in the editorial board of the IEEE Trans. on Smart Grid.