Inverse Optimal Power Flow: Assessing the Vulnerability of Power Grid Data

Priya L. Donti
Dept. of Computer Science
Dept. of Engr. & Public Policy
Carnegie Mellon University
Pittsburgh, PA 15213
pdonti@cs.cmu.edu

Inês Lima Azevedo
Dept. of Engr. & Public Policy
Carnegie Mellon University
Pittsburgh, PA 15213
iazevedo@cmu.edu

J. Zico Kolter
Dept. of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
zkolter@cs.cmu.edu

Abstract

We formulate inverse optimal power flow, an algorithm that assesses the extent to which private power grid data is exposed by publicly-available data by inverting the AC optimal power flow optimization problem. We find that we are able to learn private information such as electricity generation costs and (to some extent) grid structural parameters on a 14-node test case.

1 Introduction

In the electricity sector, there is a great need to protect critical information that could compromise fair electricity market operation or power grid cybersecurity [1, 2]. At the same time, grid operators such as PJM and governmental entities such as the Environmental Protection Agency regularly publish quantities such as five-minute electricity prices and hourly power outputs of electricity generators for the purposes of market transparency and emissions monitoring. We investigate the question of whether and to what extent critical power grid information is exposed by published information, given our knowledge that these private and public quantities are related via an optimization problem called AC optimal power flow (ACOPF). To do this, we formulate an algorithm called *inverse optimal power flow* (inverse OPF) that uses a neural network to learn private quantities from public quantities.

2 Related work

Inverse problems. Inverse problems seek to predict model inputs or decision parameters from model outputs. For instance, prior work has used techniques from game theory, graph theory, and bi-level optimization to identify power grid structure [3] and energy demands [4] from grid outputs. Our work seeks to apply techniques from optimization and deep learning to inverse power flow.

Differentiating through optimization problems. Recent work has explored differentiating through optimization problems within deep networks [5, 6]. We employ some of the innovations in differentiable quadratic programming to formulate and solve our inverse optimal power flow problem.

3 Background: AC optimal power flow

We now present the AC optimal power flow (ACOPF) optimization problem [7], which plays a crucial role in our formulation of inverse optimal power flow. ACOPF is solved by power system operators to pick power system quantities that minimize the overall cost of delivering power. Specifically, for a power grid with n nodes, operators must determine quantities $z \equiv \begin{bmatrix} \text{angle}(v)^T & |v|^T & p_g^T & q_g^T \end{bmatrix}^T$,

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

where $v \in \mathbb{C}^n$ are the voltages at each node and $p_g, q_g \in \mathbb{R}^n$ are the real and imaginary parts of the power injections (e.g. from electricity generators) at all system nodes. These quantities solve

$$
\begin{array}{lll}
\underset{z \equiv \left[\text{angle}(v)^T \;\; |v|^T \;\; p_g^T \;\; q_g^T\right]^T}{\text{minimize}} & f_c(p_g) & \\
\text{subject to} & Az = b & \text{(linear equality constraints)} \\
& Gz \leq h & \text{(linear inequality constraints)} \\
& (p_g - p_d) + (q_g - q_d)j = \text{diag}(v)\bar{Y}\bar{v} & \text{(power flow constraint).}
\end{array}
\tag{1}
$$

Here, $f_c : \mathbb{R}^n \to \mathbb{R}$ is a cost function parameterized by electricity generation costs c; $p_d, q_d \in \mathbb{R}^n$ are the real and imaginary power demands at all system nodes; $Y \in \mathbb{C}^{n \times n}$ is the *nodal admittance matrix* that describes how power flows throughout the system; the linear equality and inequality constraints encode attributes of system nodes, lines, and generators; and we use the notation $\bar{x}$ to denote the complex conjugate of x. We note that the dual variable $\lambda \in \mathbb{R}^n$ on the power flow constraint corresponds to the electricity prices at each node.

In general, problem (1) is non-convex and NP-hard. As such, it is in practice common to assume that the objective is quadratic, linearize the power flow constraint using its Jacobian J at some point z_0 [7], and then solve the resulting problem iteratively via sequential quadratic programming [8]. Specifically, we write the linearized quadratic program corresponding to (1) as

$$
\begin{array}{ll}
\underset{z = \left[\text{angle}(v)^T \;\; |v|^T \;\; p_g^T \;\; q_g^T\right]^T}{\text{minimize}} & p_g^T \, \text{diag}(c_q) p_g + c_a^T p_g \\
\text{subject to} & \tilde{A}z = \tilde{b} \\
& Gz \leq h,
\end{array}
\tag{2}
$$

where $\tilde{A} = \left[A^T \quad J(z_0)^T\right]^T$ and $\tilde{b} = \left[b^T \quad k(z_0)^T\right]^T$ collect both the original linear constraints and the linearized power flow constraint $J(z_0)z = k(z_0)$, and where $c = \left[c_q^T \quad c_a^T\right]^T$ now contains the quadratic and linear cost parameters $c_q, c_a \in \mathbb{R}^n$, respectively, for power generation at each node.

4 Inverse optimal power flow

We now describe our inverse optimal power flow algorithm, which attempts to learn private electric grid information from public information via ACOPF. Specifically, given public information on real powers p_g, p_d and electricity prices λ, we seek to estimate private generator cost parameters c and the nodal admittance matrix Y, where all variables are as described in Section 3. We do so by constructing estimates $\hat{c}^\star$ and $\hat{Y}^\star$ of c and Y, respectively, whose corresponding ACOPF outputs are close to the true values of the publicly-available quantities p_g and λ. Mathematically, this problem can be formulated under some loss function ℓ on publicly-available quantities as

$$
\begin{array}{ll}
\hat{c}^\star, \hat{Y}^\star = \underset{\hat{c}, \hat{Y}}{\text{argmin}} & \ell\Big((p_g, \lambda), (\hat{p_g}, \hat{\lambda})\Big) \\
\text{subject to} & \hat{p_g}, \hat{\lambda} = \text{ACOPF}(\hat{c}, \hat{Y}, p_d),
\end{array}
\tag{3}
$$

where the constraint denotes that $\hat{p_g}$ and $\hat{\lambda}$ are the values of generator power injections and power prices produced by solving the ACOPF problem (1) with cost parameters $\hat{c}$, admittance matrix $\hat{Y}$, and nodal power demands p_d. We solve this problem iteratively via Algorithm 1, using backpropagation within a neural network to compute the needed gradients. We note that while (3) maximizes the agreement between true and estimated *public* quantities, the objective of actual interest is the agreement between the true and estimated *private* quantities. However, there are potentially multiple distinct sets of inputs to ACOPF that would produce identical public outputs. Thus, we must use enough data when executing Algorithm 1 to ensure that there is a unique set of private parameters that can produce the correct public outputs across *all* input data points.

4.1 Optimizing the inverse OPF problem

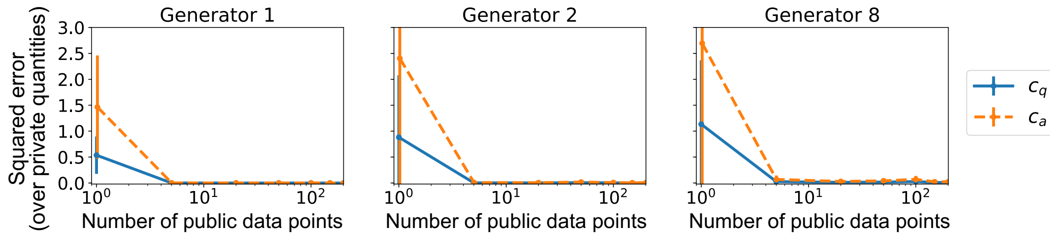

Figure 1: Squared error of guesses for quadratic (c_q) and linear (c_a) generator costs when all generators' costs are unknown (lower is better). Each plotted point represents five runs over a given amount of public data. We find that all cost parameters are identifiable with as little as 5 data points.

Algorithm 1 Inverse OPF Optimization

input: $\{(p_g^{(i)}, \lambda^{(i)}) \mid i = 1, \ldots, m\}$ *// public data*
initialize $\hat{c}, \hat{Y}$ *// some initial guess*
for $t = 1, \ldots, T$ **do**
  for $i = 1, \ldots, m$ **do**
    compute $\ell\left((p_g^{(i)}, \lambda^{(i)}), (\hat{p_g}^{(i)}, \hat{\lambda}^{(i)})\right)$
    // update guesses if loss has not converged
    if $\ell((p_g^{(i)}, \lambda^{(i)}), (\hat{p_g}^{(i)}, \hat{\lambda}^{(i)})) \neq 0$ **then**
      update $\hat{c}$ with $\nabla_{\hat{c}}\, \ell((p_g^{(i)}, \lambda^{(i)}), (\hat{p_g}^{(i)}, \hat{\lambda}^{(i)}))$
      update $\hat{Y}$ with $\nabla_{\hat{Y}}\, \ell((p_g^{(i)}, \lambda^{(i)}), (\hat{p_g}^{(i)}, \hat{\lambda}^{(i)}))$
    else
      return $\hat{c}, \hat{Y}$
    end if
  end for
end for

The main technical challenge of this approach is in computing the gradients $\nabla_\theta\, \ell((p_g^{(i)}, \lambda^{(i)}), (\hat{p_g}^{(i)}, \hat{\lambda}^{(i)}))$ for each $\theta \in \{\hat{c}, \hat{Y}\}$, as this involves taking the gradient through the solutions to the ACOPF optimization problem. Specifically, we must compute the terms

$$\frac{\partial \ell}{\partial \theta} = \frac{\partial \ell}{\partial \hat{p_g}(\theta)} \frac{\partial \hat{p_g}(\theta)}{\partial \theta} + \frac{\partial \ell}{\partial \hat{\lambda}(\theta)} \frac{\partial \hat{\lambda}(\theta)}{\partial \theta}, \tag{4}$$

where $\frac{\partial \hat{p_g}(\theta)}{\partial \theta}$ and $\frac{\partial \hat{\lambda}(\theta)}{\partial \theta}$ are the Jacobians of optimal primal and dual variables, respectively, in problem (1), with respect to our parameter estimate θ (and where we denote the dependence of $\hat{p_g}$ and $\hat{\lambda}$ on each θ here explicitly). To compute these Jacobians, we use the method presented in [5] to take gradients through the optimal quadratic program (2) solved during the last iteration of sequential quadratic programming. At a high level, this involves differentiating through the KKT optimality conditions of (2) and using the implicit function theorem to get a set of linear equations we can solve to get the necessary gradients. More details on this approach are described in Appendix A.

5 Experiments

We test our algorithm on a modified version of the IEEE 14-bus test case with three generators located at nodes 1, 2, and 8, respectively. More details about this system are included in Appendix B. We train our network on up to 201 public outputs generated from Grid Optimization (GO) Competition simulations [9]. We use the loss function

$$\ell((p_g, \lambda), (\hat{p_g}, \hat{\lambda})) = 100\|p_g - \hat{p_g}\|_2^2 + \|\lambda - \hat{\lambda}\|_2^2$$

for (1), where the weighting term adjusts for differences in magnitude between values of p_g and λ.

5.1 Cost parameters

We test the scenario in which all electricity generation cost parameters are unknown (but the admittance matrix is known). Results for runs over different amounts of training data are shown in Figure 1, with initial guesses for each cost parameter sampled from a Gaussian distribution to encode market participants' prior knowledge of cost distributions. We find that we are able to completely learn the cost parameters for this system with as little as 5 public data points. Even though our test system is small, given that real power grid data is published with hourly granularity (i.e. 8760 data points per year), there is cause to believe that publicly-available data may expose generator cost parameters on the actual power system as well.

5.2 Admittance matrix parameters

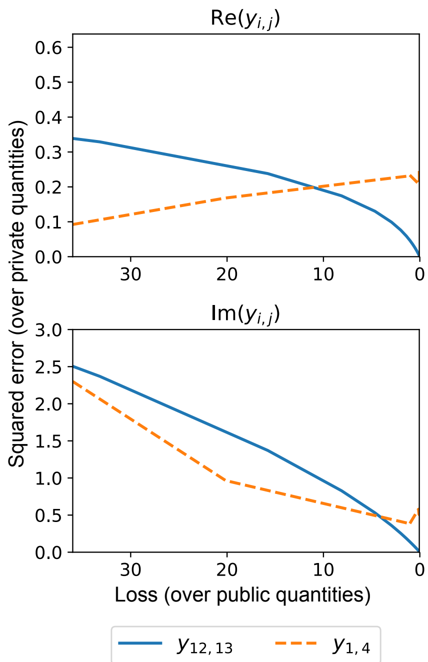

Figure 2: Squared error of sample admittance matrix parameters (real and imaginary parts plotted separately) as training loss on our 201 public data points goes to zero. Our estimate for $Y_{12,13}$ converges, but our estimate for $Y_{1,4}$ diverges.

Admittance matrix parameters are potentially more challenging to learn than cost parameters, as the choice of admittance matrix parameters can potentially render problem (1) infeasible before or during training. As such, in our preliminary experiments, we test whether we can learn one admittance matrix parameter at a time, where our initial guess involves perturbing this parameter with Gaussian noise reflecting the variability across all admittance matrix parameters. (We assume all other cost and admittance matrix parameters are known.) As illustrated via representative results in Figure 2, our preliminary tests suggest that some admittance matrix parameters are readily identifiable while others may be harder to identify.

6 Conclusions and future work

We find that published power grid information can expose private information. Future work includes a more thorough investigation of admittance matrix parameters on the 14-node system, as well as assessments on larger systems. While we address power systems here, our method could be applied to *any* setting in which private and public information are related via a known optimization problem; extension of our method to other settings also remains as important future work.

Acknowledgments

This work is supported by the Department of Energy's Computational Science Graduate Fellowship under grant number DE-FG02-97ER25308.

References

[1] Frank A Wolak. Measuring Unilateral Market Power in Wholesale Electricity Markets: The California Market, 1998-2000. *American Economic Review*, 93(2):425–430, 2003.

[2] David Watts. Security and vulnerability in electric power systems. In *35th North American power symposium*, volume 2, pages 559–566, 2003.

[3] Ye Yuan, Omid Ardakanian, Steven Low, and Claire Tomlin. On the Inverse Power Flow Problem. *arXiv preprint arXiv:1610.06631*, 2016.

[4] James Anderson, Fengyu Zhou, and Steven H Low. Disaggregation for Networked Power Systems. In *2018 Power Systems Computation Conference (PSCC)*, pages 1–7. IEEE, 2018.

[5] Brandon Amos and J Zico Kolter. OptNet: Differentiable Optimization as a Layer in Neural Networks. In *International Conference on Machine Learning*, pages 136–145, 2017.

[6] Josip Djolonga and Andreas Krause. Differentiable Learning of Submodular Models. In *Advances in Neural Information Processing Systems*, pages 1013–1023, 2017.

[7] Allen J. Wood, Bruce F. Wollenberg, and Gerald B. Sheblé. *Optimal Power Flow*, chapter 8, pages 350–402. John Wiley & Sons, 2014.

[8] Paul T Boggs and Jon W Tolle. Sequential Quadratic Programming. *Acta numerica*, 4:1–51, 1995.

[9] Grid Optimization (GO) Competition. Datasets. `https://gocompetition.energy.gov/content/datasets`, 2018.

A Details on computing gradients through ACOPF

To compute the gradients $\nabla_\theta \ell((p_g^{(i)}, \lambda^{(i)}), (\hat{p_g}^{(i)}, \hat{\lambda}^{(i)}))$ for each $\theta \in \{\hat{c}, \hat{Y}\}$, we must take the gradient through the solutions to the ACOPF optimization problem. Specifically, we must compute the terms

$$\frac{\partial \ell}{\partial \theta} = \frac{\partial \ell}{\partial \hat{p_g}(\theta)} \frac{\partial \hat{p_g}(\theta)}{\partial \theta} + \frac{\partial \ell}{\partial \hat{\lambda}(\theta)} \frac{\partial \hat{\lambda}(\theta)}{\partial \theta}, \tag{A.1}$$

where $\frac{\partial \hat{p_g}(\theta)}{\partial \theta}$ and $\frac{\partial \hat{\lambda}(\theta)}{\partial \theta}$ are the Jacobians of optimal primal and dual variables, respectively, in problem (1), with respect to our parameter estimate θ (and where we denote the dependence of $\hat{p_g}$ and $\hat{\lambda}$ on each θ here explicitly). To compute these Jacobians at the last sequential quadratic programming iterate, we use the method described in [5], implicitly differentiating through the KKT optimality conditions of (2) to obtain linear equations we can solve to obtain the required gradients:

$$\begin{bmatrix} \operatorname{diag}(c_q) & G^T & \tilde{A}^T \\ \operatorname{diag}(\nu^\star) G & \operatorname{diag}(Gz^\star - h) & 0 \\ \tilde{A} & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial z^\star}{\partial \theta} \\ \frac{\partial \nu^\star}{\partial \theta} \\ \frac{\partial \tilde{\kappa}^\star}{\partial \theta} \end{bmatrix} = \begin{bmatrix} -\frac{\partial \operatorname{diag}(c_q)}{\partial \theta} z^\star - \frac{\partial c_a}{\partial \theta} - \frac{\partial G^T}{\partial \theta} \nu^\star - \frac{\partial \tilde{A}^T}{\partial \theta} \tilde{\kappa}^\star \\ -\operatorname{diag}(\nu^\star) \frac{\partial G}{\partial \theta} z^\star + \operatorname{diag}(\nu^\star) \frac{\partial h}{\partial \theta} \\ -\frac{\partial \tilde{A}}{\partial \theta} z^\star + \frac{\partial \tilde{b}}{\partial \theta} \end{bmatrix}, \tag{A.2}$$

where ν are the dual variables on the linear inequality constraints, and $\tilde{\kappa} = \begin{bmatrix} \kappa^T & \lambda^T \end{bmatrix}^T$ contains the dual variables κ on the original linear inequality constraints and λ on the linearized power flow constraint in (1). Here, we note that $\tilde{A} = \begin{bmatrix} A^T & J(z^\star)^T \end{bmatrix}^T$ and $\tilde{b} = \begin{bmatrix} b^T & k(z^\star)^T \end{bmatrix}^T$. While this equation may look complex, fundamentally, the left side of this equation contains the generalized Jacobian of the KKT optimality conditions of our convex problem, and the terms on the right side are the gradients of optimization problem parameters.

In practice, we solve a slightly different set of equations to efficiently compute these gradients in the context of a neural network, similar to the method described in [5]. In particular, we modify Equation (7) of [5] to incorporate the gradients of the loss with respect to both the optimal primal variable *and* the optimal dual variables on the equality constraints as

$$\begin{bmatrix} d_z \\ d_\nu \\ d_{\tilde{\kappa}} \end{bmatrix} = \begin{bmatrix} \operatorname{diag}(c_q) & G^T \operatorname{diag}(\nu^\star) & \tilde{A}^T \\ G & \operatorname{diag}(Gz^\star - h) & 0 \\ \tilde{A} & 0 & 0 \end{bmatrix}^{-1} \begin{bmatrix} \left(\frac{\partial \ell}{\partial z^\star}\right)^T \\ 0 \\ \left(\frac{\partial \ell}{\partial \tilde{\kappa}^\star}\right)^T \end{bmatrix}. \tag{A.3}$$

We then use the resultant values of d_z, d_ν, and $d_{\tilde{\kappa}}$ in the rest of the computations presented in [5].

B Details on the 14-node test case

We test our algorithm on a modified version of the IEEE 14-bus test case[1] with three generators located at nodes 1, 2, and 8, respectively. A schematic of this system is shown in Figure B.1. The power generation costs for each generator are $f_1(p_{g_1}) = 2p_{g_1}^2 + 5p_{g_1}$, $f_2(p_{g_2}) = 4p_{g_2}^2 + 2p_{g_2}$, and $f_8(p_{g_8}) = 5p_{g_8}^2 + 1p_{g_8}$, and the primitive admittance matrix parameters used to construct the admittance matrix can be found at the link in Footnote 1.

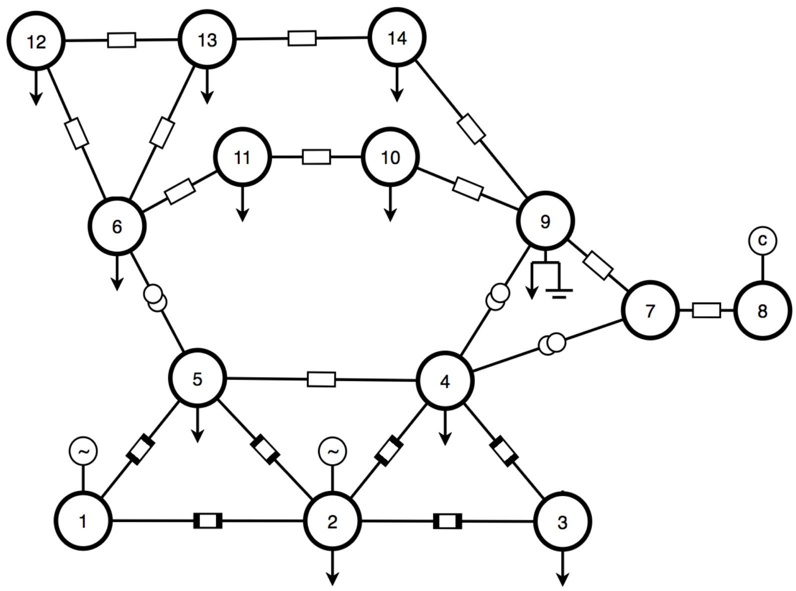

Figure B.1: The 14-node system on which we run our experiments.

[1] https://www.cs.cmu.edu/~zkolter/course/15-884/assignments.html