

# PACR: Parameter-Optimized AC Power Flow Restoration for AC Feasible DCOPF Dispatch

Michael A. Boateng, *Student Member, IEEE*, Russell Bent, *Senior Member, IEEE*,  
Sidhant Misra, *Senior Member, IEEE*, Parikshit Pareek, *Member, IEEE*,  
Pascal Van Hentenryck, *Senior Member, IEEE*, and Daniel K. Molzahn, *Senior Member, IEEE*

**Abstract**—The DC optimal power flow is widely used in power system operations because of its computational efficiency and scalability. However, DC dispatches are not guaranteed to satisfy the nonlinear AC power-flow equations or associated operational limits. This paper develops a parameterized, differentiable AC power-flow restoration method for mapping DC dispatches to AC-consistent operating points. The method incorporates distributed slack for active-power balancing and PV/PQ switching for reactive-power regulation, both implemented using smooth differentiable surrogates with tunable parameters, including slack participation factors, voltage setpoints, and regulation steepness. These parameters are trained offline by differentiating through the AC restoration equations using the implicit function theorem. Once trained, the optimized parameters are fixed and used directly during AC power-flow recovery from DC dispatches. The approach is evaluated on IEEE, ACTIVSg, and PEGASE test systems using setpoints computed by standard DC optimal power flow. Results show that the optimized restoration method improves AC feasibility recovery across various systems relative to conventional single-slack AC power-flow recovery. On the 9,241-bus case, the optimized method improves cost difference by 80% relative to the conventional recovery baseline and improves solving time relative to ACOPT by 75%.

**Index Terms**—DCOPF, ACPF, Learnable Parameters, Differentiable Optimization, Supervised Learning, Feasibility Restoration.

## NOMENCLATURE

### A. Parameters:

$\mathcal{N}$	Set of buses; $\mathcal{N} = \{1, \dots, N\}$
$\mathcal{E}$	Set of lines $(i, j)$ ; reverse direction $(j, i)$
$\mathcal{G}, \mathcal{L}$	Set of generators and loads; $\mathcal{G}, \mathcal{L} \subseteq \mathcal{N}$
$\mathbf{r}, \mathbf{b}, \mathbf{g}$	Line resistance, susceptance, conductance
$\mathbf{Y}$	Complex branch line admittance
$\Phi$	Line-Bus PTDF matrix, size $\mathcal{E} \times \mathcal{N}$
$S_{tx}^{\max}$	Apparent power transformer limit
$I_{line}^{\max}$	Current flow line limit
$\mathbf{p}_g^{\min}, \mathbf{p}_g^{\max}$	Active power generation limits
$\mathbf{q}_g^{\min}, \mathbf{q}_g^{\max}$	Reactive power generation limits
$\mathbf{v}^{\min}, \mathbf{v}^{\max}$	Voltage magnitude limits
$\epsilon$	Numerical convergence tolerance

### B. Variables:

$\mathbf{p}_g, \mathbf{q}_g$	Active and reactive power generation
$\mathbf{s}_g$	Complex power generation; $\mathbf{p}_g + \mathbf{j} \cdot \mathbf{q}_g$
$\mathbf{p}_d, \mathbf{q}_d$	Active and reactive power demand
$\theta^{\text{dc}}, \theta^{\text{ac}}$	Voltage angles from DC and AC models
$\ell$	Total active-power losses across network

## I. INTRODUCTION

THE DC optimal power flow (DCOPF) problem is a core computational tool in power system operations, markets, and planning due to its tractability, scalability, and economic alignment with time-critical decision-making [1]. The DCOPF is valuable in settings requiring repeated solves across large scenario sets, fast contingency screening, or integration within learning loops [2]–[10]. However, DCOPF solutions often violate the nonlinear AC power flow equations and/or operational constraints such as voltage limits and reactive power capabilities when these DCOPF setpoints are evaluated with an AC power flow (ACPF) [1], [11]–[13]. To address this challenge, this paper develops a structured learning approach for *AC feasibility restoration*.

In standard ACPF restoration of DCOPF dispatches, convergence and solve-time performance are strongly influenced by how active- and reactive-power infeasibilities are handled. Active-power imbalance is often resolved using a single-slack-bus formulation, but such formulations can struggle to recover AC feasibility from DCOPF approximations, often leading to constraint violations or infeasible restored states [11], [12], [14]. On the reactive-power side, iterative solvers typically rely on discrete PV/PQ switching, which can incur excessive outer-loop recalculations when generators reach reactive limits, thereby increasing computational cost and creating numerical convergence difficulties [15], [16].

Recent work has shown that both distributed slack and improved reactive-power adjustments can substantially strengthen AC restoration. In particular, our earlier work introduced a structured DCOPF→ACPF restoration pipeline that incorporates distributed slack and PV/PQ switching directly within the AC restoration step, and demonstrated that doing so materially improves the recovery of AC-feasible operating points from DC solutions [14]. More broadly, distributed-slack formulations spread active-power imbalance across multiple generators, helping to avoid individual generators from exceeding their limits during restoration [17], [18], while homotopy methods and smoothing of reactive-power curves can reduce Newton-Raphson iterations by mitigating the repeated recalculations caused by discrete PV/PQ switching [16], [19]–[21].

This paper builds directly on these structured restoration frameworks. Rather than introducing new restoration policies, it studies how the distributed-slack and PV/PQ policies can be parameterized, smoothed, and learned to yield an even more effective approach. This is important because the quality of the

TABLE I: TUNABLE PARAMETERS IN THE PACR METHOD

Parameter (role)	Interpretation
$\pi$ (softplus steepness)	Smooth headroom map for slack weights ( $\tilde{h}_i$ )
$\phi$ (softmax temperature)	Smooth participation-factor map ( $\alpha_i$ ) from headroom
$\psi_s^\dagger$ (sigmoid steepness)	Smooth PV/PQ regulation / PV-PQ surrogate
$\psi_t$ (tolerance band)	Margin of reactive limits for stable switching
$v_i^{\text{SP}\dagger}$ (voltage setpoint)	Tunable reference used in the PV/PQ surrogate

<sup>†</sup> The learned quantities are the voltage setpoints  $\mathbf{v}^{\text{SP}}$  and softmax temperature  $\phi$  influencing participation factors. Steepness parameters  $\{\pi, \psi_s, \psi_t\}$  are selected from empirically stable intervals for robust Newton convergence.

### Model 1: DC Optimal Power Flow (DCOPF)

$$\min \sum_{i \in \mathcal{G}} (c_{2,i} p_{g,i}^2 + c_{1,i} p_{g,i} + c_{0,i}) \quad (\text{Cost Minimization}) \quad (1a)$$

$$\text{s.t. } p_{g,i} - p_{d,i} = \sum_{j \in \mathcal{E}} \vec{p}_{j,i} - \sum_{j \in \mathcal{E}} \vec{p}_{i,j} \quad \forall i \in \mathcal{N} \quad (\text{Balance Eqn.}) \quad (1b)$$

$$\vec{p}_{i,j} = b_{i,j} (\theta_i^{\text{dc}} - \theta_j^{\text{dc}}) \quad \forall (i,j) \in \mathcal{E} \quad (\text{DC Flow}) \quad (1c)$$

$$|\vec{p}_{i,j}| \leq p_{i,j}^{\text{max}} \quad \forall (i,j) \in \mathcal{E} \quad (\text{Line Limit}) \quad (1d)$$

$$\mathbf{p}_g^{\text{min}} \leq \mathbf{p}_g^{\text{dc}} \leq \mathbf{p}_g^{\text{max}} \quad (\text{Gen. Active Power}) \quad (1e)$$

### Model 2: Parameter-Optimized ACPF Restoration (PACR)

#### Power Injection Mismatches:

$$\Delta p_i = \underbrace{p_{g,i}^{\text{dc}} + \alpha_i \cdot \ell - p_{d,i}}_{\text{smooth distributed slack}} - \sum_{j \in \mathcal{N}} v_i v_j (g_{i,j} \cos \theta_{i,j} + b_{i,j} \sin \theta_{i,j}) \quad \forall i \in \mathcal{N} \quad (2a)$$

$$\Delta q_i = \underbrace{q_{g,i}^{\text{ac}}(v_i)}_{\text{smooth PV/PQ}} - q_{d,i} - \sum_{j \in \mathcal{N}} v_i v_j (g_{i,j} \sin \theta_{i,j} - b_{i,j} \cos \theta_{i,j}) \quad \forall i \in \mathcal{N} \quad (2b)$$

#### Smooth Distributed Slack (Participation)\*:

$$\tilde{h}_i = \frac{1}{\pi} \ln(1 + e^{\pi(p_{g,i}^{\text{max}} - p_{g,i}^{\text{dc}})}) - \frac{\ln 2}{\pi} \quad \forall i \in \mathcal{G} \quad (2c)$$

$$\alpha_i = \frac{e^{\phi \tilde{h}_i}}{\sum_{j \in \mathcal{G}} e^{\phi \tilde{h}_j}}, \quad \sum_{i \in \mathcal{G}} \alpha_i = 1 \quad \forall i \in \mathcal{G} \quad (2d)$$

#### Smooth PV/PQ Control\*:

$$\underbrace{q_{g,i}^{\text{max}} \leftarrow q_{g,i}^{\text{max}} - \psi_t, \quad q_{g,i}^{\text{min}} \leftarrow q_{g,i}^{\text{min}} + \psi_t}_{\text{updated reactive-power limits}}$$

$$q_{g,i}^{\text{ac}} = q_{g,i}^{\text{min}} + \frac{q_{g,i}^{\text{max}} - q_{g,i}^{\text{min}}}{1 + e^{\psi_s(v_i - v_i^{\text{SP}}) + \ln\left(\frac{q_{g,i}^{\text{max}} - q_{g,i}^{\text{SP}}}{q_{g,i}^{\text{SP}} - q_{g,i}^{\text{min}}}\right)}} \quad \forall i \in \mathcal{G} \quad (2e)$$

#### Augmented Jacobian System:

$$\underbrace{\begin{bmatrix} \frac{\partial \Delta \mathbf{p}}{\partial \theta} & \frac{\partial \Delta \mathbf{p}}{\partial \mathbf{v}} & \boldsymbol{\alpha} \\ \frac{\partial \Delta \mathbf{q}}{\partial \theta} & \frac{\partial \Delta \mathbf{q}}{\partial \mathbf{v}} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}^T & \mathbf{0} \end{bmatrix}}_{\mathbf{J}_{\text{ac}}} \begin{bmatrix} \Delta \boldsymbol{\theta} \\ \Delta \mathbf{v} \\ \Delta \ell \end{bmatrix} = - \begin{bmatrix} \Delta \mathbf{p} \\ \Delta \mathbf{q} \\ \ell \end{bmatrix} \quad (2f)$$

**Convergence:**  $\|\Delta \mathbf{p}\|_\infty, \|\Delta \mathbf{q}\|_\infty < \epsilon$ .

\* PACR by default uses smooth surrogates for distributed slack and PV/PQ control, and approaches the discrete logic of these controls [14], as smoothing is removed; see Fig. 2.

restoration outcome depends critically on parameter choices such as participation factors and voltage setpoints, and poor

choices can lead to slow convergence or infeasible solutions [6], [22], [23]. To address this challenge, this paper develops a supervised learning framework that optimizes restoration parameters offline using ACOF solutions as ground truth. Smoothing is introduced to make the restoration method differentiable and the parameter-learning more tractable. In summary, the paper's main contributions are:

1) Developing a parameter-optimized ACPF restoration (PACR) formulation with smooth approximations for distributed slack (softplus headroom, softmax participation factors) and reactive power regulation (sigmoid PV/PQ curves), replacing discrete logic used in standard formulations.

2) Designing an implicit-function theorem-based parameter learning approach for ACPF restoration, enabling gradient-based optimization of parameters for differentiable correction of DCOPF dispatches to AC feasible points.

3) Demonstrating the transferability of optimized restoration parameters from the smooth differentiable ACPF formulation to the discrete ACPF recovery framework, while showing that the smooth formulation provides the strongest overall trade-off between AC feasibility, dispatch accuracy, and computational efficiency relative to the prior discrete framework [14].

The remainder of the paper is organized as follows. Section II reveals the parameter-optimized AC restoration (PACR) method. Section III describes the DCOPF  $\rightarrow$  ACPF feasibility restoration pipeline. Section IV reports the numerical results, and Section V concludes and provides future research directions.

## II. PARAMETERIZING THE ACPF RESTORATION METHOD

This section introduces the parameterized DCOPF  $\rightarrow$  ACPF restoration pipeline used in PACR. As shown in Fig. 1, load inputs first define the DCOPF problem, whose solution provides active-power dispatch and angle initializations for AC restoration. A parameterized ACPF model then restores the DC operating point to an AC-feasible state through two internal regulation modules: distributed slack for active-power balancing and PV/PQ regulation for reactive-power control. The key idea is to parameterize these internal restoration policies, so that the correction step itself becomes more effective and differentiable. Figure 2 illustrates the corresponding smooth surrogates, and Table I summarizes the PACR parameters used.

We emphasize that all learned parameters are internal to the PACR method and are trained offline to improve the DCOPF  $\rightarrow$  ACPF correction. They do not modify operational AGC participation factors, AVR setpoints, or other controls in the actual system, nor do they model dynamic system response. The discussion below follows the two main restoration modules in turn: smooth distributed slack share for active-power balancing, and smooth PV/PQ switching for reactive-power and voltage control.

### A. Distributed Slack for Active Power Balancing

The first lever in PACR is the distributed-slack mechanism embedded in Model 2, which is initialized from the standard DCOPF setpoints obtained from Model 1. As also illustrated by the distributed-slack controls in the caption of Fig. 1, its role is to absorb the active-power imbalance that appears when the DCOPF dispatch is mapped into the nonlinear AC equations.

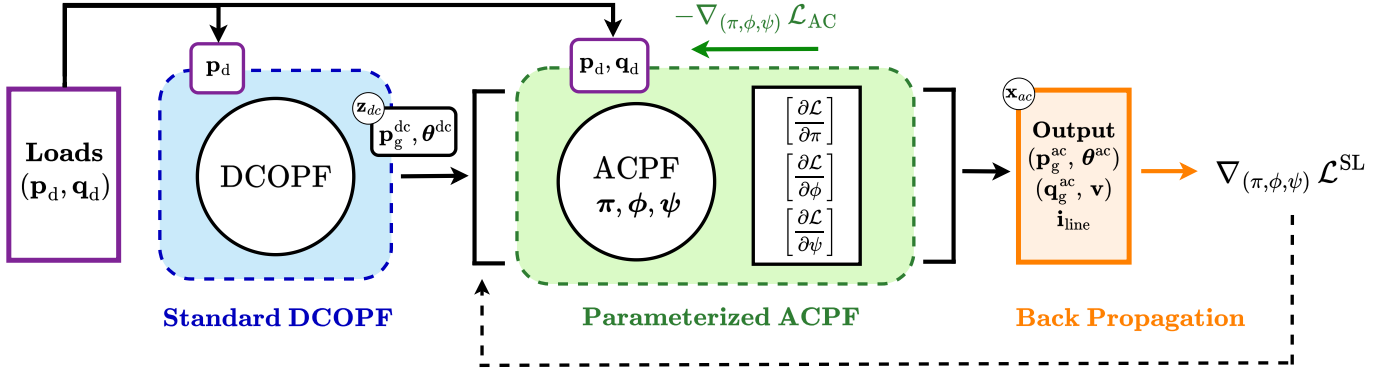


Fig. 1: **Load** inputs  $(\mathbf{p}_d, \mathbf{q}_d)$  feed a **DCOPF** ( $\text{DC}^{\text{BASE}}$ ) via  $\mathbf{p}_d$ , producing DC setpoints  $\mathbf{z}_{dc} = (\mathbf{p}_g^{\text{dc}}, \boldsymbol{\theta}^{\text{dc}})$ . These initialize the **ACPF** that maps  $(\mathbf{p}_d, \mathbf{q}_d, \mathbf{z}_{dc}) \mapsto \mathbf{x}_{ac} = \{(\mathbf{p}_g^{\text{ac}}, \boldsymbol{\theta}^{\text{ac}}), (\mathbf{q}_g^{\text{ac}}, \mathbf{v}), \mathbf{i}_{\text{line}}\}$ . The ACPF contains switchable *regulation controls*: 1) *Distributed slack control* with parameters  $\pi$  (softplus headroom steepness) and  $\phi$  (softmax participation temperature), and 2) *PV/PQ regulation control* with  $\psi = \{\psi_s, \psi_t, \mathbf{v}^{\text{sp}}\}$  (sigmoid steepness, tolerance, and voltage setpoint). Both *controls 1) & 2)* are *smooth (S)* implementations, thus yielding  $\text{DCAC}^{\text{SS}}$ , and replacing the discrete pipelines  $\text{DCAC}^{\text{DD}}$  [14]. **Outputs** are trained with supervised learning (SL), using ACOFP targets.

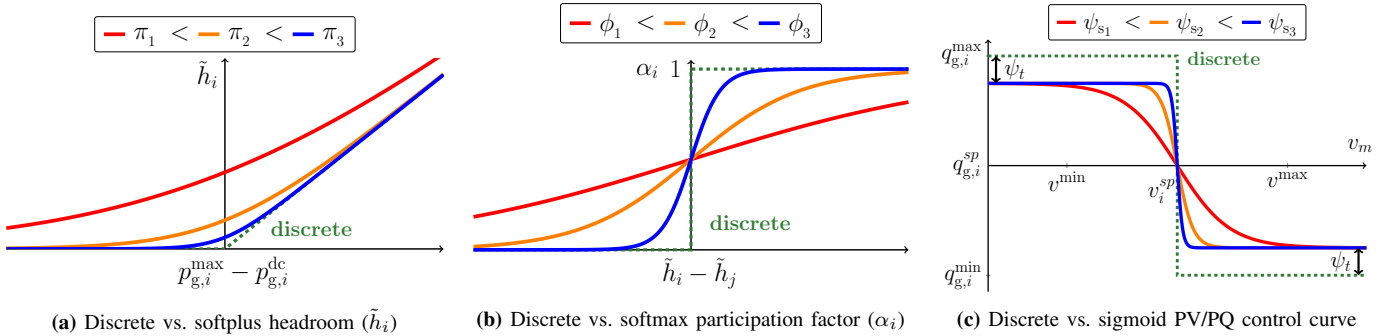


Fig. 2: Discrete vs. Smooth AC surrogates. (a) Softplus headroom for slack allocation, (b) softmax participation factors, and (c) differentiable sigmoid PV/PQ control. **Green** denotes discrete baselines [14] and **Red/Orange/Blue** denote increasing smoothness. The smoothness adjustments are tuned via  $(\pi, \phi, \psi_s)$  and shaped by  $v^{\text{sp}}$  (curve shift) and  $\psi_t$  (Q-limit tolerance before  $q_{g,i}^{\text{min}}/q_{g,i}^{\text{max}}$  limits engage).

Rather than assigning all imbalance to a single slack bus, the restoration distributes this correction across generators according to their available headroom [14], [17], [18]. Prior work shows that headroom-aware slack allocation is effective at reducing post-restoration active-power violations [14].

In the discrete setting, headroom and participation choices are assigned through non-smooth logic. Model 2 replaces this with a differentiable two-stage mapping. First, the softplus surrogate in Fig. 2a computes smooth generator headroom  $\tilde{h}_i$  through (2c). Second, the softmax mapping in Fig. 2b converts these values into normalized participation factors  $\alpha_i$  through (2d). The parameters  $\pi$  and  $\phi$  control the sharpness of these approximations and determine how closely the smooth model approaches the corresponding discrete rule. These participation factors enter directly into the active-power mismatch equations (2a) and the augmented Jacobian system (2f), allowing slack allocation and AC restoration to be solved simultaneously.

### B. PV/PQ Switching for Reactive Power and Voltage Control

The second lever in PACR is the reactive-power regulation mechanism in Model 2, again applied after the standard DCOPF initialization from Model 1. This part of the restoration replaces the conventional discrete PV/PQ switching rule used in previous AC power flow pipelines [14]. In the standard formulation, a generator remains in PV mode while it can regulate voltage

within its reactive limits, and switches to PQ mode once one of those limits is reached. Although physically meaningful, this rule introduces discontinuities into the restoration map, making the ACPF harder to solve and unsuitable for gradient-based training [16], [19], [20]. PACR replaces this discrete switching rule with a smooth surrogate.

Figure 2c shows the transition from discrete PV/PQ switching to a differentiable sigmoid-based PV/PQ control curve. Instead of abruptly changing modes at  $q_{g,i}^{\text{min}}$  or  $q_{g,i}^{\text{max}}$ , the smooth regulation in Model 2, given by (2e), creates a continuous relationship between voltage deviation and reactive-power response. The steepness parameter  $\psi_s$  controls how sharply the sigmoid approximates the discrete switch,  $\psi_t$  introduces a tolerance band near the reactive limits to improve numerical robustness, and the voltage setpoint  $v_i^{\text{sp}}$  shifts the curve to define the desired operating point. The logarithmic offset in (2e) ensures that when  $v_i = v_i^{\text{sp}}$ , the reactive injection equals the desired setpoint  $q_{g,i}^{\text{sp}}$ , as shown in Appendix A. Embedding this smooth PV/PQ relation into the reactive mismatch equations (2b) and the augmented Jacobian system (2f) yields a tunable restoration procedure compatible with Newton-based AC correction.

## III. AC RESTORATION LEARNING ALGORITHM

The discussion below focuses on the optimization of the DCOPF→ACPF restoration pipeline. In particular, we show

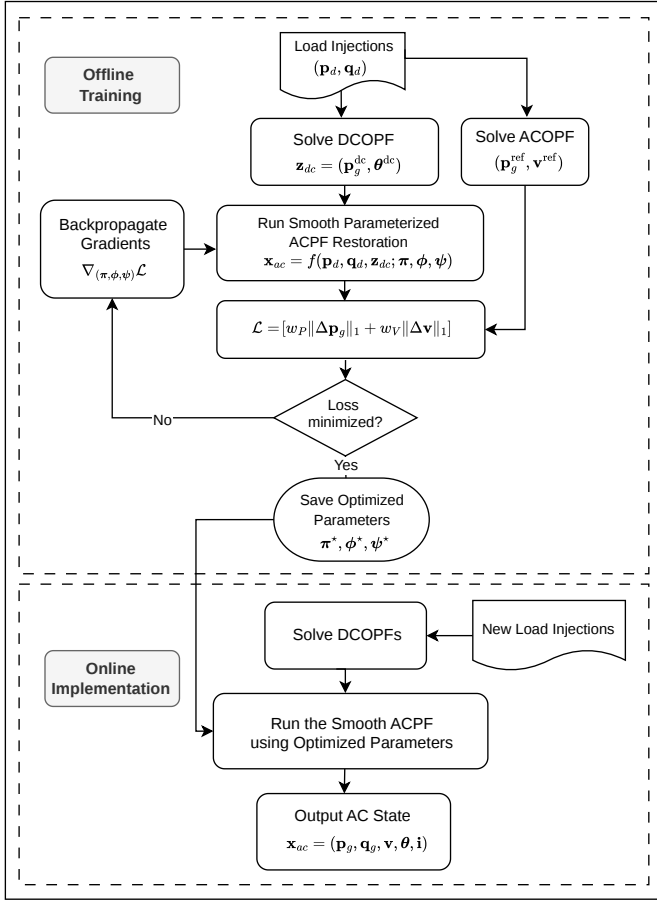


Fig. 3: Flowchart of the algorithm, illustrating the offline training and online implementation of the DCOPF→ACPF pipeline.

how the parameter-optimized ACPF restoration (PACR) model can be trained to improve the differentiable correction of DCOPF dispatches to AC-feasible operating points through smooth distributed slack and smooth reactive-power regulation. Fig. 3 summarizes the offline–online workflow, and Algorithm 1 outlines the corresponding parameter-optimization training procedure. The two main ingredients here are the sensitivity analysis required to differentiate through the converged augmented ACPF equations, and the supervised objective used to train PACR parameters against ACOF reference solutions.

#### A. Sensitivities of the DCOPF → ACPF Pipeline

The smooth PACR formulation enables gradient-based parameter optimization through the implicit function theorem (IFT). Starting from a DCOPF initialization  $\mathbf{z}_{dc}$  and reactive demand  $\mathbf{q}_d$ , the augmented ACPF system is solved with Newton–Raphson using smooth distributed slack and smooth PV/PQ switching. At convergence, the restored AC state:

$$\mathbf{x}_{ac} = (\mathbf{p}_g^{\text{ac}}, \boldsymbol{\theta}^{\text{ac}}, \mathbf{q}_g^{\text{ac}}, \mathbf{v})$$

satisfies the following:

$$\mathbf{g}(\mathbf{x}_{ac}; \boldsymbol{\pi}, \boldsymbol{\phi}, \boldsymbol{\psi}) = \mathbf{0},$$

where  $\boldsymbol{\pi}$  governs the smooth headroom map,  $\boldsymbol{\phi}$  controls participation factors, and  $\boldsymbol{\psi} = \{\boldsymbol{\psi}_s, \boldsymbol{\psi}_t, \mathbf{v}^{\text{SP}}\}$  defines the smooth PV/PQ regulation, and includes voltage setpoints.

Because the converged solution is defined implicitly by the augmented ACPF equations, its parameter sensitivities can be computed from the converged Jacobian  $\mathbf{J}_{ac}$  in (2f). For any parameter block  $\boldsymbol{\xi} \in \{\boldsymbol{\pi}, \boldsymbol{\phi}, \boldsymbol{\psi}\}$ ,

$$\frac{\partial \mathbf{x}_{ac}}{\partial \boldsymbol{\xi}} = -\mathbf{J}_{ac}^{-1} \frac{\partial \mathbf{g}}{\partial \boldsymbol{\xi}}. \quad (3)$$

This reuses the same augmented Jacobian needed in the forward solve and avoids explicit unrolling of all Newton iterations. Additional derivations are given in Appendix B.

#### B. Optimizing the Parameterized ACPF Restoration Method

Parameter optimization requires careful loss function selection to balance AC feasibility, DC consistency, and computational efficiency. The DCAC<sup>SS</sup> formulation serves as the differentiable pipeline for computing optimal parameters  $\boldsymbol{\pi}^*, \boldsymbol{\phi}^*, \boldsymbol{\psi}^*$  through Algorithm 1. Supervised learning is used for training, with AC references  $(\mathbf{p}_g^{\text{ref}}, \mathbf{v}^{\text{ref}})$  obtained from ACOF solutions. The full-batch loss over  $\mathcal{D}$  is:

$$\mathcal{L} = \frac{1}{|\mathcal{D}|} \sum_{\mathcal{D}} [w_P \|\mathbf{p}_g^{\text{ac}} - \mathbf{p}_g^{\text{ref}}\|_1 + w_V \|\mathbf{v} - \mathbf{v}^{\text{ref}}\|_1]. \quad (4)$$

The gradient-computation step in Algorithm 1 combines the loss derivatives with the sensitivity analysis from (3), enabling efficient parameter updates through Adam. The loss function (4) is augmented with a reference regularization term  $w_{\text{reg}} (\|\boldsymbol{\psi} - \boldsymbol{\psi}_{\text{ref}}\|_2^2 + \|\boldsymbol{\pi} - \boldsymbol{\pi}_{\text{ref}}\|_2^2 + \|\boldsymbol{\phi} - \boldsymbol{\phi}_{\text{ref}}\|_2^2)$  to prevent overfitting and improve parameter stability. This term discourages large deviations from nominal and stable smoothness values rather than driving the parameters toward zero. For evaluation, DCAC<sup>BASE</sup> denotes the conventional single-slack restoration baseline without regulation or parameter optimization, while this paper’s DCAC<sup>SS</sup> and the DCAC<sup>DD</sup> [14] incorporate smooth and discrete distributed-slack/PV–PQ regulation, respectively. Each method is tested with either *unoptimized parameters (init)* or *optimized parameters (opt)*, where the latter are learned through Algorithm 1.

## IV. NUMERICAL EXPERIMENTS AND DISCUSSIONS

The DCOPF→ACPF pipeline described in Section III-B is evaluated through offline parameter optimization using supervised learning. This section presents the computational setup and results. Experiments were conducted on an IEEE test case {ieee\_118}, ACTIVSg networks {ACTIVSg\_200, ACTIVSg\_500}, and PEGASE networks {pegase\_1354, pegase\_2869, pegase\_9241}. For each case, 5,000 perturbed loading scenarios were generated, yielding 30,000 total scenarios across the six systems; these were split into 4,000 training and 1,000 test samples per case. Computations were carried out on the Darwin high-performance computing system at Los Alamos National Laboratory, using single 24-core compute nodes equipped with 32 GB RAM. ACOF and DCOPF were modeled using `PowerModels.jl` [24] and solved using `Ipopt` [25], while the ACPF implementation was developed using `pandapower.py` [26]. ACOF was used as labeled targets for supervised learning. Active power demands ( $\mathbf{p}_d$ ) were perturbed using bounded Gaussian

**Algorithm 1: Adam Parameter Training using PACR<sup>1</sup>**


---

**Input:** Training set  $\mathcal{D} = \{(\mathbf{p}_d, \mathbf{q}_d, \mathbf{p}_g^{\text{ref}}, \mathbf{v}^{\text{ref}})\}$   
Fixed DCOPF map  $\text{DC}(\cdot)$  producing  $\mathbf{z}_{\text{dc}} = (\mathbf{p}_g^{\text{dc}}, \boldsymbol{\theta}^{\text{dc}})$   
Initial parameters  $\boldsymbol{\pi}_1 = \{\pi\}$ ,  $\phi_1 = \{\phi\}$ ,  $\boldsymbol{\psi}_1 = \{\psi_s, \psi_t, \mathbf{v}^{\text{SP}}\}$   
Voltage limits  $\mathbf{v}^{\text{min}}, \mathbf{v}^{\text{max}}$ , gen. limits  $\mathbf{p}_g^{\text{min}}, \mathbf{p}_g^{\text{max}}$   
**Output:** Trained parameters  $\boldsymbol{\pi}^*, \phi^*, \boldsymbol{\psi}^*$   
**Hyperparameters:** Max iterations  $N_{\text{iter}}$ ; tolerance  $\epsilon$ ; Adam  
stepsize  $\eta$ ; moments  $\beta_1, \beta_2$ ; weights  
 $w_P, w_V, w_{\text{reg}}$

- 1 Initialize Adam moments  $\mathbf{m}_\xi, \mathbf{s}_\xi \leftarrow \mathbf{0}$  for  $\xi \in \{\boldsymbol{\pi}, \phi, \boldsymbol{\psi}\}$
- 2 **for**  $j = 1, \dots, N_{\text{iter}}$  **do**
- 3   Initialize full-batch tracking loss  $\mathcal{L} \leftarrow 0$   
   // 1) Full-Batch Forward Pass
- 4   **foreach**  $(\mathbf{p}_d, \mathbf{q}_d, \mathbf{p}_g^{\text{ref}}, \mathbf{v}^{\text{ref}}) \in \mathcal{D}$  **do**
- 5      $(\mathbf{p}_g^{\text{dc}}, \boldsymbol{\theta}^{\text{dc}}) \leftarrow \text{DCOPF}$
- 6      $(\mathbf{p}_g^{\text{ac}}, \mathbf{q}_g^{\text{ac}}, \mathbf{v}, \mathbf{i}) \leftarrow \text{ACPF}_{\pi_j, \phi_j, \boldsymbol{\psi}_j}(\mathbf{p}_d, \mathbf{q}_d, \mathbf{p}_g^{\text{dc}}, \boldsymbol{\theta}^{\text{dc}})$
- 7      $\mathcal{L}^{\text{sc}} \leftarrow w_P \|\mathbf{p}_g^{\text{ac}} - \mathbf{p}_g^{\text{ref}}\|_1 + w_V \|\mathbf{v} - \mathbf{v}^{\text{ref}}\|_1$
- 8      $\mathcal{L} \leftarrow \mathcal{L} + \mathcal{L}^{\text{sc}}$  // Scenario loss
- // 2) Objective and Gradient Compute
- 9    $\mathcal{J} \leftarrow \frac{1}{|\mathcal{D}|} \mathcal{L} +$   
    $w_{\text{reg}} (\|\boldsymbol{\pi}_j - \boldsymbol{\pi}_{\text{ref}}\|_2^2 + \|\phi_j - \phi_{\text{ref}}\|_2^2 + \|\boldsymbol{\psi}_j - \boldsymbol{\psi}_{\text{ref}}\|_2^2)$   
   Compute sensitivities and full-batch gradients of  $\mathcal{J}$ :
- 10    $\mathbf{g}_\pi = \nabla_{\boldsymbol{\pi}} \mathcal{J}$ ,  $\mathbf{g}_\phi = \nabla_{\phi} \mathcal{J}$ ,  $\mathbf{g}_\psi = \nabla_{\boldsymbol{\psi}} \mathcal{J}$
- 11   **if**  $\max\{\|\mathbf{g}_\pi\|_\infty, \|\mathbf{g}_\phi\|_\infty, \|\mathbf{g}_\psi\|_\infty\} < \epsilon$  **then**
- 12     **break** // Convergence achieved
- 13   // 3) Adam Parameter Update
- 14   **foreach**  $\xi \in \{\boldsymbol{\pi}, \phi, \boldsymbol{\psi}\}$  **do**
- 15      $\mathbf{m}_\xi \leftarrow \beta_1 \mathbf{m}_\xi + (1 - \beta_1) \mathbf{g}_\xi$
- 16      $\mathbf{s}_\xi \leftarrow \beta_2 \mathbf{s}_\xi + (1 - \beta_2) (\mathbf{g}_\xi \odot \mathbf{g}_\xi)$
- 17      $\hat{\mathbf{m}}_\xi \leftarrow \mathbf{m}_\xi / (1 - \beta_1^j)$ ,  $\hat{\mathbf{s}}_\xi \leftarrow \mathbf{s}_\xi / (1 - \beta_2^j)$
- 18      $\xi_{j+1} \leftarrow \xi_j - \eta \frac{\hat{\mathbf{m}}_\xi}{\sqrt{\hat{\mathbf{s}}_\xi + \epsilon_{\text{adam}}}}$
- 19 **return**  $\boldsymbol{\pi}^* = \boldsymbol{\pi}_j$ ,  $\phi^* = \phi_j$ ,  $\boldsymbol{\psi}^* = \boldsymbol{\psi}_j$
- 20 The optimized PACR parameters define  $\text{DCAC}^{\text{SSopt}}$ .

---

multiplicative noise, where the scaling factors were sampled from  $\mathcal{N}(1.0, (15\%)^2)$ . Reactive power demands ( $\mathbf{q}_d$ ) were then recomputed from randomly sampled power factors in  $[0.95, 1.0]$ . ACOPF scenarios that failed to converge were excluded from the training dataset. The convergence rate exceeded 90% across all ACPF variants with  $\text{DCAC}^{\text{BASE}}$  as setpoints. For computational efficiency, only voltage setpoints  $\mathbf{v}^{\text{SP}}$  for PV/PQ regulation and the softmax temperature  $\phi$  for distributed-slack participation were optimized. The remaining smooth-control parameters  $\{\pi, \psi_s, \psi_t\}$  were fixed to reference smoothing values selected from preliminary sweeps to ensure robust Newton-Raphson convergence. These reference values define moderately smooth approximations of the discrete controls, rather than serving as optimization variables. For example, appropriate PV/PQ sigmoid steepness selection reduced ACPF iterations from 35 to 4 on the `pegase_1354` case.

Parameter optimization was performed using Adam from `scipy.optimize.minimize` with box constraints:  $\phi \in [50, 900]$  for the participation temperature and  $\mathbf{v}^{\text{SP}} \in [0.95, 1.05]$  for voltage setpoints. The regularization term penalizes deviations from the stable reference smoothing values, rather than driving the parameters toward zero. This discourages unnecessarily steep or ill-conditioned smooth approximations

while allowing the optimized parameters to improve AC feasibility restoration. Loss function weights were used to select primary importance of generation dispatch and voltage profiles. Gradients were computed using ACPF derivatives. To maximize computational efficiency during optimization, the ACPF solver leverages sparse Jacobian construction with `scipy.sparse` CSC matrices, executing fast linear system updates via `scipy.sparse.linalg.spsolve`. This interface utilizes SuperLU, for large, nonsymmetric sparse systems via LU decomposition. Optimized parameters were deployed in both  $\text{DCAC}^{\text{SSopt}}$  and  $\text{DCAC}^{\text{DDopt}}$  variants to verify transferability of the training method. These optimized pipelines were compared against *initial parameter values*: voltage setpoints initialized to 1.0 p.u. and the initial participation factors corresponding to a single-slack allocation, with participation factor 1.0 assigned to the designated slack bus and 0.0 assigned to other buses. Newton-Raphson convergence tolerances was set to  $10^{-6}$  p.u. Mean absolute error (MAE) and cost difference (CD) are accuracies relative to ACOPF ( $|\mathcal{G}|$  is the number of generators):

$$\text{MAE} = \frac{1}{|\mathcal{G}|} \|\mathbf{p}_g - \mathbf{p}_g^{\text{ACOPF}}\|_1. \quad (5)$$

$$\text{CD} = \frac{|\text{Cost}^{\text{DCAC}} - \text{Cost}^{\text{ACOPF}}|}{\text{Cost}^{\text{ACOPF}}} \cdot 100. \quad (6)$$

#### A. AC Feasibility Restoration Assessment: Initialized versus Optimized Smooth Pipelines ( $\text{DCAC}^{\text{SSinit}}$ vs. $\text{DCAC}^{\text{SSopt}}$ ):

This subsection evaluates the effect of parameter optimization within the smooth AC restoration pipeline. Specifically, the initialized smooth pipeline,  $\text{DCAC}^{\text{SSinit}}$  is compared against the optimized smooth pipeline,  $\text{DCAC}^{\text{SSopt}}$ , with  $\text{DCAC}^{\text{BASE}}$  included as a conventional single-slack reference. The goal is to assess whether the learned smooth distributed-slack and PV/PQ regulation parameters improve AC feasibility restoration from the lossless initialized pipeline, while maintaining favorable cost, dispatch-error, and computational performance.

1) *Constraint Violation Table Analysis:* Table II compares active-power, reactive-power, voltage-limit, and line-limit violations for  $\text{DCAC}^{\text{BASE}}$ ,  $\text{DCAC}^{\text{SSinit}}$ , and  $\text{DCAC}^{\text{SSopt}}$  across 1,000 test samples. The single-slack baseline frequently leaves residual AC infeasibilities, especially reactive-power violations, with average violation rates of 18.46%, 15.69%, and 14.74% on `pegase_1354`, `pegase_2869`, and `pegase_9241`, respectively. In contrast, both smooth pipelines eliminate active- and reactive-power violations across all tested systems. The optimized smooth pipeline further improves the remaining feasibility behavior relative to the initialized smooth pipeline. It removes the small line-limit violations observed for  $\text{DCAC}^{\text{SSinit}}$  on `ACTIVSg_500` and reduces the average voltage-violation rates from 0.21% to 0.03% on `pegase_2869` and from 3.75% to 2.61% on `pegase_9241`. Thus, parameter optimization preserves major feasibility gains of the initialized smooth pipeline while further reducing residual voltage and line-limit violations in the more challenging cases.

2) *Evaluation-Loss and Performance Table Analysis:* Table IV reports the evaluation loss (4) for the baseline and

TABLE II: AVERAGE CONSTRAINT VIOLATION RATES AND MAXIMUM VIOLATION MAGNITUDES FOR  $DCAC^{BASE}$ ,  $DCAC^{SS_{INIT}}$ , AND  $DCAC^{SS_{OPT}}$  ACROSS 1,000 LOAD SAMPLES (USING  $DC_{BASE}$  SETPOINTS)

Test case	Method	Active Power (p.u.)		Reactive Power (p.u.)		Voltage Limit (p.u.)		Line Limit (%)	
		% viol.	Max	% viol.	Max	% viol.	Max	% viol.	Max
ieee_118	$DCAC^{BASE}$	1.61	0.01	5.56	2.93	0.05	0.01	0.00	0.20
	$DCAC^{SS_{INIT}}$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	$DCAC^{SS_{OPT}}$	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>
ACTIVSg_200	$DCAC^{BASE}$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	$DCAC^{SS_{INIT}}$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	$DCAC^{SS_{OPT}}$	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>
ACTIVSg_500	$DCAC^{BASE}$	0.00	0.00	8.03	0.06	0.00	0.00	0.00	0.00
	$DCAC^{SS_{INIT}}$	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.71
	$DCAC^{SS_{OPT}}$	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>
pegase_1354	$DCAC^{BASE}$	0.46	0.02	18.46	18.28	0.00	0.00	0.00	0.00
	$DCAC^{SS_{INIT}}$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	$DCAC^{SS_{OPT}}$	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>
pegase_2869	$DCAC^{BASE}$	0.51	2.63	15.69	13.73	0.00	0.00	0.00	0.00
	$DCAC^{SS_{INIT}}$	0.00	0.00	0.00	0.00	0.21	0.02	0.00	0.00
	$DCAC^{SS_{OPT}}$	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.03</b>	<b>0.01</b>	<b>0.00</b>	<b>0.00</b>
pegase_9241	$DCAC^{BASE}$	2.31	14.66	14.74	10.71	17.23	0.07	2.31	44.73
	$DCAC^{SS_{INIT}}$	0.00	0.00	0.00	0.00	3.75	0.02	0.00	0.00
	$DCAC^{SS_{OPT}}$	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>2.61</b>	<b>0.02</b>	<b>0.10</b>	<b>0.05</b>

TABLE III: PERFORMANCE TABLE FOR  $DCAC^{BASE}$ ,  $DCAC^{SS_{INIT}}$ , AND  $DCAC^{SS_{OPT}}$  ACROSS 1,000 SAMPLES ( $DC_{BASE}$  SETPOINTS)

Test case	Method	Cost Diff. (%)		MAE* (p.u.)		Iter. Count		Solve Time (s)	
		val.	%Improv.	val.	%Improv.	val.	%Improv.	val.	%Improv.
ieee_118	$DCAC^{BASE}$	1.488	+ <b>33.8</b>	0.025	+ <b>8.0</b>	4.0	—	0.073	—
	$DCAC^{SS_{INIT}}$	0.993	+ <b>0.8</b>	0.023	0.0	5.0	-8.0	0.269	+ <b>24.2</b>
	$DCAC^{SS_{OPT}}$	0.985	—	0.023	—	5.4	—	0.204	—
ACTIVSg_200	$DCAC^{BASE}$	0.035	—	0.003	0.0	2.0	—	0.065	—
	$DCAC^{SS_{INIT}}$	0.524	-13.9	0.003	0.0	4.0	+ <b>25.0</b>	0.258	+ <b>46.5</b>
	$DCAC^{SS_{OPT}}$	0.597	—	0.003	—	3.0	—	0.138	—
ACTIVSg_500	$DCAC^{BASE}$	1.383	+ <b>70.1</b>	0.010	0.0	3.0	—	0.113	—
	$DCAC^{SS_{INIT}}$	2.196	+ <b>81.2</b>	0.010	0.0	3.4	+ <b>5.9</b>	0.266	+ <b>15.4</b>
	$DCAC^{SS_{OPT}}$	0.413	—	0.010	—	3.2	—	0.225	—
pegase_1354	$DCAC^{BASE}$	0.394	+ <b>88.8</b>	0.050	+ <b>20.0</b>	4.0	—	0.361	—
	$DCAC^{SS_{INIT}}$	0.333	+ <b>86.8</b>	0.048	+ <b>16.7</b>	4.8	+ <b>10.4</b>	1.102	+ <b>19.9</b>
	$DCAC^{SS_{OPT}}$	0.044	—	0.040	—	4.3	—	0.883	—
pegase_2869	$DCAC^{BASE}$	0.458	+ <b>95.4</b>	0.042	+ <b>28.6</b>	4.0	—	0.915	—
	$DCAC^{SS_{INIT}}$	0.387	+ <b>94.6</b>	0.041	+ <b>26.8</b>	5.3	+ <b>17.0</b>	2.324	+ <b>19.8</b>
	$DCAC^{SS_{OPT}}$	0.021	—	0.030	—	4.4	—	1.865	—
pegase_9241	$DCAC^{BASE}$	0.631	+ <b>80.3</b>	0.038	+ <b>34.2</b>	3.0	—	3.389	—
	$DCAC^{SS_{INIT}}$	0.510	+ <b>75.7</b>	0.035	+ <b>28.6</b>	7.1	+ <b>15.5</b>	10.516	+ <b>25.3</b>
	$DCAC^{SS_{OPT}}$	0.124	—	0.025	—	6.0	—	7.860	—

The % Improv. report values relative to  $DCAC^{SS_{OPT}}$  for each metric.  $DCAC^{SS_{OPT}}$  uses optimized values for smooth distributed slack and smooth PV/PQ switching. The initialized smooth pipeline,  $DCAC^{SS_{INIT}}$ , uses initial parameter values: voltage setpoints initialized to 1.0 p.u., with participation factor 1.0 assigned to the slack bus and 0.0 to all other buses.

TABLE IV: TEST LOSS VALUES ACROSS 1,000 SAMPLES COMPARING  $DCAC^{BASE}$ ,  $DCAC^{SS_{INIT}}$  AND  $DCAC^{SS_{OPT}}$  (USING  $DC_{BASE}$  SETPOINTS)

Test Case	Evaluation Loss $\mathcal{L}$		
	* $DCAC^{BASE}$	* $DCAC^{SS_{INIT}}$	$DCAC^{SS_{OPT}}$
ieee_118	0.212	0.029	<b>0.026</b>
ACTIVSg_200	0.113	0.004	<b>0.002</b>
ACTIVSg_500	0.122	0.015	<b>0.002</b>
pegase_1354	0.131	<b>0.001</b>	0.002
pegase_2869	0.145	0.041	<b>0.031</b>
pegase_9241	0.254	0.042	<b>0.025</b>

\* $DCAC^{BASE}$  and \* $DCAC^{SS_{INIT}}$  denote unoptimized pipelines; their losses are evaluated using (4) upon ACPF convergence.

smooth pipelines. Compared with  $DCAC^{SS_{INIT}}$ , the optimized smooth pipeline reduces the evaluation loss on most systems, including ACTIVSg\_200, ACTIVSg\_500, pegase\_2869, and pegase\_9241. For example, the loss decreases from 0.042 to 0.025 on pegase\_9241 and from 0.041 to 0.031

on pegase\_2869. The baseline produces much larger losses across all cases, confirming that the smooth restoration structure substantially improves recovery quality over conventional single-slack ACPF. This is consistent with the findings in [14]. Table III further shows that optimization improves cost consistency, dispatch accuracy, and computational performance on the larger systems. On pegase\_1354,  $DCAC^{SS_{OPT}}$  reduces the CD from 0.333% to 0.044% relative to  $DCAC^{SS_{INIT}}$ , while reducing MAE from 0.048 to 0.040 p.u. On pegase\_2869, the cost difference decreases from 0.387% to 0.021% and MAE reduces from 0.041 to 0.030 p.u. On pegase\_9241, the cost difference is lowered also, from 0.510% to 0.124% and MAE follows that trend, from 0.035 to 0.025 p.u. The optimized smooth pipeline also lowered average solve time on these large cases. These results show that optimizing the smooth restoration parameters improves both AC feasibility restoration and ACPF-tracking performance relative to the initialized smooth pipeline, while remaining substantially better than the single-slack baseline.

TABLE V: AVERAGE CONSTRAINT VIOLATION RATES AND MAXIMUM VIOLATION MAGNITUDES FOR  $\text{DCAC}^{\text{BASE}}$ ,  $\text{DCAC}^{\text{DD}_{\text{init}}}$ , AND  $\text{DCAC}^{\text{DD}_{\text{opt}}}$  ACROSS 1,000 LOAD SAMPLES (USING  $\text{DC}_{\text{BASE}}$  SETPOINTS)

Test case	Method	Active Power (p.u.)		Reactive Power (p.u.)		Voltage Limit (p.u.)		Line Limit (%)	
		% viol.	Max	% viol.	Max	% viol.	Max	% viol.	Max
ieee_118	$\text{DCAC}^{\text{BASE}}$	1.61	0.01	5.56	2.93	0.05	0.01	0.00	0.20
	$\text{DCAC}^{\text{DD}_{\text{init}}}$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	$\text{DCAC}^{\text{DD}_{\text{opt}}}$	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>
ACTIVSg_200	$\text{DCAC}^{\text{BASE}}$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	$\text{DCAC}^{\text{DD}_{\text{init}}}$	0.00	0.00	0.00	0.00	0.07	0.03	0.00	0.00
	$\text{DCAC}^{\text{DD}_{\text{opt}}}$	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>
ACTIVSg_500	$\text{DCAC}^{\text{BASE}}$	0.00	0.00	8.03	0.06	0.00	0.00	0.00	0.00
	$\text{DCAC}^{\text{DD}_{\text{init}}}$	0.00	0.00	0.00	0.00	0.00	0.00	0.16	1.46
	$\text{DCAC}^{\text{DD}_{\text{opt}}}$	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>
pegase_1354	$\text{DCAC}^{\text{BASE}}$	0.46	0.02	18.46	18.28	0.00	0.00	0.00	0.00
	$\text{DCAC}^{\text{DD}_{\text{init}}}$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	$\text{DCAC}^{\text{DD}_{\text{opt}}}$	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>
pegase_2869	$\text{DCAC}^{\text{BASE}}$	0.51	2.63	15.69	13.73	0.00	0.00	0.00	0.00
	$\text{DCAC}^{\text{DD}_{\text{init}}}$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	$\text{DCAC}^{\text{DD}_{\text{opt}}}$	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>
pegase_9241	$\text{DCAC}^{\text{BASE}}$	2.31	14.66	14.74	10.71	17.23	0.07	2.31	44.73
	$\text{DCAC}^{\text{DD}_{\text{init}}}$	0.00	0.00	0.00	0.00	0.89	0.03	0.00	0.00
	$\text{DCAC}^{\text{DD}_{\text{opt}}}$	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>0.00</b>	<b>1.02</b>	<b>0.04</b>	<b>0.00</b>	<b>0.00</b>

TABLE VI: PERFORMANCE TABLE FOR  $\text{DCAC}^{\text{BASE}}$ ,  $\text{DCAC}^{\text{DD}_{\text{init}}}$  [14], AND  $\text{DCAC}^{\text{DD}_{\text{opt}}}$  ACROSS 1,000 SAMPLES ( $\text{DC}_{\text{BASE}}$  SETPOINTS)

Test case	Method	Cost Diff. (%)		MAE* (p.u.)		Iter. Count		Solve Time (s)	
		val.	%Improv.	val.	%Improv.	val.	%Improv.	val.	%Improv.
ieee_118	$\text{DCAC}^{\text{BASE}}$	1.488	<b>+31.0</b>	0.025	<b>+8.0</b>	4.0	—	0.073	—
	$\text{DCAC}^{\text{DD}_{\text{init}}}$	1.035	<b>+0.8</b>	0.023	0.0	6.2	-1.6	0.319	<b>+1.3</b>
	$\text{DCAC}^{\text{DD}_{\text{opt}}}$	1.027	—	0.023	—	6.3	—	0.315	—
ACTIVSg_200	$\text{DCAC}^{\text{BASE}}$	0.035	—	0.003	<b>+33.3</b>	2.0	—	0.065	—
	$\text{DCAC}^{\text{DD}_{\text{init}}}$	0.522	-14.2	0.003	<b>+33.3</b>	4.0	-50.0	0.186	-14.5
	$\text{DCAC}^{\text{DD}_{\text{opt}}}$	0.596	—	0.002	—	6.0	—	0.213	—
ACTIVSg_500	$\text{DCAC}^{\text{BASE}}$	1.383	-27.4	0.010	0.0	3.0	—	0.113	—
	$\text{DCAC}^{\text{DD}_{\text{init}}}$	0.160	—	0.010	0.0	4.1	-53.7	0.288	-30.6
	$\text{DCAC}^{\text{DD}_{\text{opt}}}$	1.762	—	0.010	—	6.3	—	0.376	—
pegase_1354	$\text{DCAC}^{\text{BASE}}$	0.394	<b>+23.9</b>	0.050	<b>+6.0</b>	4.0	—	0.361	—
	$\text{DCAC}^{\text{DD}_{\text{init}}}$	0.354	<b>+15.3</b>	0.049	<b>+4.1</b>	9.7	<b>+16.5</b>	1.496	0.0
	$\text{DCAC}^{\text{DD}_{\text{opt}}}$	0.300	—	0.047	—	8.1	—	1.496	—
pegase_2869	$\text{DCAC}^{\text{BASE}}$	0.458	<b>+36.0</b>	0.042	<b>+9.5</b>	4.0	—	0.915	—
	$\text{DCAC}^{\text{DD}_{\text{init}}}$	0.406	<b>+27.8</b>	0.041	<b>+7.3</b>	13.6	<b>+25.0</b>	4.168	<b>+3.3</b>
	$\text{DCAC}^{\text{DD}_{\text{opt}}}$	0.293	—	0.038	—	10.2	—	4.031	—
pegase_9241	$\text{DCAC}^{\text{BASE}}$	0.631	<b>+17.4</b>	0.038	<b>+13.2</b>	3.0	—	3.389	—
	$\text{DCAC}^{\text{DD}_{\text{init}}}$	0.533	<b>+2.3</b>	0.036	<b>+8.3</b>	16.4	<b>+6.7</b>	15.410	<b>+17.3</b>
	$\text{DCAC}^{\text{DD}_{\text{opt}}}$	0.521	—	0.033	—	15.3	—	12.743	—

The % Improv. report values relative to  $\text{DCAC}^{\text{DD}_{\text{opt}}}$  for each metric.  $\text{DCAC}^{\text{DD}_{\text{opt}}}$  uses optimized values from  $\text{DCAC}^{\text{SS}_{\text{opt}}}$ . See Section III-B. The initialized discrete pipeline,  $\text{DCAC}^{\text{DD}_{\text{init}}}$ , uses initial parameter values: voltage setpoints initialized to 1.0 p.u., with participation factor 1.0 assigned to the slack bus and 0.0 to all other buses.

TABLE VII: SOLVE TIME OF OPTIMIZED  $\text{DCAC}$  VARIANTS UNDER BASE CASE (USING  $\text{DC}_{\text{BASE}}$  SETPOINTS)

Test case	$\text{DCAC}^{\text{SS}_{\text{opt}}}$ (s)	$\text{DCAC}^{\text{DD}_{\text{opt}}}$ (s)	ACOPF (s)
ieee_118	<b>0.20</b>	0.21 (+4.8)	0.64 (+68.8)
ACTIVSg_200	<b>0.16</b>	0.20 (+20.0)	0.36 (+55.6)
ACTIVSg_500	<b>0.24</b>	0.32 (+25.0)	1.48 (+83.8)
pegase_1354	<b>0.78</b>	1.53 (+49.0)	2.79 (+72.0)
pegase_2869	<b>1.92</b>	3.76 (+48.9)	4.93 (+61.1)
pegase_9241	<b>7.62</b>	11.97 (+36.3)	30.28 (+74.8)

Parentheses show  $\text{DCAC}^{\text{SS}_{\text{opt}}}$  solve-time % Improv. relative to that method.  $\text{DCAC}^{\text{SS}_{\text{opt}}}$  uses the smooth PACR approach, while  $\text{DCAC}^{\text{DD}_{\text{opt}}}$  uses discrete mode.

### B. AC Feasibility Restoration Assessment: Initialized versus Optimized Discrete Pipelines (( $\text{DCAC}^{\text{DD}_{\text{init}}}$ ) vs. ( $\text{DCAC}^{\text{DD}_{\text{opt}}}$ )):

This subsection evaluates the effect of transferring the optimized parameters learned from the smooth PACR formulation to the corresponding discrete restoration pipeline. Specifically, the initialized discrete pipeline,  $\text{DCAC}^{\text{DD}_{\text{init}}}$  [14], is compared against the optimized discrete pipeline,  $\text{DCAC}^{\text{DD}_{\text{opt}}}$ , with  $\text{DCAC}^{\text{BASE}}$  included as a conventional single-slack reference.

This was done to assess whether the learned distributed-slack participation factors and voltage-setpoint values improve AC feasibility restoration in the discrete ACPF procedure relative to the initialized discrete pipeline.

1) *Constraint Violation Table Analysis:* Table V reports constraint violations for  $\text{DCAC}^{\text{BASE}}$ ,  $\text{DCAC}^{\text{DD}_{\text{init}}}$ , and  $\text{DCAC}^{\text{DD}_{\text{opt}}}$ . As in the smooth-pipeline comparison in Section IV-A, the single-slack baseline leaves significant residual infeasibilities, including reactive-power violation rates of 8.03% on ACTIVSg\_500 and 18.46% on pegase\_1354. On the largest case, pegase\_9241, the baseline also produces average active-power, voltage-limit, and line-limit violations.

Both discrete restoration pipelines substantially improve feasibility relative to this baseline. The initialized discrete pipeline already removes all active- and reactive-power violations, but small residual voltage or line-limit violations remain on a few cases. The optimized discrete pipeline improves several of these residual violations: it removes the small voltage violation on ACTIVSg\_200, removes the

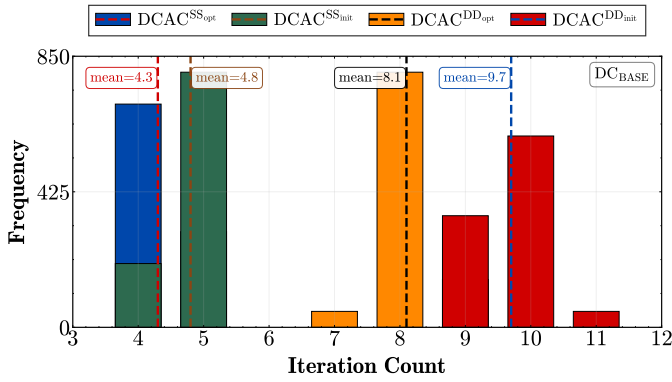
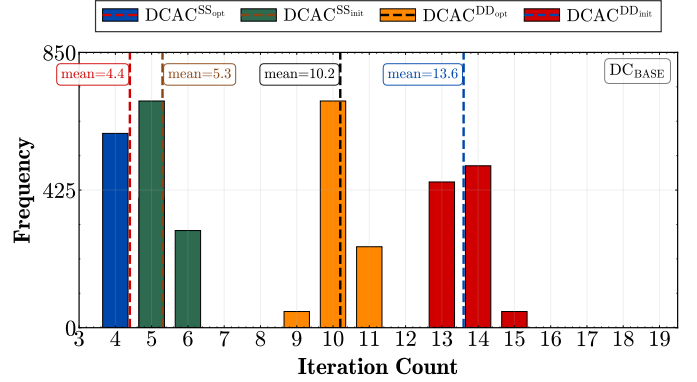
(a) Iteration count for `pegase_1354` on 1,000 samples.(b) Iteration count for `pegase_2869` on 1,000 samples.

Fig. 4: Iteration count comparison for the four DCAC variants under load uncertainty using 1,000 samples per case. (a) `pegase_1354`. (b) `pegase_2869`. The compared methods are  $\text{DCAC}^{\text{SS}_{\text{opt}}}$  (blue),  $\text{DCAC}^{\text{SS}_{\text{init}}}$  (green),  $\text{DCAC}^{\text{DD}_{\text{opt}}}$  (orange), and  $\text{DCAC}^{\text{DD}_{\text{init}}}$  [14] (red). The ‘opt’ variant uses parameters trained via Algorithm 1, and ‘init’ uses initial flat-start parameter values.

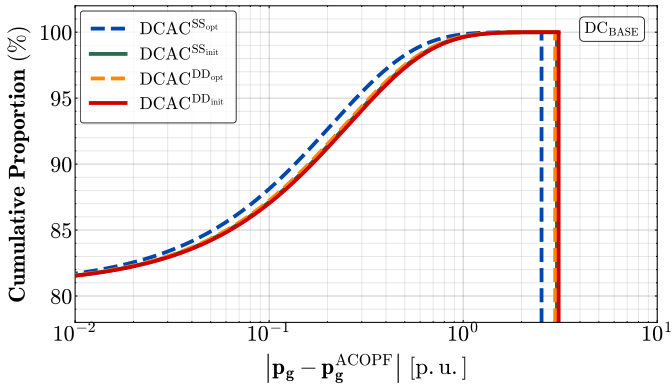
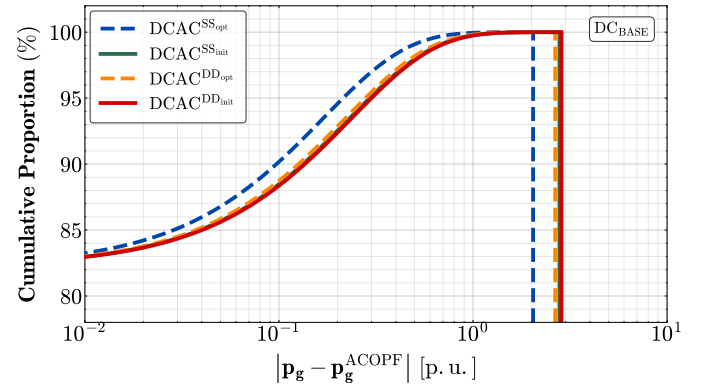
(a) Cumulative plot of  $p_g$  error for `pegase_2869` on 1,000 samples.(b) Cumulative plot of  $p_g$  error for `pegase_9241` on 1,000 samples.

Fig. 5: Empirical cumulative distribution of active-power dispatch error,  $|p_g - p_g^{\text{ACOPF}}|$ , using 1,000 samples per case. (a) `pegase_2869`. (b) `pegase_9241`. The DCAC methods are  $\text{DCAC}^{\text{SS}_{\text{opt}}}$  (blue),  $\text{DCAC}^{\text{SS}_{\text{init}}}$  (green),  $\text{DCAC}^{\text{DD}_{\text{opt}}}$  (orange), and  $\text{DCAC}^{\text{DD}_{\text{init}}}$  [14] (red). The ‘opt’ variant uses parameters trained via Algorithm 1, and ‘init’ uses initial flat-start parameter values.

line-limit violation on `ACTIVSg_500`, and achieves zero violations on `ieee_118`, `ACTIVSg_200`, `ACTIVSg_500`, `pegase_1354`, and `pegase_2869`. On `pegase_9241`, it eliminates active-power, reactive-power, and line-limit violations, while leaving a small average voltage-violation rate of 1.02%. These results show that the optimized parameters learned in the smooth formulation transfer effectively to the discrete restoration setting. This is important because many existing ACPF tools use discrete PV/PQ switching and distributed-slack logic; therefore, the learned parameters can improve standard discrete recovery procedures without requiring a fully differentiable solver implementation.

2) *Performance Table Analysis*: Table VI compares cost difference, generator active-power MAE, iteration count, and solve time for the baseline and discrete restoration pipelines. The optimized discrete pipeline improves the restored operating point relative to the initialized discrete pipeline on the larger PEGASE systems. On `pegase_1354`, the cost difference decreases from 0.354% to 0.300% and MAE decreases from 0.049 to 0.047 p.u. On `pegase_9241`, MAE decreases from 0.036 to 0.033 p.u., while solve time is lowered from 15.410 s to 12.743 s. Relative to the single-slack baseline,  $\text{DCAC}^{\text{DD}_{\text{opt}}}$  provides stronger feasibility restoration and better

dispatch accuracy on the large cases. For `pegase_2869`, it attains a 0.038 p.u. MAE and 0.293% cost difference, compared with 0.042 p.u. and 0.458% for  $\text{DCAC}^{\text{BASE}}$ . For `pegase_9241`, it achieves 0.033 p.u. MAE and 0.521% cost difference, compared with 0.038 p.u. and 0.631% for the baseline. Although the discrete restoration pipeline requires more Newton iterations than the single-slack ACPF baseline, it produces a much stronger AC feasibility recovery, confirming that optimized distributed slack and voltage regulation improve the quality of the restored AC operating point.

### C. Complete Performance Comparison of Discrete and Smooth DCAC Pipelines ( $\text{DCAC}^{\text{DD}}$ vs. $\text{DCAC}^{\text{SS}}$ ):

This subsection directly compares the proposed smooth restoration pipeline against the discrete restoration pipeline [14], after parameter optimization. The purpose is to separate the benefit of the learned parameters from the additional benefit of using a smooth differentiable ACPF formulation. The results show that although both optimized pipelines improve restoration quality, as shown in Sections IV-A and IV-B, the smooth pipeline achieves better accuracy, with substantially lower computational effort than our former discrete framework [14].

1) *Optimized Pipeline Solve-Time Analysis:* Table VII compares the solve times of  $\text{DCAC}^{\text{SSopt}}$ ,  $\text{DCAC}^{\text{DDopt}}$ , and ACOFP across all test cases using  $\text{DC}_{\text{BASE}}$  setpoints. The optimized smooth pipeline is the fastest method in every case, with solve times ranging from 0.16 s on `ACTIVSg_200` to 7.62 s on `pegase_9241`. Relative to the optimized discrete pipeline,  $\text{DCAC}^{\text{SSopt}}$  reduces solve time by 4.8%–49.0%. Relative to ACOFP, the reduction is larger, ranging from 55.6% to 83.8%.

This shows that the smooth formulation improves computational efficiency while preserving the speed advantage of DC-based dispatch followed by AC feasibility restoration. The gains over  $\text{DCAC}^{\text{DDopt}}$  suggest that avoiding discrete distributed-slack and PV/PQ switching updates reduces the computational burden of the restoration step.

2) *Iteration Count Figure Analysis:* Figures 4a and 4b compare Newton iteration-count distributions for `pegase_1354` and `pegase_2869` across 1,000 load-uncertainty scenarios using  $\text{DC}_{\text{BASE}}$  setpoints. The optimized smooth pipeline  $\text{DCAC}^{\text{SSopt}}$  concentrates most of its probability mass at the lowest iteration counts, with mean values of approximately 4.3 and 4.4 iterations on `pegase_1354` and `pegase_2869`, respectively. The initialized smooth version  $\text{DCAC}^{\text{SSinit}}$  remains competitive, with means of 4.8 and 5.3 iterations, indicating that the smooth formulation itself provides a favorable numerical structure even before training. By contrast, the discrete pipelines require substantially more iterations. On `pegase_1354`,  $\text{DCAC}^{\text{DDopt}}$  and  $\text{DCAC}^{\text{DDinit}}$  average approximately 8.1 and 9.7 iterations, respectively. On `pegase_2869`, the corresponding averages are approximately 10.2 and 13.6.

These results highlight a tradeoff between the two implementations. The smooth pipeline provides the best accuracy and speed because it avoids outer-loop bus-type switching and discontinuous PV/PQ updates [15], [16]. However, the discrete pipeline is often more directly compatible with traditional power-flow packages that implement PV/PQ switching and distributed slack through discrete logic. The relatively small degradation of  $\text{DCAC}^{\text{DDopt}}$  compared with  $\text{DCAC}^{\text{SSopt}}$  therefore suggests that parameters optimized in the smooth formulation can still be useful in standard discrete ACPF implementations.

3) *Cumulative Error Plot Analysis:* Figures 5a and 5b report the empirical cumulative distributions of the generator active-power dispatch error,  $|\mathbf{p}_g - \mathbf{p}_g^{\text{ACOPF}}|$ , for `pegase_2869` and `pegase_9241`. Near the  $10^{-2}$  p.u. threshold, all variants show similar cumulative behavior. For `pegase_2869`,  $\text{DCAC}^{\text{SSopt}}$  places approximately 81.9% of generator errors below this threshold, compared with 81.8% for  $\text{DCAC}^{\text{DDopt}}$  and about 81.6% for the initialization-based variants. For `pegase_9241`, the optimized smooth pipeline again gives the highest proportion, approximately 83.3%, while the other methods remain close at about 83.0%.

The main separation appears in the right tail. On `pegase_2869`, the worst-case error for  $\text{DCAC}^{\text{SSopt}}$  is approximately 2.53 p.u., compared with roughly 2.97–3.17 p.u. for the other variants. On `pegase_9241`, the optimized smooth pipeline reaches a worst-case error of approximately 2.29 p.u., while the other variants lie between about 2.99 and 3.22 p.u. Thus, the optimized smooth formulation does not only improve average behavior; it also reduces the largest dispatch deviations,

which is important for robust restoration under load uncertainty.

## V. CONCLUSION

This work develops a smooth, differentiable ACPF restoration learning method (PACR) for mapping DCOPF dispatches to AC-consistent operating points. The method replaces discrete restoration logic in our previous work [14], with smooth surrogates, including softmax-based distributed slack and sigmoid PV/PQ regulation, allowing voltage setpoints and slack-participation parameters to be optimized. Experiments on IEEE, ACTIVSg, and PEGASE systems show that the optimized pipelines improve AC feasibility, reduce Newton iterations, and lower dispatch errors relative to conventional single-slack ACPF recovery. The gains are strongest on large PEGASE cases: the optimized smooth DCAC method reduces cost difference by approximately 95% on the 2,869-bus case and 80% on the 9,241-bus case, while solving the 9,241-bus case about 75% faster than ACOFP. Overall, the optimized smooth pipeline is the best-performing restoration method, and its learned participation factors and voltage setpoints also transfer effectively to the discrete DCAC pipeline [14].

Future work will extend the differentiable restoration method to additional corrective controls, including transformer taps and capacitor shunts, and will explore end-to-end self-supervised training of the full DCOPF→ACPF pipeline. This direction connects naturally with recent work on parameterized and optimized DCOPF models [7], [8], [27], while posing a more demanding learning problem: the DCOPF parameters must be optimized not only for dispatch cost, but also for their downstream effect on AC inequality constraint violations.

## REFERENCES

- [1] B. Stott, J. Jardim, and O. Alsac, “DC power flow revisited,” *IEEE Transactions on Power Systems*, vol. 24, no. 3, pp. 1290–1300, 2009.
- [2] W. Chen, M. Tanneau, and P. Van Hentenryck, “End-to-end feasible optimization proxies for large-scale economic dispatch,” *IEEE Transactions on Power Systems*, vol. 39, pp. 4723–4734, Mar. 2024.
- [3] R. Ferrando, L. Pagnier, R. Mieth, Z. Liang, Y. Dvorkin, D. Bienstock, and M. Chertkov, “Physics-informed machine learning for electricity markets: A NYISO case study,” *IEEE Transactions on Energy Markets, Policy and Regulation*, vol. 2, pp. 40–51, Mar. 2024.
- [4] M. Li, S. Kolouri, and J. Mohammadi, “Learning to solve optimization problems with hard linear constraints,” *IEEE Access*, vol. 11, pp. 59995–60004, 2023.
- [5] Y. Chen and M. K. Singh, “Optimally Linearizing Power Flow Equations for Improved Power System Dispatch,” 2025. arXiv preprint arXiv:2504.03076.
- [6] B. Taheri and D. K. Molzahn, “AC power flow feasibility restoration via a state estimation-based post-processing algorithm,” *Electric Power Systems Research*, vol. 235, Oct. 2024. Presented at the *23rd Power Systems Computation Conference (PSCC)*.
- [7] B. Taheri and D. K. Molzahn, “AC-Informed DC Optimal Transmission Switching Problems via Parameter Optimization,” *IEEE Transactions on Power Systems*, vol. 40, pp. 5422–5433, November 2025.
- [8] G. E. Constante-Flores, A. H. Quisaguano, A. J. Conejo, and C. Li, “AC-Network-Informed DC Optimal Power Flow for Electricity Markets,” 2024. arXiv preprint arXiv:2410.18413.
- [9] P. L. Donti, D. Rolnick, and J. Z. Kolter, “DC3: A learning method for optimization with hard constraints,” in *International Conference on Learning Representations (ICLR)*, 2021.
- [10] W. Huang, X. Pan, M. Chen, and S. H. Low, “DeepOPF-V: Solving AC-OPF problems efficiently,” *IEEE Transactions on Power Systems*, vol. 37, no. 1, pp. 800–803, 2022.

- [11] K. Baker, "Solutions of DC OPF are never AC feasible," in *Proceedings of the Twelfth ACM International Conference on Future Energy Systems, e-Energy '21*, (New York, NY, USA), p. 264–268, Association for Computing Machinery, 2021.
- [12] X. Fang, Z. Yang, J. Yu, and Y. Wang, "AC feasibility restoration in market clearing: Problem formulation and improvement," *IEEE Trans. on Industrial Informatics*, vol. 18, pp. 7597–7607, Nov. 2022.
- [13] Y. Wang and Z. Yang, "Co-optimize dispatch and pricing in market clearing problem to balance conflicting market objectives," *Applied Energy*, vol. 403, p. 127065, 2026.
- [14] M. A. Boateng, R. Bent, S. Misra, P. Pareek, P. Van Hentenryck, and D. Molzahn, "Towards AC feasibility of DCOFP dispatch," to appear in *Electric Power Systems Research*, 2026. Presented at the *24th Power Systems Computation Conference (PSCC)*, June 8–12, 2026.
- [15] J. Zhao, H. Chiang, P. Ju, and H. Li, "On PV-PQ bus type switching logic in power flow computation," in *16th Power Systems Computation Conference (PSCC)*, 2008.
- [16] L. Zeng, H.-D. Chiang, L. S. Neves, and L. F. C. Alberto, "On the accuracy of power flow and load margin calculation caused by incorrect logical PV/PQ switching: Analytics and improved methods," *International Journal of Electrical Power & Energy Systems*, vol. 147, 2023.
- [17] V. N. Bharatwaj, A. R. Abhyankar, and P. R. Bijwe, "Iterative DCOFP model using distributed slack bus," in *IEEE Power and Energy Society General Meeting*, 2012.
- [18] S. V. Dhople, Y. C. Chen, A. Al-Digs, and A. D. Domínguez-García, "Reexamining the distributed slack bus," *IEEE Transactions on Power Systems*, vol. 35, no. 6, pp. 4870–4879, 2020.
- [19] M. Jereminov, D. M. Bromberg, A. Pandey, M. R. Wagner, and L. Pileggi, "Evaluating feasibility within power flow," *IEEE Transactions on Smart Grid*, vol. 11, no. 4, pp. 3522–3534, 2020.
- [20] T. McNamara, A. Pandey, A. Agarwal, and L. Pileggi, "Two-stage homotopy method to incorporate discrete control variables into AC-OPF," *Electric Power Systems Research*, vol. 212, p. 108283, 2022. Presented at the *22nd Power Systems Computation Conference (PSCC)*.
- [21] A. Agarwal, A. Pandey, M. Jereminov, and L. Pileggi, "Continuously differentiable analytical models for implicit control within power flow," *arXiv:1811.02000*, Nov. 2018.
- [22] A. S. Zamzam and K. Baker, "Learning optimal solutions for extremely fast ac optimal power flow," in *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, pp. 1–6, 2020.
- [23] A. Van Boven and K. Baker, "Bus type switching to reduce bound violations in AC power flow," *arXiv:2511.08643*, 2025.
- [24] C. Coffrin, R. Bent, K. Sundar, Y. Ng, and H. Wang, "PowerModels.jl: An open-source framework for exploring power flow formulations," in *20th Power Systems Computation Conference (PSCC)*, 2018.
- [25] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, pp. 25–57, 2006.
- [26] L. Thurner, A. Scheidler, F. Schäfer, J.-H. Menke, J. Dollichon, F. Meier, and J. M. Myrzik, "pandapower—An open-source Python tool for convenient modeling, analysis, and optimization of electric power systems," *IEEE Transactions on Power Systems*, vol. 33, p. 6510, 2018.
- [27] A. W. Roseberg and M. Klamkin, "Differentiable optimization for deep learning-enhanced DC approximation of AC optimal power flow," *arXiv:2504.01970*, 2025.

## APPENDIX A

### Q–V SIGMOID CURVE AT VOLTAGE SETPOINT

This appendix verifies that the smooth Q–V regulation law passes through the prescribed setpoint  $(v_i^{sp}, q_{g,i}^{sp})$ .

*Claim:* If  $v_i = v_i^{sp}$ , then  $q_{g,i}^{ac} = q_{g,i}^{sp}$ .

*Proof:* Let  $q_{g,i}^{min} < q_{g,i}^{sp} < q_{g,i}^{max}$  and  $\psi_s > 0$ .

$$q_{g,i}^{ac} = q_{g,i}^{min} + \frac{q_{g,i}^{max} - q_{g,i}^{min}}{1 + \exp\left(\psi_s(v_i - v_i^{sp}) + \ln \frac{q_{g,i}^{max} - q_{g,i}^{sp}}{q_{g,i}^{sp} - q_{g,i}^{min}}\right)}. \quad (7)$$

At  $v_i = v_i^{sp}$ ,

$$q_{g,i}^{ac} = q_{g,i}^{min} + \frac{q_{g,i}^{max} - q_{g,i}^{min}}{1 + \frac{q_{g,i}^{max} - q_{g,i}^{sp}}{q_{g,i}^{sp} - q_{g,i}^{min}}} = q_{g,i}^{min} + (q_{g,i}^{sp} - q_{g,i}^{min}) = q_{g,i}^{sp}. \quad (8)$$

Hence, the parameterized sigmoid passes through  $(v_i^{sp}, q_{g,i}^{sp})$ .

## APPENDIX B

### DIFFERENTIATING THROUGH THE AUGMENTED ACPF

This appendix provides the mathematical framework for computing gradients through the augmented ACPF system to enable parameter optimization via backpropagation.

Let the converged augmented ACPF solution be  $\mathbf{x}_{ac} = (\boldsymbol{\theta}, \mathbf{v}, \ell, \mathbf{p}_g^{ac}, \mathbf{q}_g^{ac})$  and define the trainable parameters

$$\boldsymbol{\xi} \triangleq \{\pi_s, \phi_s, \psi_s, \mathbf{v}^{sp}\}, \quad (9)$$

where  $\psi_t$  is treated as a fixed (or bounded) parameter that shifts effective reactive limits (see Model 2). For each scenario,  $\mathbf{x}_{ac}$  satisfies the implicit system

$$\mathbf{g}(\mathbf{x}_{ac}; \boldsymbol{\xi}) = \begin{bmatrix} \Delta \mathbf{p}(\mathbf{x}_{ac}; \boldsymbol{\xi}) \\ \Delta \mathbf{q}(\mathbf{x}_{ac}; \boldsymbol{\xi}) \\ \ell^{\text{tot}}(\mathbf{x}_{ac}) \end{bmatrix} = \mathbf{0}. \quad (10)$$

Given the full-batch objective:

$$\mathcal{J} = \frac{1}{|\mathcal{D}|} \sum_{\mathcal{D}} [w_P \|\mathbf{p}_g^{ac} - \mathbf{p}_g^{\text{ref}}\|_1 + w_V \|\mathbf{v} - \mathbf{v}^{\text{ref}}\|_1] + \mathcal{R}(\boldsymbol{\xi}), \quad (11)$$

gradients are computed by implicit differentiation. Let  $\mathbf{J}_{ac} = \partial \mathbf{g} / \partial \mathbf{x}_{ac}$ . For each scenario,

$$\mathbf{J}_{ac}^\top \boldsymbol{\lambda} = \left( \frac{\partial \mathcal{L}^{\text{sc}}}{\partial \mathbf{x}_{ac}} \right)^\top, \quad \frac{\partial \mathcal{J}}{\partial \boldsymbol{\xi}} = -\frac{1}{|\mathcal{D}|} \sum_{\mathcal{D}} \boldsymbol{\lambda}^\top \frac{\partial \mathbf{g}}{\partial \boldsymbol{\xi}} + \frac{\partial \mathcal{R}}{\partial \boldsymbol{\xi}}. \quad (12)$$

### A. Where Parameters Enter the Mismatches

Parameters affect  $\Delta \mathbf{p}$  through smooth distributed slack  $p_{g,i}^{\text{dc}} + \alpha_i \ell$  and affect  $\Delta \mathbf{q}$  through the smooth Q–V map  $q_{g,i}^{ac}(v_i)$ :

$$\frac{\partial \Delta p_i}{\partial \alpha_i} = \ell, \quad \frac{\partial \Delta q_i}{\partial \boldsymbol{\xi}} = \frac{\partial q_{g,i}^{ac}}{\partial \boldsymbol{\xi}}.$$

### B. Differentiable Building Blocks

a) *Softplus headroom:* Let  $r_i = p_{g,i}^{\text{max}} - p_{g,i}^{\text{dc}}$  and  $\tilde{h}_i = \frac{1}{\pi_s} \ln(1 + e^{\pi_s r_i}) - \frac{\ln 2}{\pi_s}$ . Then

$$\frac{\partial \tilde{h}_i}{\partial r_i} = \sigma(\pi_s r_i), \quad \sigma(u) = \frac{1}{1 + e^{-u}}. \quad (13)$$

b) *Softmax participation:*  $\alpha_i = \frac{e^{\phi_s \tilde{h}_i}}{\sum_{j \in \mathcal{G}} e^{\phi_s \tilde{h}_j}}$  implies

$$\frac{\partial \alpha_i}{\partial \phi_s} = \alpha_i \left( \tilde{h}_i - \sum_{j \in \mathcal{G}} \alpha_j \tilde{h}_j \right), \quad \frac{\partial \alpha_i}{\partial \tilde{h}_k} = \phi_s \alpha_i (\mathbb{I}[i = k] - \alpha_k). \quad (14)$$

(Chain rule gives  $\partial \alpha_i / \partial \pi_s$  through  $\tilde{h}$  if  $\pi_s$  is trained.)

c) *Sigmoid Q–V regulation:* Let  $q_{g,i}^{ac} = q_{g,i}^{\text{min,eff}} + \Delta q_i^{\text{eff}} s_i$  where  $s_i = (1 + e^{\zeta_i})^{-1}$  and  $\zeta_i = \psi_s(v_i - v_i^{sp}) + (\dots)$ . The  $(\dots)$  term collects optional offsets (e.g., the log-offset enforcing passage through  $(v_i^{sp}, q_{g,i}^{sp})$ ) and is omitted here for brevity; including it preserves differentiability and only adds chain-rule terms (Model 2). Then  $s_i(1 - s_i)$  provides compact derivatives:

$$\frac{\partial q_{g,i}^{ac}}{\partial \psi_s} = -\Delta q_i^{\text{eff}} s_i(1 - s_i)(v_i - v_i^{sp}), \quad \frac{\partial q_{g,i}^{ac}}{\partial v_i^{sp}} = \Delta q_i^{\text{eff}} \psi_s s_i(1 - s_i). \quad (15)$$