Paper Summary - End-to-End Feasible Optimization Proxies for Large-Scale Economic Dispatch

Ilay Menachem

January 18, 2026

Notation

Sets and Indices

$i \in \mathcal{N}$ buses

$e \in \mathcal{E}$ branches

$g \in \mathcal{G}$ generators

Variables

p_g Energy dispatch of generator g

r_g Reserve dispatch of generator g

ξ_e Thermal limit violation on branch e

Domains

$$\mathcal{H} = \left\{ \mathbf{p} \in \mathbb{R}^{|\mathcal{G}|} \mid 0 \leq \mathbf{p} \leq \overline{\boldsymbol{p}} \right\}$$
$$\mathcal{S}_D = \left\{ \mathbf{p} \in \mathbb{R}^{|\mathcal{G}|} \mid \mathbf{e}^{\top} \mathbf{p} = D \right\} \cap \mathcal{H}$$

Parameters

d_i Active power demand at bus i

c_g Production cost function of generator g

$\underline{f}_e, \overline{f}_e$ Lower and upper thermal limits on branch e

M_{th} Thermal violation penalty cost

$\overline{p}_g$ Maximum output of generator g

$\overline{r}_g$ Maximum reserve of generator g

R Minimum reserve requirement

Φ Power Transfer Distribution Factor matrix

Problem statement

The problem of economic dispatch is to find the optimal $\boldsymbol{p}$, and $\boldsymbol{r}$ such that the total production cost plus a penalty for thermal limit violations is minimized. here is the formal problem statement:

$$\min_{\boldsymbol{p}, \boldsymbol{r}, \boldsymbol{\xi}_{th}} c(\boldsymbol{p}) + M_{th} ||\boldsymbol{\xi}_{th}||_1 \tag{1}$$

subject to

1. $e^T \boldsymbol{p} = e^T \boldsymbol{d}$ (global power balance)

2. $e^T \boldsymbol{r} \geq R$ (minimum reserve requirement)

3. $\boldsymbol{p} + \boldsymbol{r} \leq \overline{\boldsymbol{p}}$ (generator output limits)

4. $0 \leq \boldsymbol{p} \leq \overline{\boldsymbol{p}}$ (generator output limits)

5. $0 \leq \boldsymbol{r} \leq \overline{\boldsymbol{r}}$ (generator reserve limits)

6. $\underline{\boldsymbol{f}} - \boldsymbol{\xi}_{th} \leq \boldsymbol{\Phi}(\boldsymbol{p} - \boldsymbol{d}) \leq \overline{\boldsymbol{f}} + \boldsymbol{\xi}_{th}$ (thermal limit violations)

7. $\boldsymbol{\xi}_{th} \geq 0$ (thermal limit violations non-negativity)

Introduction

There is a great trend in grid management to use Deep Learning to optimize grid operations. there is also a trend In Deep learning to respect problem structure and constraints, such as equivariant neural networks and networks that respect inequalities. networks that respect problem structure and constraints have shown to learn more efficiently and achive better performance. This paper proposes a new closed form way to make a neural network that solves the economic dispatch problem respect the problem constraints.

Previous works

Optimization Proxies for Optimal Power Flow (OPF)

The majority of existing research that employs Supervised Learning to approximate OPF solutions. While applied successfully to both DC-OPF and AC-OPF, SL models rely on fixed grid topologies and generator commitments. Real-world changes to these parameters require computationally expensive retraining and data re-generation. Graph Neural Networks (GNNs) have been proposed to handle topology changes, but they have mostly been tested on small networks. Recently, Self-Supervised Learning has emerged as a promising alternative to Supervised learning. By training the model to minimize the objective function and constraint penalties directly, Self-Supervised Learning eliminates the need for labeled data and offline optimization solving.

Ensuring Feasibility

A critical limitation of ML proxies is that predictions often violate physical constraints. Literature has explored several strategies:

- **Restricted Training**: Artificially shrinking the feasible region to force solutions "inside" the bounds, which is computationally cumbersome.

- **Active Set Learning**: Predicting which constraints are active to recover a solution, though incorrect classifications lead to infeasibility.

- **Physics-Informed Models**: Adding penalty terms to the loss function. While this reduces violations, it does not guarantee a feasible output.

- **Post-Processing**: Using projection steps or power flow solvers to repair solutions after prediction. This adds significant computational time.

- **End-to-End Layers**: Recent methods embed feasibility restoration into the neural network itself. However, current implementations either fail to guarantee full feasibility or rely on restrictive assumptions (e.g., convexity) that do not hold for general problems.

Scalability Challenges

There is a significant disconnect between academic studies and industrial reality. Most research utilizes small test systems with fewer than 300 buses. In contrast, real-world power grids contain tens of thousands of buses. Very few existing studies report results on systems larger than 6,000 buses , making it difficult to extrapolate current findings to industry-scale operations.

The repair layers

To ensure that the neural network output respects the problem constraints, the paper proposes to add a sigmoid, and two repair layers to the end of the network. the sigmoid ensures that the generator output limits are respected, the first layer ensures that the power balance constraint is respected, and the second layer ensures that the reserve requirement is respected. in the paper it is shown that if there is a feasible solution, the repair layers will output a feasible solution.
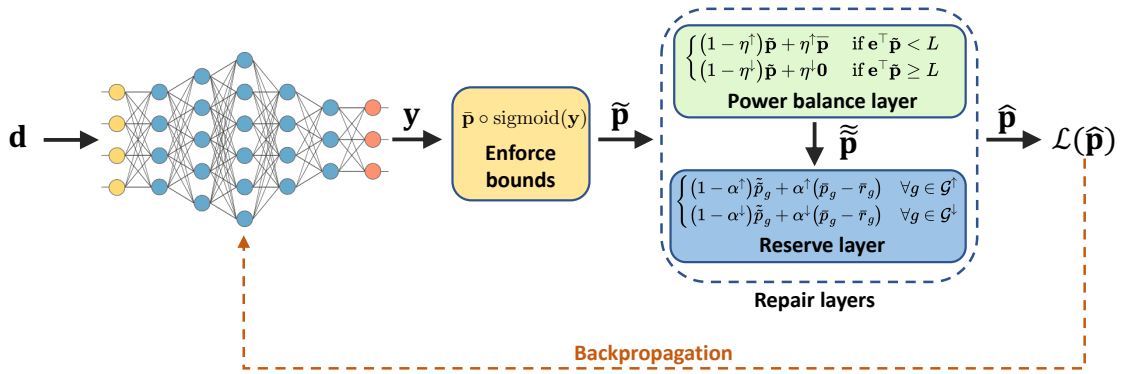
The power balance repair layer $\mathcal{P}$ takes as input an initial dispatch vector $\mathbf{p} \in \mathcal{H}$ and outputs a dispatch vector $\tilde{\mathbf{p}}$ that also satisfies the power balance constraint.
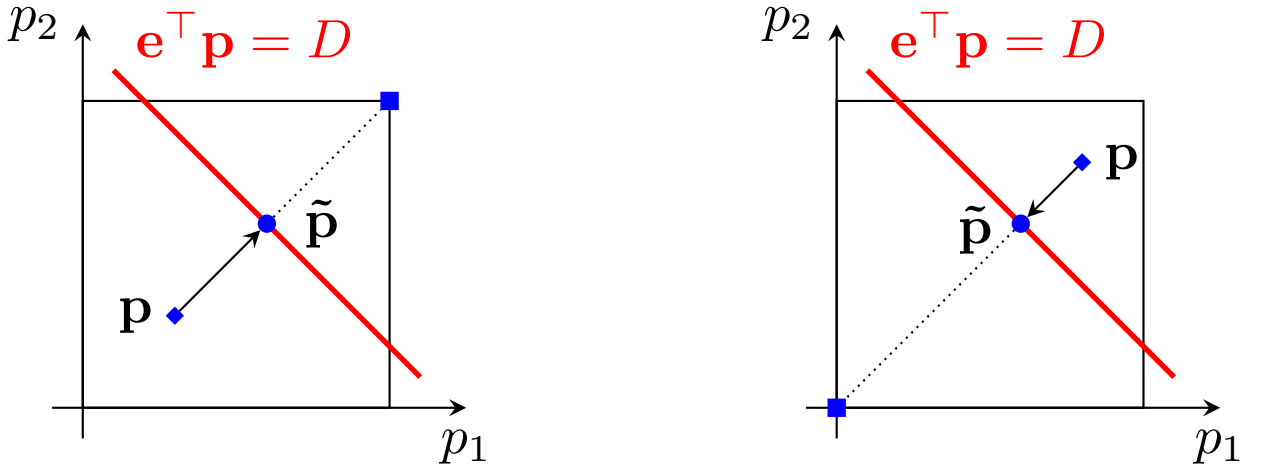

Figure 1: Illustration of the power balance layer with input $\mathbf{p}$ and output $\tilde{\mathbf{p}}$. Left: $\mathbf{e}^\top \mathbf{p} < D$ (energy shortage) and generators' dispatches are increased. Right: $\mathbf{e}^\top \mathbf{p} > D$ (energy surplus) and generators' dispatches are decreased.

the output is given by:

$$\mathcal{P}(\mathbf{p}) = \begin{cases} (1 - \eta^\uparrow)\mathbf{p} + \eta^\uparrow\bar{\mathbf{p}} & \text{if } \mathbf{e}^\top\mathbf{p} < \mathbf{e}^\top\mathbf{d} \\ (1 - \eta^\downarrow)\mathbf{p} + \eta^\downarrow\mathbf{0} & \text{if } \mathbf{e}^\top\mathbf{p} > \mathbf{e}^\top\mathbf{d} \end{cases}$$

$$\eta^\uparrow = \frac{\mathbf{e}^\top\mathbf{d} - \mathbf{e}^\top\mathbf{p}}{\mathbf{e}^\top\bar{\mathbf{p}} - \mathbf{e}^\top\mathbf{p}}, \quad \eta^\downarrow = \frac{\mathbf{e}^\top\mathbf{p} - \mathbf{e}^\top\mathbf{d}}{\mathbf{e}^\top\mathbf{p} - \mathbf{e}^\top\mathbf{0}}$$

The reserve repair layer $\mathcal{R}$ takes as input an initial dispatch vector $\mathbf{p} \in \mathcal{S}_D$ and outputs a dispatch vector $\tilde{\mathbf{p}}$ that also satisfies the reserve requirement.
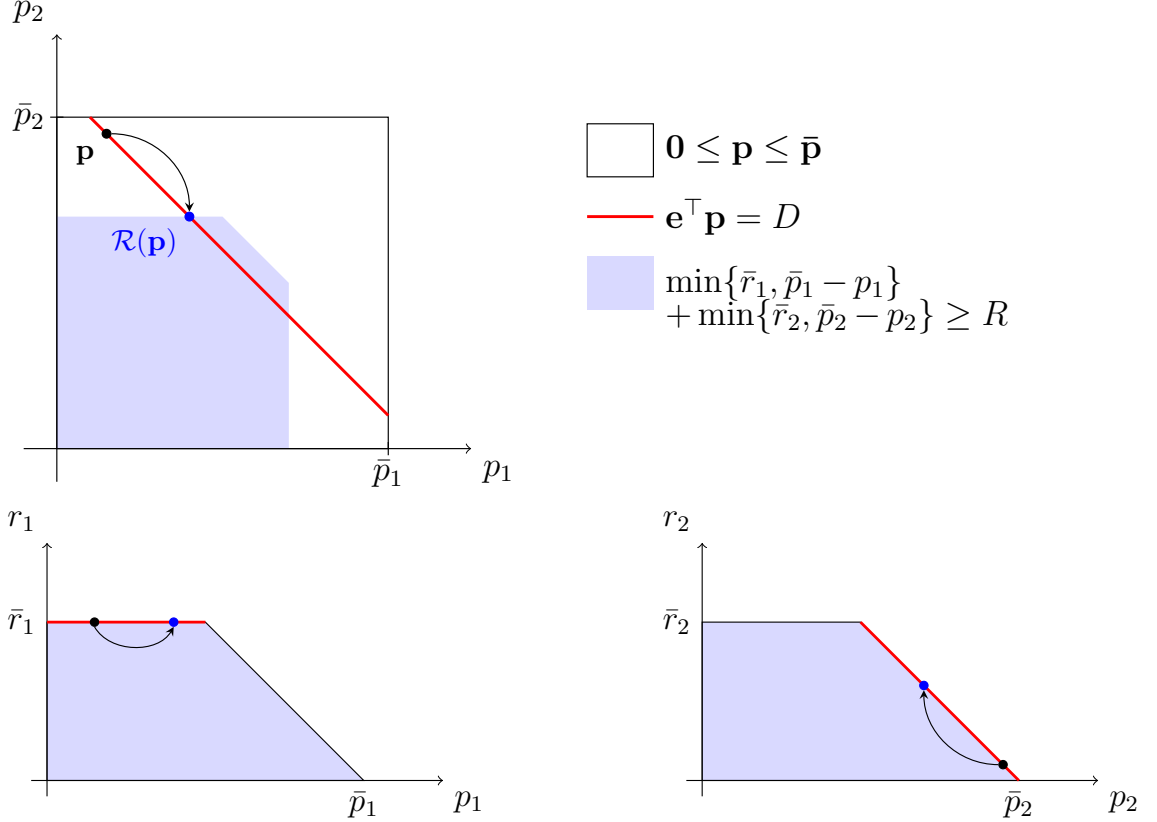

Figure 2: Illustration of the reserve feasibility layer for $\bar{\mathbf{p}}=(1,1)$, $\bar{\mathbf{r}}=(0.5, 0.5)$, $D=1.1$, $R=0.8$ and the initial prediction $\mathbf{p}=(0.15, 0.95)$. The recovered feasible dispatch is $\tilde{\mathbf{p}}=(0.4, 0.7)$. Top: effect of the layer in the (p_1, p_2) space. Bottom: effect of the layer on each generator (individually). The active constraint is shown in red. Generator 1 is in $\mathcal{G}^\uparrow$ and generator 2 is in $\mathcal{G}^\downarrow$.

the output is given by the following algorithm:

Experimental results

Dataset

The 6 datasets are derived from reference test cases in the PGLib library. The authors generated many datapoints by perturbing the reference cases. They created two categories of datasets: ED (Economic Dispatch without reserves) and ED-R (with reserves).

the ground truth solutions were computed using the Gurobi optimizer.

Algorithm 1 Reserve Repair Layer

Require: Initial prediction $\mathbf{p} \in \mathcal{S}_D$, maximum limits $\bar{\mathbf{p}}, \bar{\mathbf{r}}$, reserve requirement R

1: $\Delta_R \leftarrow R - \sum_g \min\{\bar{r}_g, \bar{p}_g - p_g\}$
2: $\mathcal{G}^{\uparrow} \leftarrow \{g \mid p_g \leq \bar{p}_g - \bar{r}_g\}$
3: $\mathcal{G}^{\downarrow} \leftarrow \{g \mid p_g > \bar{p}_g - \bar{r}_g\}$
4: $\Delta^{\uparrow} \leftarrow \sum_{g \in \mathcal{G}^{\uparrow}} (\bar{p}_g - \bar{r}_g) - p_g$
5: $\Delta^{\downarrow} \leftarrow \sum_{g \in \mathcal{G}^{\downarrow}} p_g - (\bar{p}_g - \bar{r}_g)$
6: $\Delta \leftarrow \max(0, \min(\Delta_R, \Delta^{\uparrow}, \Delta^{\downarrow}))$
7: $\alpha^{\uparrow} \leftarrow \Delta/\Delta^{\uparrow}, \; \alpha^{\downarrow} \leftarrow \Delta/\Delta^{\downarrow},$
8: Energy dispatch adjustment

$$\tilde{p}_g = \begin{cases} (1 - \alpha^{\uparrow})p_g + \alpha^{\uparrow}(\bar{p}_g - \bar{r}_g) & \forall g \in \mathcal{G}^{\uparrow} \\ (1 - \alpha^{\downarrow})p_g + \alpha^{\downarrow}(\bar{p}_g - \bar{r}_g) & \forall g \in \mathcal{G}^{\downarrow} \end{cases}$$

9: **return** $\mathcal{R}(\mathbf{p}) = \tilde{\mathbf{p}}$

System Name	Buses ($|\mathcal{N}|$)	Generators ($|\mathcal{G}|$)	Branches ($|\mathcal{E}|$)	Type
ieee300	300	69	411	Academic
pegase1k	1,354	260	1,991	Realistic
rte6470	6,470	761	9,005	Real (French Grid)
pegase9k	9,241	1,445	16,049	Realistic
pegase13k	13,659	4,092	20,467	Realistic
goc30k	30,000	3,526	35,393	Synthetic (Large Scale)

Implementation details

the loss functions for supervised learning and self-supervised learning were:

$$\mathcal{L}^{SL}(\hat{p}, p^*) = \underbrace{\frac{1}{|\mathcal{G}|}||\hat{p} - p^*||_1}_{\text{MAE of Dispatch}} + \underbrace{\mu M_{th}||\xi_{th}(\hat{p})||}_{\text{Thermal Constraints}} + \underbrace{\lambda \psi(\hat{p})}_{\text{Hard Constraints}}$$

$$\mathcal{L}^{SSL}(p^*) = \underbrace{c(p^*)}_{\text{Generation Cost}} + \underbrace{M_{th}\xi_{th}(p^*)}_{\text{Thermal Penalties}} + \underbrace{\lambda \psi(\hat{p})}_{\text{Hard Constraints}}$$

$$\psi(\hat{p}) = M_{pb}|e^T \boldsymbol{d} - e^T \hat{p}| + M_r \xi_r(\hat{p})$$

where the hyperparameters were: $M_{th} = 1500$, $M_{pb} = 3500$, $M_r = 1100$.

The models are simple Feed Forward Neural Networks with a sigmoid activation function and the repair layers. the dataset was split into 40000/5000/5000 for training/validation/test. the optimization was done using the Adam optimizer. the learning rate was reduced by a factor of 10 when the validation loss stopped improving for 10 consecutive epochs. the training was stopped when the validation loss stopped improving for 20 consecutive epochs. the hyperparameters were optimized using grid search.

Results

The results have shown that the E2ELR model preformed better than the other models in terms of accuracy on every dataset. The E2ELR models also took around the same time to infer a solution as the other models.