$\mathcal{P}^2$ LAB
@EE, IIT Roorkee

Optimization Proxies for Power Systems

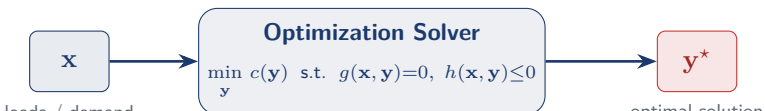
The Idea, Data Efficiency, Feasibility & Confidence

Parikshit Pareek
Assistant Professor
Department of Electrical Engineering, IIT Roorkee
Ph.D., NTU Singapore • Postdoc, Los Alamos National Laboratory

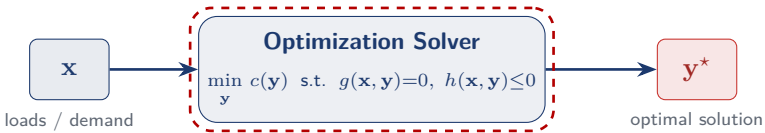
pareek@ee.iitr.ac.in

Invited Talk • Department of Electrical Engineering, IIT Delhi • 10 June 2026

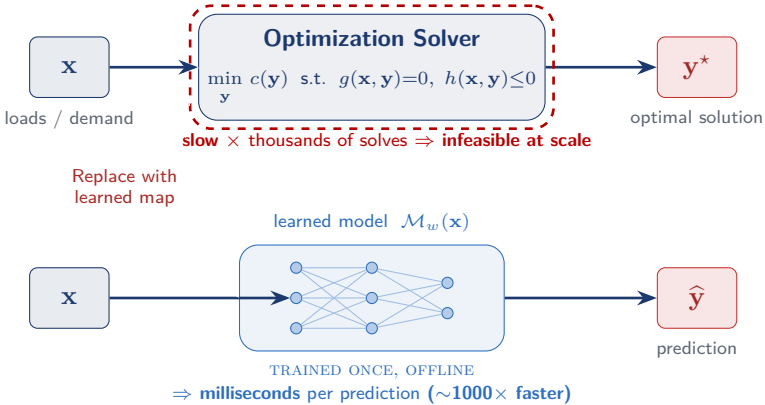
What is an Optimization Proxy?



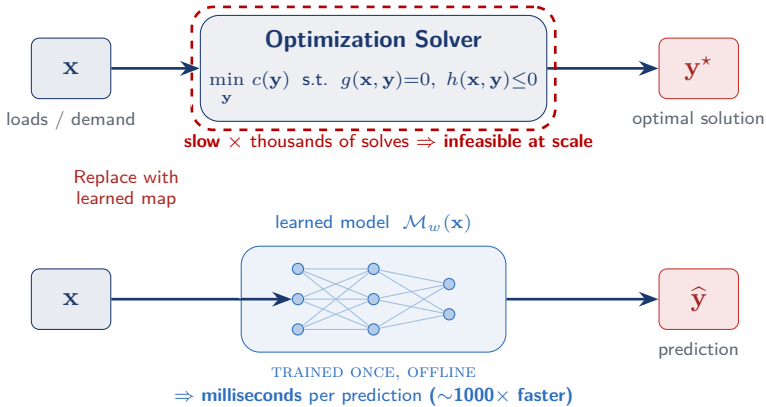
What is an Optimization Proxy?



What is an Optimization Proxy?



What is an Optimization Proxy?



Fast — but is $\widehat{\mathbf{y}}$ **feasible**? **optimal**? **trustworthy**? ⇒

Data Efficiency · **Feasibility** · **Confidence**

Before We Walk Further:

There are Two Ways ML Meets Optimization!

Before We Walk Further:

There are Two Ways ML Meets Optimization!

Optimization problem

$\min_{\mathbf{y}} c(\mathbf{y})$ s.t. $g(\mathbf{x}, \mathbf{y}) = 0,\ h(\mathbf{x}, \mathbf{y}) \leq 0$

THIS TALK

1. Learn the solution

end-to-end optimization proxy

Replace the solver entirely:

$\mathbf{x} \to \mathcal{M}_w(\mathbf{x}) \to \widehat{\mathbf{y}} \approx \mathbf{y}^\star$

Fast, but approximate — must check feasibility & optimality.

2. Learn inside the solver

ML-accelerated exact solving

Keep the solver; learn its choices:

- learn to branch [Gasse '19]
- cut selection [Tang '20]
- node / variable ordering
- warm-starts & primal heuristics

Still **exact** — only **faster**.

Loss Functions Across Proxies: Who Uses What

- Proxy losses range from pure label-matching to fully objective-driven

Loss Functions Across Proxies: Who Uses What

Proxy losses range from pure label-matching to fully objective-driven

Loss Family	Schematic Training Loss	Representative Work	Venue
MSE/MAE + PF recovery	$\left\|\hat{\mathbf{y}} - \mathbf{y}^\star\right\|^2$	Pan et al. (2021)	IEEE TPS'21
Supervised + Lagrangian penalty	$\left\|\hat{\mathbf{y}} - \mathbf{y}^\star\right\|^2 + \boldsymbol{\lambda}^\top \nu(\hat{\mathbf{y}})$	Fioretto et al. (2020b)	AAAI'20
Completion + correction (soft loss)	$c(\hat{\mathbf{y}}) + \lambda \left\|\text{ReLU}[h]\right\|^2$	Donti et al. (2021)	ICLR'21
Self-supervised primal–dual	$c(\hat{\mathbf{y}}) + \hat{\boldsymbol{\lambda}}^\top g + \hat{\boldsymbol{\mu}}^\top h + \frac{\rho}{2}\nu(\cdot)$	Park and Van Hentenryck (2023b)	AAAI'23
Unsupervised (objective + penalty)	$c(\hat{\mathbf{y}}) + \lambda\, \mathcal{V}(\hat{\mathbf{y}})$	Huang et al. (2024)	IEEE TPS'24
Physics-informed (KKT)	$\left\|\hat{\mathbf{y}} - \mathbf{y}^\star\right\|^2 + \lambda \left\|\text{KKT res.}\right\|$	Nellikkath and Chatzivasileiadis (2022a)	EPSR'22

Trend

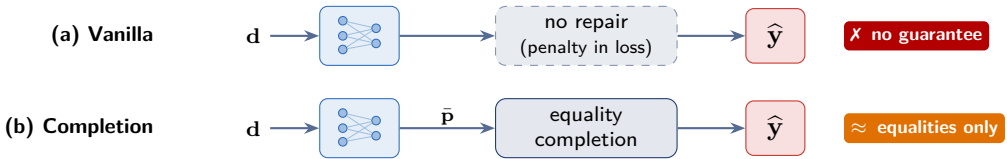
Move from **label-hungry** MSE → **label-free** objective+penalty;

How Proxies Enforce Feasibility: An Architecture Spectrum

(a) Vanilla

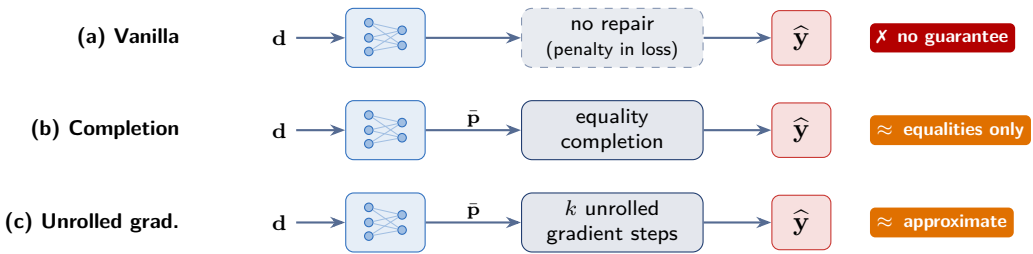
✗ no guarantee ≈ partial / approximate ✓ guaranteed feasible ⇒ **(e)** repairs feasibility *by construction*, at scale.

How Proxies Enforce Feasibility: An Architecture Spectrum



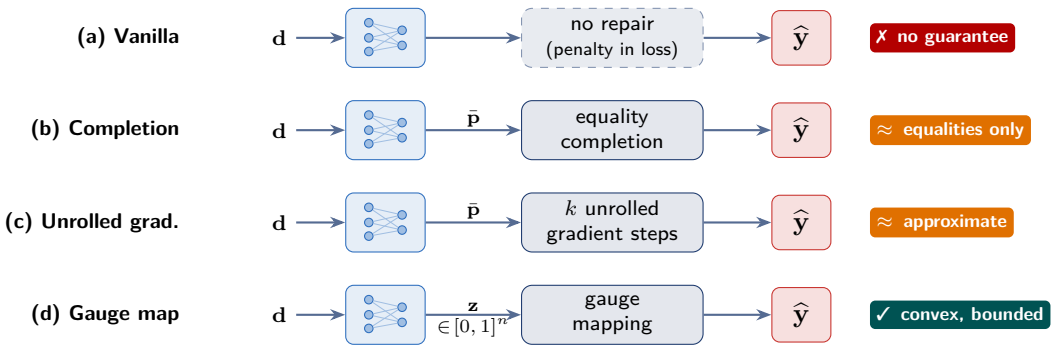
✗ no guarantee ≈ partial / approximate ✓ guaranteed feasible ⇒ **(e)** repairs feasibility *by construction*, at scale.

How Proxies Enforce Feasibility: An Architecture Spectrum



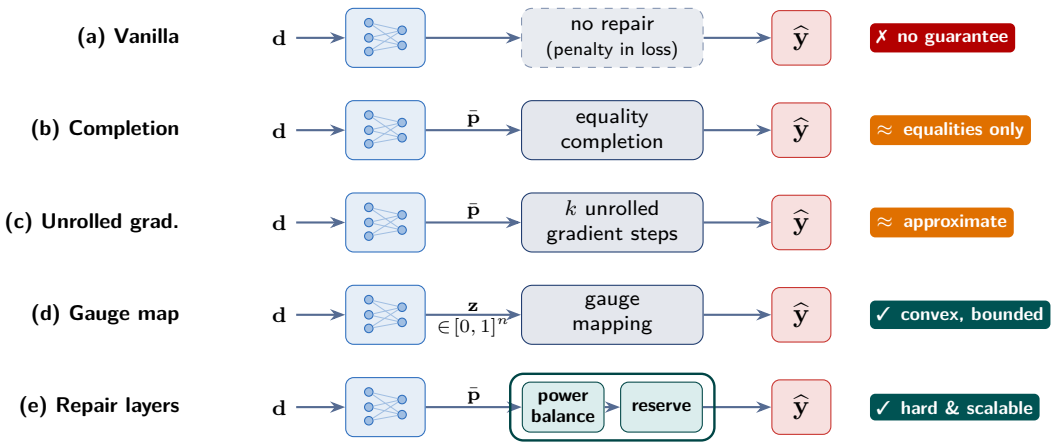
✗ no guarantee ≈ partial / approximate ✓ guaranteed feasible ⇒ **(e)** repairs feasibility *by construction*, at scale.

How Proxies Enforce Feasibility: An Architecture Spectrum



✗ no guarantee ≈ partial / approximate ✓ guaranteed feasible ⇒ **(e)** repairs feasibility *by construction*, at scale.

How Proxies Enforce Feasibility: An Architecture Spectrum



✗ no guarantee ≈ partial / approximate ✓ guaranteed feasible ⇒ **(e)** repairs feasibility *by construction*, at scale.

The ML's Time Equation

$$T_{Total} = T_{Data} + T_{Training} + T_{Prediction}$$

T_{Data}	$T_{Training}$	$T_{Prediction}$
Training Data Generation Validation Data Generation	Hyperparameter Optimization Variational Inference	Prediction Time Validation Time

The Problem

How to Overcome Constraints on Training Time and Data?

Time and Data Constraints

- **Optimization proxies must be trained efficiently within strict time constraints**
  - Operational time constraints
  - Speed requirements for large-scale simulations

Time and Data Constraints

- **Optimization proxies must be trained efficiently within strict time constraints**
  - Operational time constraints
  - Speed requirements for large-scale simulations
- **Obtaining training data is time-consuming or not feasible**
  - Large-scale problem solutions produce only a single data point
  - Limited data related to rare or less probable events

Time and Data Constraints

- **Optimization proxies must be trained efficiently within strict time constraints**
  - Operational time constraints
  - Speed requirements for large-scale simulations
- **Obtaining training data is time-consuming or not feasible**
  - Large-scale problem solutions produce only a single data point
  - Limited data related to rare or less probable events
- **Physical systems require 'some' guarantees on learning quality**
  - Vanilla statistical validation demands a large validation dataset

    `Hoeffding's Inequality:` $\varepsilon \propto 1/\sqrt{N}$

Bayesian for Rescue under Low Data

- **What are Bayesian Neural Networks (BNNs)?**
  - BNNs treat weights as probability distributions instead of fixed values
  - Provide a probability distribution over outputs instead of single deterministic predictions.

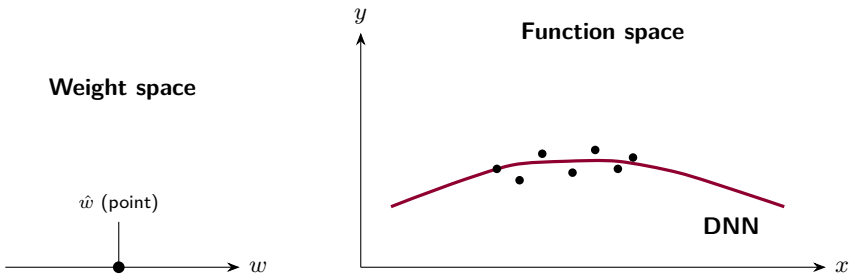
Bayesian for Rescue under Low Data

- **What are Bayesian Neural Networks (BNNs)?**
  - BNNs treat weights as probability distributions instead of fixed values
  - Provide a probability distribution over outputs instead of single deterministic predictions.
- **Why are BNNs better than DNNs for low data?**
  - Incorporate **prior knowledge** and effectively model uncertainty in data and parameters.
  - Prevent overfitting: **Weight Distribution**, **Regularization via Priors**, **Posterior Averaging**

Bayesian for Rescue under Low Data

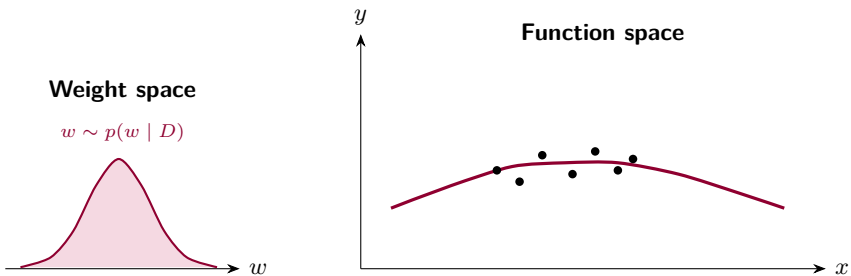
- **What are Bayesian Neural Networks (BNNs)?**
  - BNNs treat weights as probability distributions instead of fixed values
  - Provide a probability distribution over outputs instead of single deterministic predictions.
- **Why are BNNs better than DNNs for low data?**
  - Incorporate **prior knowledge** and effectively model uncertainty in data and parameters.
  - Prevent overfitting: **Weight Distribution**, **Regularization via Priors**, **Posterior Averaging**
- **Advantages over DNNs:**
  - Separates two types of uncertainty:
    - **Epistemic uncertainty**: Uncertainty in model parameters $p(w|D)$.
    - **Aleatoric uncertainty**: Noise inherent in data $p(y|x, w)$.
  - Assigns high uncertainty to points far from the training set, avoiding overconfident predictions.

From DNN to BNN: One Picture



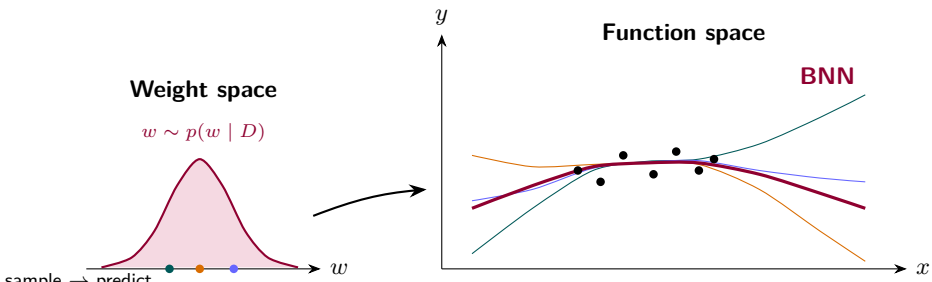
DNN: fixed weights $\Rightarrow$ *one* function — confident even where there is no data.

From DNN to BNN: One Picture



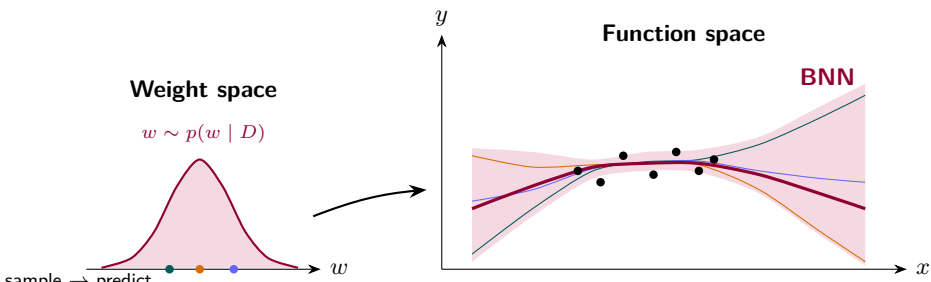
BNN: each weight is a distribution, $w \sim p(w \mid D)$ — not a single value.

From DNN to BNN: One Picture



Sampling weights ⇒ an **ensemble** of functions: they agree on data, disagree away from it.

From DNN to BNN: One Picture



Posterior averaging ⇒ mean ± uncertainty. The band **widens far from data**.

Our Problem, Target, Motivation and Idea

Standard Constrained Optimization Problem

$$\min_{\mathbf{y}} \quad c(\mathbf{y}) \tag{1a}$$

$$\text{s.t.} \quad g(\mathbf{x}, \mathbf{y}) = 0 \tag{1b}$$

$$h(\mathbf{x}, \mathbf{y}) \leqslant 0 \tag{1c}$$

$\mathbf{x}$ is given (input vector)

Our Problem, Target, Motivation and Idea

- **Standard Constrained Optimization Problem**

$$\min_{\mathbf{y}} \quad c(\mathbf{y}) \tag{1a}$$

$$\text{s.t.} \quad g(\mathbf{x}, \mathbf{y}) = 0 \tag{1b}$$

$$h(\mathbf{x}, \mathbf{y}) \leqslant 0 \tag{1c}$$

$\mathbf{x}$ is given (input vector)

- **Target:**
  Develop a **Fast Evaluating** proxy such that: $\mathbf{y} = \mathcal{M}_w(\mathbf{x})$

Our Problem, Target, Motivation and Idea

- **Standard Constrained Optimization Problem**

$$\min_{\mathbf{y}} \quad c(\mathbf{y}) \tag{1a}$$

$$\text{s.t.} \quad g(\mathbf{x}, \mathbf{y}) = 0 \tag{1b}$$

$$h(\mathbf{x}, \mathbf{y}) \leqslant 0 \tag{1c}$$

$\mathbf{x}$ is given (input vector)

- **Target:**
  Develop a **Fast Evaluating** proxy such that: $\mathbf{y} = \mathcal{M}_w(\mathbf{x})$
- **Motivation:**
  Training-validation data collection is **Expensive** & need to **Train** $\mathcal{M}_w(\mathbf{x})$ **Fast**

Our Problem, Target, Motivation and Idea

- **Standard Constrained Optimization Problem**

$$\min_{\mathbf{y}} \quad c(\mathbf{y}) \tag{1a}$$
$$\text{s.t.} \quad g(\mathbf{x}, \mathbf{y}) = 0 \tag{1b}$$
$$h(\mathbf{x}, \mathbf{y}) \leqslant 0 \tag{1c}$$
$$\mathbf{x} \text{ is given (input vector)}$$

- **Target:**
  Develop a **Fast Evaluating** proxy such that: $\mathbf{y} = \mathcal{M}_w(\mathbf{x})$
- **Motivation:**
  Training-validation data collection is **Expensive** & need to **Train** $\mathcal{M}_w(\mathbf{x})$ **Fast**
- **Idea:**
  Using large unlabeled dataset to enforce **Feasibility** & limited supervised dataset for **Optimality + Feasibility**

The TL;DR: An Optimal Solution Has To Be Feasible

Feasibility Condition

$$\mathcal{F}(\mathbf{y}, \mathbf{x}) = \lambda_e \underbrace{\left\| g(\mathbf{x}, \mathbf{y}) \right\|^2}_{\texttt{Equality Gap}} + \lambda_i \underbrace{\left\| \mathsf{ReLU}[h(\mathbf{x}, \mathbf{y})] \right\|^2}_{\texttt{Inequality Gap}}. \tag{2}$$

Feasibility Condition on Network Weights

Network weights should be such that, for any input, predicted output provides $\mathcal{F}(\mathbf{y}, \mathbf{x}) = 0$

If input sampling is cheap, we can create an **Augmented Labeled Dataset** for **Free**

$$\mathcal{D}^f = \{(\mathbf{x}_j, \mathcal{F}(\cdot, \mathbf{x}) = 0)\}_{j=1}^{M}$$

Sandwich Model

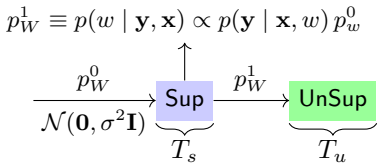
Alternating Training Phases for BNNs in Time-Constrained Bursts

$$p_W^1 \equiv p(w \mid \mathbf{y}, \mathbf{x}) \propto p(\mathbf{y} \mid \mathbf{x}, w)\, p_w^0$$

$$\xrightarrow[\mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})]{p_W^0} \underbrace{\text{Sup}}_{T_s}$$

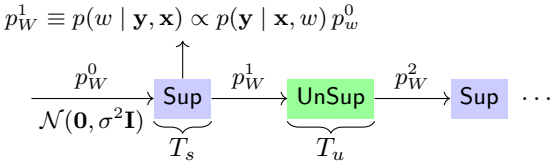
Sandwich Model

Alternating Training Phases for BNNs in Time-Constrained Bursts



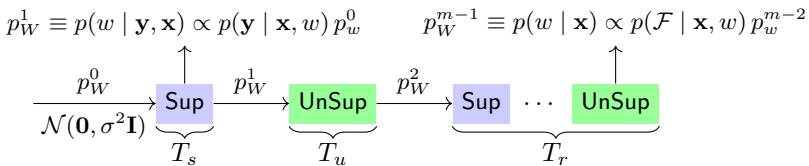
Sandwich Model

Alternating Training Phases for BNNs in Time-Constrained Bursts



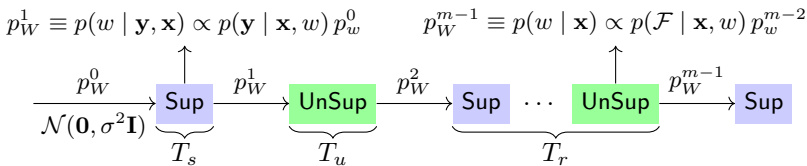
Sandwich Model

Alternating Training Phases for BNNs in Time-Constrained Bursts



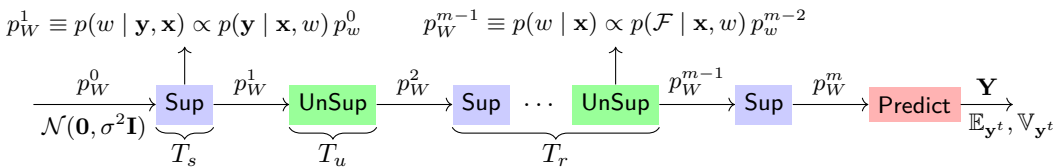
Sandwich Model

Alternating Training Phases for BNNs in Time-Constrained Bursts



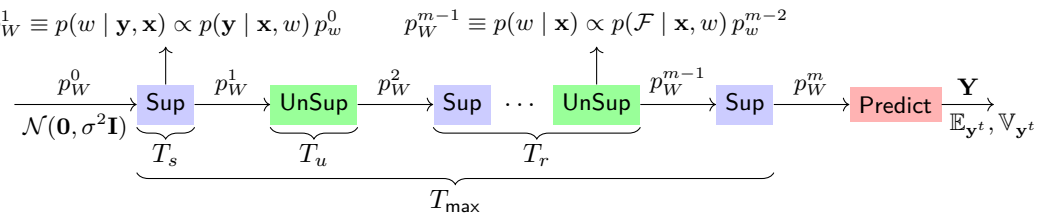
Sandwich Model

Alternating Training Phases for BNNs in Time-Constrained Bursts



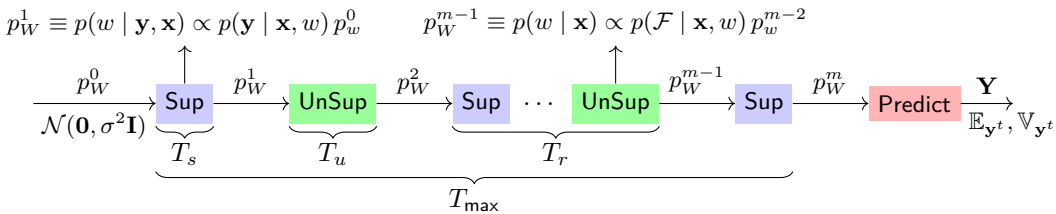
Sandwich Model

Alternating Training Phases for BNNs in Time-Constrained Bursts



Sandwich Model

Alternating Training Phases for BNNs in Time-Constrained Bursts



➠ `Selection via Posterior`: Among BNN weights, select the one providing **'Best'** results for the selected criteria

$$W^\star = \arg\min_j \left[\max_i \left| g_i(\mathbf{x}^t, \mathbf{Y}_{\cdot j}) \right| \right] : \quad \text{Minimizes the maximum equality gap}$$

Bounding the Error

- We want to know, how far expected error can be compared to the error we calculated!

$$\mathbb{P}\left\{\left|\mathbb{E}[|e|] - \frac{1}{M}\sum_{i=1}^{M}|e_i|\right| \leqslant \varepsilon\right\} \geqslant 1-\delta$$

- Error bound ε in PCBs, provided by different concentration inequalities.

Hoeffding's	Empirical Bernstein	Theoretical Bernstein
$R\sqrt{\frac{\log(2/\delta)}{2M}}$	$\sqrt{\frac{2\widehat{\mathbb{V}}_e \log(3/\delta)}{M}} + \frac{3R\log(3/\delta)}{M}$	$\sqrt{\frac{2\mathbb{V}_e \log(1/\delta)}{M}} + \frac{2R\log(1/\delta)}{3M}$

- `Hypothesis:` $\alpha\,\mathsf{MPV} \geqslant \mathbb{V}_e = \underbrace{\mathbb{E}_M\left[\mathbb{V}_W[e]\right] + \mathbb{E}_W\left[\mathbb{V}_M[e]\right]}_{\texttt{Total Variance in Error}} \geqslant \mathbb{V}_{|e|}$
- `Proposed Bound:`

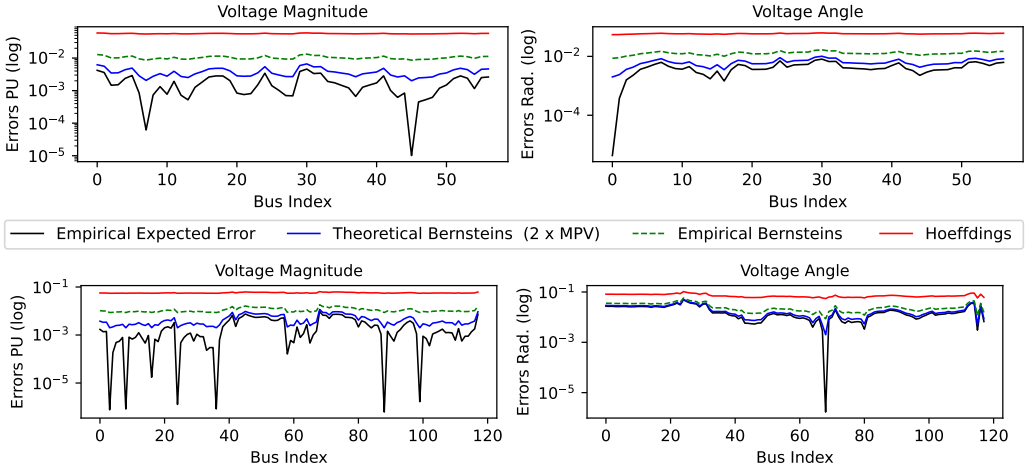
$$\sqrt{\frac{2(2\times \mathsf{MPV})\log(1/\delta)}{M}} + \frac{2R\log(1/\delta)}{3M}$$

Matchup: Sandwich, BNN & DNN

Table: Comparative performance results for the ACOPF Problem for **'case118'** with 512 labeled training samples, 2048 unlabeled samples, and $T_{\text{max}} = 600$ sec.

Method	Gap%	Max Eq.	Mean Eq.	Max Ineq.	Mean Ineq.
Sandwich BNN SvP (Ours)	**1.484**	**0.089**	0.018	0.008	**0.000**
Sandwich BNN (Ours)	1.485	0.100	**0.016**	0.008	0.000
Supervised BNN SvP (Ours)	1.568	0.147	0.022	0.013	0.000
Supervised BNN (Ours)	1.567	0.205	0.020	0.013	0.000
Naïve MAE	1.638	2.166	0.187	**0.000**	0.000
Naïve MSE	1.622	3.780	0.242	0.000	0.000
MAE + Penalty	1.577	1.463	0.102	0.000	0.000
MSE + Penalty	1.563	2.637	0.125	0.000	0.000
LD + MAE	1.565	1.284	0.083	0.000	0.000

Probabilistic Bounds



➧ Theoretical Bernstein bounds with 2 $\times$MPV are tightest.

We consider $\delta = 0.95$ and 1000 *out-of-sample* testing data points i.e. $M = 1000$

Conclusions

- **The Idea:** learned proxies replace slow solvers with millisecond predictions ($\sim$1000$\times$ faster) — for probabilistic OPF, $N-1$ screening and planning at scale.
- **Data Efficiency:** Bayesian NNs with the *Sandwich* schedule reach solver-quality solutions under tight label and training-time budgets; cheap unlabeled inputs enforce feasibility for free.
- **Feasibility:** an optimal solution *must* be feasible — feasibility-by-construction layers turn 'no guarantee' into 'guaranteed feasible'.
- **Confidence:** distribution-free concentration bounds — Theoretical Bernstein with $2\times$MPV — give the tightest probabilistic error guarantees.

Learning to Solve, Learning to Trust

P. Pareek, A. Jayakumar, K. Sundar, S. Misra, and D. Deka. Optimization proxies using limited labeled data and training time – a semi-supervised Bayesian neural network approach. ICML 2025.

References I

Y. Bengio, A. Lodi, and A. Prouvost. Machine learning for combinatorial optimization: A methodological tour d'horizon. *European Journal of Operational Research*, 290(2):405–421, 2021.

M. Chatzos, T. W. Mak, and P. Van Hentenryck. Spatial network decomposition for fast and scalable AC-OPF learning. *IEEE Transactions on Power Systems*, 37(4):2601–2612, 2022. doi: 10.1109/TPWRS.2021.3124726.

W. Chen, S. Park, M. Tanneau, and P. Van Hentenryck. Learning optimization proxies for large-scale security-constrained economic dispatch. *Electric Power Systems Research*, 213:108566, 2022. doi: 10.1016/j.epsr.2022.108566.

W. Chen, M. Tanneau, and P. Van Hentenryck. End-to-end feasible optimization proxies for large-scale economic dispatch. *IEEE Transactions on Power Systems*, 39(2):4723–4734, 2024. doi: 10.1109/TPWRS.2023.3317352.

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.

F. Fioretto, T. W. Mak, and P. Van Hentenryck. Predicting AC optimal power flows: Combining deep learning and Lagrangian dual methods. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 630–637, 2020a.

F. Fioretto, T. W. Mak, and P. Van Hentenryck. Predicting AC optimal power flows: Combining deep learning and Lagrangian dual methods. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 34, pages 630–637, 2020b. doi: 10.1609/aaai.v34i01.5403.

M. Gasse, D. Chételat, N. Ferroni, L. Charlin, and A. Lodi. Exact combinatorial optimization with graph convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

W. Huang and M. Chen. DeepOPF-NGT: Fast no ground truth deep learning-based approach for AC-OPF problems. In *ICML 2021 Workshop on Tackling Climate Change with Machine Learning*, 2021.

W. Huang, M. Chen, and S. H. Low. Unsupervised learning for solving AC optimal power flows: Design, analysis, and experiment. *IEEE Transactions on Power Systems*, 39(6):7102–7114, 2024.

References II

E. B. Khalil, P. Le Bodic, L. Song, G. Nemhauser, and B. Dilkina. Learning to branch in mixed integer programming. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

M. Li, S. Kolouri, and J. Mohammadi. Learning to solve optimization problems with hard linear constraints. *IEEE Access*, 11: 59995–60004, 2023. doi: 10.1109/ACCESS.2023.3285199.

V. Nair, S. Bartunov, F. Gimeno, I. von Glehn, P. Lichocki, I. Lobov, B. O'Donoghue, N. Sonnerat, C. Tjandraatmadja, P. Wang, et al. Solving mixed integer programs using neural networks. *arXiv preprint arXiv:2012.13349*, 2020.

R. Nellikkath and S. Chatzivasileiadis. Physics-informed neural networks for AC optimal power flow. *Electric Power Systems Research*, 212:108412, 2022a. doi: 10.1016/j.epsr.2022.108412.

R. Nellikkath and S. Chatzivasileiadis. Physics-informed neural networks for AC optimal power flow. *Electric Power Systems Research*, 212:108412, 2022b. doi: 10.1016/j.epsr.2022.108412.

Y. Ng, S. Misra, L. A. Roald, and S. Backhaus. Statistical learning for DC optimal power flow. In *2018 Power Systems Computation Conference (PSCC)*, pages 1–7, 2018.

X. Pan, T. Zhao, M. Chen, and S. Zhang. DeepOPF: A deep neural network approach for security-constrained DC optimal power flow. *IEEE Transactions on Power Systems*, 36(3):1725–1735, 2021. doi: 10.1109/TPWRS.2020.3026379.

P. Pareek, A. Jayakumar, K. Sundar, S. Misra, and D. Deka. Optimization proxies using limited labeled data and training time – a semi-supervised Bayesian neural network approach. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, 2025. arXiv:2410.03085.

S. Park and P. Van Hentenryck. Self-supervised primal-dual learning for constrained optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 4052–4060, 2023a.

S. Park and P. Van Hentenryck. Self-supervised primal-dual learning for constrained optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 37, pages 4052–4060, 2023b. doi: 10.1609/aaai.v37i4.25520.

References III

Y. Tang, S. Agrawal, and Y. Faenza. Reinforcement learning for integer programming: Learning to cut. In *Proceedings of the 37th International Conference on Machine Learning (ICML)*, volume 119 of *PMLR*, pages 9367–9376, 2020.

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)*, pages 1–6, 2020a.

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)*, pages 1–6, 2020b.

$\mathcal{P}^2$-LAB: Learning to Solve, Learning to Trust

LEARN

Amortized Inference & Optimization Proxies

One Forward Pass → Answer

VERIFY

Probabilistic guarantees & mechanistic understanding

Guarantees, not just guesses

DEPLOY

Power Grids, EV & DER, forecasting and planning

AI4Power

$\mathcal{P}^2$ LAB

@EE, IIT Roorkee

Scan to connect & collaborate

PhD POSITION – OPENING SOON

A **US industry-funded** PhD on **Optimization Proxies**.
Stipend ≈ **INR 60,000 / month**.
Watch this space — Write to `pareek@ee.iitr.ac.in`.