ECE 598 Computational Power Systems

Decision-focused analytics

Vladimir Dvorkin

University of Michigan

Last lecture recap

- Look around you and form teams of 2 people (1 min)
- Quickly review your notes or the slide deck (1 min)
- Share your **three** personal highlights with your partner (3 min)
- Get iClicker app ready

Closed-loop analytics

Recap: Why forecasting?

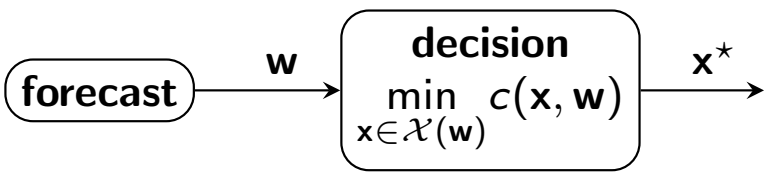
- Forecasting is the first step in **decision-making**
- Resolves (some) uncertainty of **decision-making** inputs
- Brings confidence to **decision-making**

Closed-loop analytics

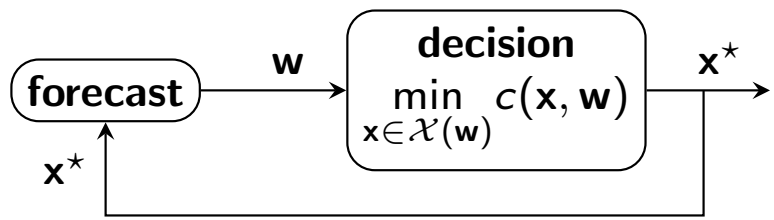
Recap: Why forecasting?

- Forecasting is the first step in **decision-making**
- Resolves (some) uncertainty of **decision-making** inputs
- Brings confidence to **decision-making**

open-loop analytics



closed-loop analytics



Decision feedback of the closed-loop analytics

- Inform forecast on the quality of decisions induced on that forecast
- Requires integration of forecast and decision-making optimization problems

Example: Wind power forecast & electricity pricing

Given a wind power forecast $\widehat{\mathbf{w}}$, solve the DC-OPF problem:

$$
\begin{aligned}
\underset{\underline{\mathbf{p}} \leqslant \mathbf{p} \leqslant \overline{\mathbf{p}}}{\text{minimize}} \quad & \mathbf{p}^\top \mathbf{C}\mathbf{p} + \mathbf{c}^\top \mathbf{p} && \textit{generator dispatch cost} \\
\text{subject to} \quad & \mathbf{1}^\top (\mathbf{p} + \widehat{\mathbf{w}} - \mathbf{d}) = 0 \ : \boldsymbol{\lambda}, && \textit{power balance condition} \\
& |\mathbf{F}(\mathbf{p} + \widehat{\mathbf{w}} - \mathbf{d})| \leqslant \overline{\mathbf{f}} \ : \boldsymbol{\lambda}_{\overline{\mathbf{f}}}, \boldsymbol{\lambda}_{\underline{\mathbf{f}}}, && \textit{power flow limits}
\end{aligned}
$$

Example: Wind power forecast & electricity pricing

Given a wind power forecast $\widehat{\mathbf{w}}$, solve the DC-OPF problem:

$$
\begin{aligned}
\underset{\underline{\mathbf{p}} \leqslant \mathbf{p} \leqslant \overline{\mathbf{p}}}{\text{minimize}} \quad & \mathbf{p}^\top \mathbf{C}\mathbf{p} + \mathbf{c}^\top \mathbf{p} && \textit{generator dispatch cost} \\
\text{subject to} \quad & \mathbf{1}^\top (\mathbf{p} + \widehat{\mathbf{w}} - \mathbf{d}) = 0 \ : \boldsymbol{\lambda}, && \textit{power balance condition} \\
& |\mathbf{F}(\mathbf{p} + \widehat{\mathbf{w}} - \mathbf{d})| \leqslant \overline{\mathbf{f}} \ : \boldsymbol{\lambda}_{\overline{\mathbf{f}}}, \boldsymbol{\lambda}_{\underline{\mathbf{f}}}, && \textit{power flow limits}
\end{aligned}
$$

Location marginal prices (LMPs) are derived from the dual solution:

$$
\boldsymbol{\pi}(\widehat{\mathbf{w}}) = \underbrace{\lambda_b \cdot \mathbf{1}}_{\text{uniform price}} - \underbrace{\mathbf{F}^\top (\boldsymbol{\lambda}_{\overline{\mathbf{f}}} - \boldsymbol{\lambda}_{\underline{\mathbf{f}}})}_{\text{adjustment due to congestion}}
$$

which are unique w.r.t forecast $\widehat{\mathbf{w}}$ under reasonable assumptions!

Example: Wind power forecast & electricity pricing

Given a wind power forecast $\widehat{\mathbf{w}}$, solve the DC-OPF problem:

$$\begin{aligned}
\underset{\underline{\mathbf{p}} \leqslant \mathbf{p} \leqslant \overline{\mathbf{p}}}{\text{minimize}} \quad & \mathbf{p}^\top \mathbf{C}\mathbf{p} + \mathbf{c}^\top \mathbf{p} && \textit{generator dispatch cost} \\
\text{subject to} \quad & \mathbf{1}^\top (\mathbf{p} + \widehat{\mathbf{w}} - \mathbf{d}) = 0 \ : \boldsymbol{\lambda}, && \textit{power balance condition} \\
& |\mathbf{F}(\mathbf{p} + \widehat{\mathbf{w}} - \mathbf{d})| \leqslant \overline{\mathbf{f}} \ : \boldsymbol{\lambda}_{\overline{\mathbf{f}}}, \boldsymbol{\lambda}_{\underline{\mathbf{f}}}, && \textit{power flow limits}
\end{aligned}$$

Location marginal prices (LMPs) are derived from the dual solution:

$$\boldsymbol{\pi}(\widehat{\mathbf{w}}) = \underbrace{\boldsymbol{\lambda}_b \cdot \mathbf{1}}_{\text{uniform price}} - \underbrace{\mathbf{F}^\top (\boldsymbol{\lambda}_{\overline{\mathbf{f}}} - \boldsymbol{\lambda}_{\underline{\mathbf{f}}})}_{\text{adjustment due to congestion}}$$

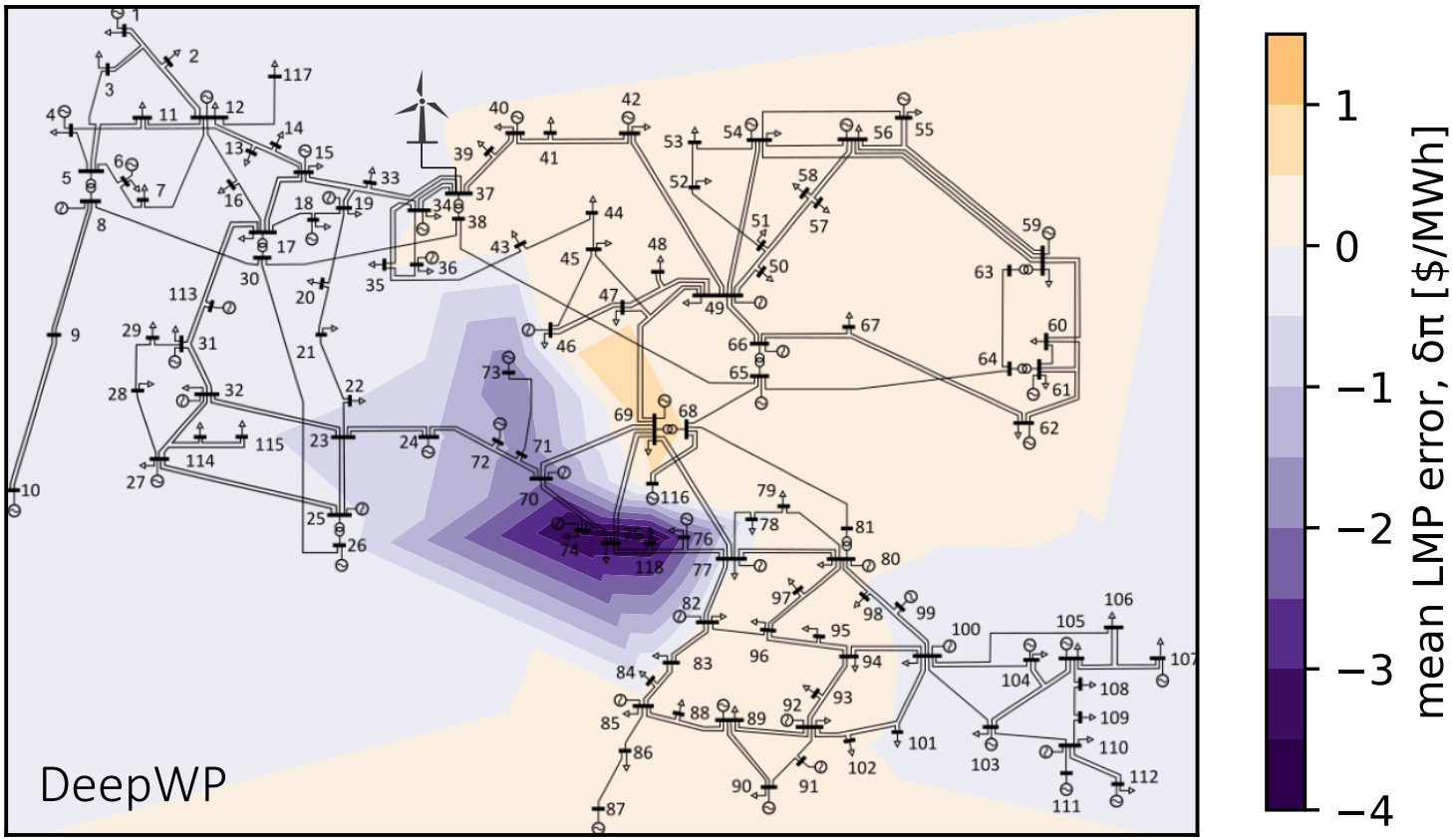
which are unique w.r.t forecast $\widehat{\mathbf{w}}$ under reasonable assumptions!

The **LMP error** is then defined as:

$$\delta\pi = \pi(\widehat{\mathbf{w}}) - \pi(\mathbf{w})$$

gap between LMPs induced on forecast ($\widehat{\mathbf{w}}$) and actual realization ($\mathbf{w}$) of wind power.

Example: Wind power forecast & electricity pricing (cont'd)



- IEEE 118-Bus system with one installed wind farm (bus 39)
- We forecast wind power output using weather forecast ($\times 2$ forecast error)
- LMP errors varying across the grid due to congestion (**Q1.** Is it fair?)
- The slack bus has the smallest LMP error (**Q2.** Why?)

Integrating forecasting and decision-making

Forecast problem

- Dataset $\mathcal{D} = \{(\boldsymbol{\varphi}_1, \mathbf{w}_1), \ldots, (\boldsymbol{\varphi}_k, \mathbf{w}_k)\}$ with features $\boldsymbol{\varphi}$ and observations $\mathbf{w}$
- Machine learning model $\mathbb{W}_{\boldsymbol{\theta}} : \mathcal{F} \mapsto \mathcal{W}$ maps features to predictions, i.e.,

$$\mathbb{W}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{\varphi}}) = \widehat{\mathbf{w}},$$

  where $\boldsymbol{\theta}$ is a model parameter to be optimized

- **Prediction-focused** learning:

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \frac{1}{2k} \sum_{i=1}^{k} \|\mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i) - \mathbf{w}_i\|_2^2 \,.$$

Decision problem

$\mathbf{w}$−parametrized optimization problem:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & c(\mathbf{x}) \\ \text{subject to} \quad & g(\mathbf{x}, \mathbf{w}) \leqslant \mathbf{0} \end{aligned}$$

- $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ is the vector of decision variable
- $\mathbf{w} \in \mathbb{R}^m$ is the input parameter (e.g., forecast from $\mathbb{W}_{\boldsymbol{\theta}}$)
- Objective function c and constraint function g

Decision is the optimal solution $\mathbf{x}^\star(\mathbf{w})$, parametrized by $\mathbf{w}$.

Decision problem

$\mathbf{w}$−parametrized optimization problem:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & c(\mathbf{x}) \\ \text{subject to} \quad & g(\mathbf{x}, \mathbf{w}) \leqslant \mathbf{0} \end{aligned}$$

- $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ is the vector of decision variable
- $\mathbf{w} \in \mathbb{R}^m$ is the input parameter (e.g., forecast from $\mathbb{W}_{\boldsymbol{\theta}}$)
- Objective function c and constraint function g

Decision is the optimal solution $\mathbf{x}^\star(\mathbf{w})$, parametrized by $\mathbf{w}$.

Decision regret is the difference of two objective function values

$$c(\mathbf{x}^\star(\widehat{\mathbf{w}})) - c(\mathbf{x}^\star(\mathbf{w}))$$

with prediction $\widehat{\mathbf{w}}$ and the actual realization $\mathbf{w}$

Decision-focused learning

- Minimize the average decision regret across k historical records
- This calls for the following **bilevel** optimization structure

$$
\begin{aligned}
\underset{\boldsymbol{\theta}}{\text{minimize}} \quad & \frac{1}{2k}\sum_{i=1}^{k} \| c(\mathbf{x}^{\star}(\widehat{\mathbf{w}}_i)) - c(\mathbf{x}^{\star}(\mathbf{w}_i)) \|_2^2 \\
\text{subject to} \quad & \widehat{\mathbf{w}}_i = \mathbb{W}_{\boldsymbol{\theta}}(\varphi_i) \qquad \text{upper level} \qquad \forall i = 1, \dots, k \\
& \text{lower level} \\
& \mathbf{x}^{\star}(\widehat{\mathbf{w}}_i) = \underset{\mathbf{x}}{\text{argmin}} \quad c(\mathbf{x}) \\
& \qquad\qquad \text{subject to} \quad g(\mathbf{x}, \widehat{\mathbf{w}}_i) \leqslant \mathbf{0} \qquad \forall i = 1, \dots, k
\end{aligned}
$$

- Upper-level: optimize the parameter of the ML model $\boldsymbol{\theta}$
- Lower-level: respond with decision $\mathbf{x}$ to parameter $\boldsymbol{\theta}$ for each record i
- **Q1.** How is it different from the prediction-focused optimization?
- **Q2.** How to solve this bilevel optimization?

KKT-based reformulation

The **Karush–Kuhn–Tucker conditions (KKTs)** of the lower-level problem:

1. Primal feasibility: $g(\mathbf{x}^\star, \widehat{\mathbf{w}}_i) \leqslant \mathbf{0}$
2. Dual feasibility: $\boldsymbol{\lambda}^\star \succeq \mathbf{0}$
3. Lagrangian optimality:

$$\nabla_{\mathbf{x}} c(\mathbf{x}^\star) + \boldsymbol{\lambda}^{\star\top} \nabla_{\mathbf{x}} g(\mathbf{x}^\star, \widehat{\mathbf{w}}_i) = \mathbf{0}$$

4. Complementary slackness: $\boldsymbol{\lambda}^{\star\top} g(\mathbf{x}^\star, \widehat{\mathbf{w}}_i) = 0$

For convex optimization, the KKTs are necessary and sufficient for optimality

Thus, we can replace the lower-level problem with its KKTs!

KKT-based reformulation (cont'd)

- Assume linear regression, i.e., $\mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i) = \boldsymbol{\theta}^\top \boldsymbol{\varphi}_i$
- KKT-based reformulation of the bilevel problem

$$
\begin{aligned}
\underset{\boldsymbol{\theta}, \mathbf{x}_i, \boldsymbol{\lambda}_i, \widehat{\mathbf{w}}_i}{\text{minimize}} \quad & \frac{1}{2k} \sum_{i=1}^{k} \| c(\mathbf{x}_i) - c(\mathbf{x}^\star(\mathbf{w}_i)) \|_2^2 \\
\text{subject to} \quad & \widehat{\mathbf{w}}_i = \boldsymbol{\theta}^\top \boldsymbol{\varphi}_i, \qquad \text{upper level} \\
\hdashrule{\linewidth}{0.4pt}{2pt} & \qquad \text{lower level} \\
& g(\mathbf{x}_i, \widehat{\mathbf{w}}_i) \leqslant \mathbf{0}, \quad \boldsymbol{\lambda}_i \geqslant \mathbf{0}, \\
& \nabla_{\mathbf{x}} c(\mathbf{x}_i) + \boldsymbol{\lambda}^\top \nabla_{\mathbf{x}_i} g(\mathbf{x}, \widehat{\mathbf{w}}_i) = \mathbf{0}, \\
& \boldsymbol{\lambda}_i^\top g(\mathbf{x}_i, \widehat{\mathbf{w}}_i) = 0, \quad \forall i = 1, \ldots, k
\end{aligned}
$$

KKT-based reformulation (cont'd)

- Assume linear regression, i.e., $\mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i) = \boldsymbol{\theta}^\top \boldsymbol{\varphi}_i$
- KKT-based reformulation of the bilevel problem

$$
\begin{aligned}
\underset{\boldsymbol{\theta}, \mathbf{x}_i, \boldsymbol{\lambda}_i, \widehat{\mathbf{w}}_i}{\text{minimize}} \quad & \frac{1}{2k} \sum_{i=1}^{k} \| c(\mathbf{x}_i) - c(\mathbf{x}^\star(\mathbf{w}_i)) \|_2^2 \\
\text{subject to} \quad & \widehat{\mathbf{w}}_i = \boldsymbol{\theta}^\top \boldsymbol{\varphi}_i, \qquad \text{upper level} \\
\hline
& g(\mathbf{x}_i, \widehat{\mathbf{w}}_i) \leqslant \mathbf{0}, \quad \boldsymbol{\lambda}_i \geqslant \mathbf{0}, \qquad \text{lower level} \\
& \nabla_{\mathbf{x}} c(\mathbf{x}_i) + \boldsymbol{\lambda}^\top \nabla_{\mathbf{x}_i} g(\mathbf{x}, \widehat{\mathbf{w}}_i) = \mathbf{0}, \\
& \boldsymbol{\lambda}_i^\top g(\mathbf{x}_i, \widehat{\mathbf{w}}_i) = 0, \quad \forall i = 1, \ldots, k
\end{aligned}
$$

- **Q1.** Where does $c(\mathbf{x}^\star(\mathbf{w}_i))$ come from?

KKT-based reformulation (cont'd)

- Assume linear regression, i.e., $\mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i) = \boldsymbol{\theta}^\top \boldsymbol{\varphi}_i$
- KKT-based reformulation of the bilevel problem

$$
\begin{aligned}
\underset{\boldsymbol{\theta}, \mathbf{x}_i, \boldsymbol{\lambda}_i, \widehat{\mathbf{w}}_i}{\text{minimize}} \quad & \frac{1}{2k} \sum_{i=1}^{k} \| c(\mathbf{x}_i) - c(\mathbf{x}^\star(\mathbf{w}_i)) \|_2^2 \\
\text{subject to} \quad & \widehat{\mathbf{w}}_i = \boldsymbol{\theta}^\top \boldsymbol{\varphi}_i, \qquad \text{upper level} \\
& \text{- -} \\
& g(\mathbf{x}_i, \widehat{\mathbf{w}}_i) \leqslant \mathbf{0}, \quad \boldsymbol{\lambda}_i \geqslant \mathbf{0}, \qquad \text{lower level} \\
& \nabla_{\mathbf{x}} c(\mathbf{x}_i) + \boldsymbol{\lambda}^\top \nabla_{\mathbf{x}_i} g(\mathbf{x}, \widehat{\mathbf{w}}_i) = \mathbf{0}, \\
& \boldsymbol{\lambda}_i^\top g(\mathbf{x}_i, \widehat{\mathbf{w}}_i) = 0, \quad \forall i = 1, \ldots, k
\end{aligned}
$$

- **Q1.** Where does $c(\mathbf{x}^\star(\mathbf{w}_i))$ come from?
- Non-linear optimization program with hard complementarity condition.
- **Q2.** How can we make it more tractable? (demonstrate in-class)
- **Q3.** How to introduce regularization?

Gradient descent-based solution

- Recall the optimization objective

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \mathcal{R} \triangleq \frac{1}{2k} \sum_{i=1}^{k} \| c(\mathbf{x}^\star(\mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)) - c_i^\star \|_2^2$$

Gradient descent-based solution

- Recall the optimization objective

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \mathcal{R} \triangleq \frac{1}{2k} \sum_{i=1}^{k} \| c(\mathbf{x}^\star(\mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)) - c_i^\star \|_2^2$$

- Derivatives of the regret w.r.t. model parameter $\boldsymbol{\theta}$ (linear regression)

$$\begin{aligned}
\frac{\partial \mathcal{R}}{\partial \boldsymbol{\theta}} =& \frac{\partial \mathcal{R}}{\partial c} \cdot \frac{\partial c}{\partial \mathbf{x}^\star} \cdot \frac{\partial \mathbf{x}^\star}{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)} \cdot \frac{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)}{\partial \boldsymbol{\theta}} \\
=& \frac{1}{k} \sum_{i=1}^{k} \left(c(\mathbf{x}^\star(\mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i))) - c_i^\star \right) \cdot \frac{\partial c}{\partial \mathbf{x}^\star} \cdot \frac{\partial \mathbf{x}^\star}{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)} \cdot \frac{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)}{\partial \boldsymbol{\theta}} \\
=& \frac{1}{k} \sum_{i=1}^{k} \left(c(\mathbf{x}^\star(\mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i))) - c_i^\star \right) \cdot \nabla_{\mathbf{x}} c(x) \cdot \frac{\partial \mathbf{x}^\star}{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)} \cdot \frac{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)}{\partial \boldsymbol{\theta}} \\
=& \frac{1}{k} \sum_{i=1}^{k} \left(c(\mathbf{x}^\star(\mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i))) - c_i^\star \right) \cdot \nabla_{\mathbf{x}} c(x_i) \cdot \frac{\partial \mathbf{x}^\star}{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)} \cdot \boldsymbol{\varphi}_i
\end{aligned}$$

Gradient descent-based solution

- Recall the optimization objective

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \mathcal{R} \triangleq \frac{1}{2k} \sum_{i=1}^{k} \| c(\mathbf{x}^\star(\mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)) - c_i^\star \|_2^2$$

- Derivatives of the regret w.r.t. model parameter $\boldsymbol{\theta}$ (linear regression)

$$\begin{aligned}
\frac{\partial \mathcal{R}}{\partial \boldsymbol{\theta}} &= \frac{\partial \mathcal{R}}{\partial c} \cdot \frac{\partial c}{\partial \mathbf{x}^\star} \cdot \frac{\partial \mathbf{x}^\star}{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)} \cdot \frac{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)}{\partial \boldsymbol{\theta}} \\
&= \frac{1}{k} \sum_{i=1}^{k} \left(c(\mathbf{x}^\star(\mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i))) - c_i^\star \right) \cdot \frac{\partial c}{\partial \mathbf{x}^\star} \cdot \frac{\partial \mathbf{x}^\star}{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)} \cdot \frac{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)}{\partial \boldsymbol{\theta}} \\
&= \frac{1}{k} \sum_{i=1}^{k} \left(c(\mathbf{x}^\star(\mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i))) - c_i^\star \right) \cdot \nabla_{\mathbf{x}} c(x) \cdot \frac{\partial \mathbf{x}^\star}{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)} \cdot \frac{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)}{\partial \boldsymbol{\theta}} \\
&= \frac{1}{k} \sum_{i=1}^{k} \left(c(\mathbf{x}^\star(\mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i))) - c_i^\star \right) \cdot \nabla_{\mathbf{x}} c(x_i) \cdot \frac{\partial \mathbf{x}^\star}{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)} \cdot \boldsymbol{\varphi}_i
\end{aligned}$$

- Regret is then minimized over iterations $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \frac{\partial \mathcal{R}}{\partial \boldsymbol{\theta}}$
- How to compute the last unknown gradient $\frac{\partial \mathbf{x}^\star}{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)}$?

Computing $\frac{\partial \mathbf{x}^\star}{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)}$ for convex quadratic programs

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & \frac{1}{2}\mathbf{x}^\top \mathbf{C}\mathbf{x} + \mathbf{c}^\top \mathbf{x} \\ \text{subject to} \quad & \mathbf{A}\mathbf{x} = \mathbf{b}, \quad \mathbf{G}\mathbf{x} \leqslant \mathbf{h} \end{aligned}$$

The Lagrangian function $\mathcal{L}(\mathbf{x}, \boldsymbol{\nu}, \boldsymbol{\lambda}) = \frac{1}{2}\mathbf{x}^\top \mathbf{C}\mathbf{x} + \mathbf{c}^\top \mathbf{x} + \boldsymbol{\nu}^\top(\mathbf{A}\mathbf{x} - \mathbf{b}) + \boldsymbol{\lambda}^\top(\mathbf{G}\mathbf{x} - \mathbf{h})$

The stationarity, primal feasibility, and complementarity slackness:

$$\begin{aligned} \mathbf{C}\mathbf{x}^\star + \mathbf{c} + \mathbf{A}^\top \boldsymbol{\nu}^\star + \mathbf{G}^\top \boldsymbol{\lambda}^\star &= \mathbf{0} \\ \mathbf{A}\mathbf{x}^\star - \mathbf{b} &= \mathbf{0} \\ \text{diag}[\boldsymbol{\lambda}^\star](\mathbf{G}\mathbf{x}^\star - \mathbf{h}) &= \mathbf{0} \end{aligned}$$

Implicit differentiation (using a product rule, i.e., $\partial \mathbf{C}\mathbf{x}^\star = \partial \mathbf{C} \cdot \mathbf{x}^\star + \mathbf{C} \cdot \partial \mathbf{x}^\star$)[1]

$$\begin{bmatrix} \mathbf{C} & \mathbf{G}^\top & \mathbf{A}^\top \\ \text{diag}[\boldsymbol{\lambda}^\star]\mathbf{G} & \text{diag}[\mathbf{G}\mathbf{x}^\star - \mathbf{h}] & \mathbf{0} \\ \mathbf{A} & \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \partial \mathbf{x} \\ \partial \boldsymbol{\lambda} \\ \partial \boldsymbol{\nu} \end{bmatrix} = \begin{bmatrix} -\partial \mathbf{C}\mathbf{x}^\star - \partial \mathbf{c} - \partial \mathbf{G}^\top \boldsymbol{\lambda}^\star - \partial \mathbf{A}^\top \boldsymbol{\nu}^\star \\ -\text{diag}[\boldsymbol{\lambda}^\star]\partial \mathbf{G}\mathbf{x}^\star + \text{diag}[\boldsymbol{\lambda}^\star]\partial \mathbf{h} \\ -\partial \mathbf{A}\mathbf{x}^\star + \partial \mathbf{b} \end{bmatrix}$$

[1] Amos B., & Kolter J. Z. Optnet: Differentiable optimization as a layer in neural networks. 2017

Computing $\frac{\partial \mathbf{x}^\star}{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)}$ for convex quadratic programs (cont'd)

$$\begin{bmatrix} \partial \mathbf{x} \\ \partial \boldsymbol{\lambda} \\ \partial \boldsymbol{\nu} \end{bmatrix} = \begin{bmatrix} \mathbf{C} & \mathbf{G}^\top & \mathbf{A}^\top \\ \text{diag}[\boldsymbol{\lambda}^\star]\mathbf{G} & \text{diag}[\mathbf{G}\mathbf{x}^\star - \mathbf{h}] & \mathbf{0} \\ \mathbf{A} & \mathbf{0} & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} -\partial \mathbf{C}\mathbf{x}^\star - \partial \mathbf{c} - \partial \mathbf{G}^\top \boldsymbol{\lambda}^\star - \partial \mathbf{A}^\top \boldsymbol{\nu}^\star \\ -\text{diag}[\boldsymbol{\lambda}^\star]\partial \mathbf{G}\mathbf{x}^\star + \text{diag}[\boldsymbol{\lambda}^\star]\partial \mathbf{h} \\ -\partial \mathbf{A}\mathbf{x}^\star + \partial \mathbf{b} \end{bmatrix}$$

Suppose that vector $\mathbf{b}$ is given by the prediction of our model, i.e., $\mathbf{b} = \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)$

Focus only on $\partial \mathbf{b}$, and set other differentials to zero

$$\begin{bmatrix} \partial \mathbf{x} \\ \partial \boldsymbol{\lambda} \\ \partial \boldsymbol{\nu} \end{bmatrix} = \begin{bmatrix} \mathbf{C} & \mathbf{G}^\top & \mathbf{A}^\top \\ \text{diag}[\boldsymbol{\lambda}^\star]\mathbf{G} & \text{diag}[\mathbf{G}\mathbf{x}^\star - \mathbf{h}] & \mathbf{0} \\ \mathbf{A} & \mathbf{0} & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \partial \mathbf{b} \end{bmatrix} = \begin{bmatrix} (\mathbf{A}^\top)^{-1} \partial \mathbf{b} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix}$$

Hence, the $\frac{\partial \mathbf{x}}{\partial \mathbf{b}} = \frac{\partial \mathbf{x}}{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)} = (\mathbf{A}^\top)^{-1}$

By analogy, compute the derivative of optimizer $\mathbf{x}$ to any other QP problem data

Gradient descent for decision-focused learning

$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \mathcal{R} \triangleq \frac{1}{2k} \sum_{i=1}^{k} \| c(\mathbf{x}^\star(\mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)) - c_i^\star \|_2^2$$

The gradient descent-like algorithms use the update

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \frac{\partial \mathcal{R}}{\partial \boldsymbol{\theta}}$$

for some step size $\alpha > 0$

- Requires solving an optimization problem at every iteration (**Q.** Why?)
- Yet, more versatile solution (no structural assumption on model $\mathbb{W}_{\boldsymbol{\theta}}$)
- Also, scales up to much larger problems (in terms of dimensions n, m and k)

Illustrative example: Setting

- Dataset $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_i, y_i), \ldots, (\mathbf{x}_n, y_n)\}$ of n records
- **Prediction-focused learning:** find the linear model $\widehat{\mathbf{y}} = \mathbf{w}^\top \widehat{\mathbf{x}}$ by solving

$$\underset{\mathbf{w}}{\text{minimize}} \quad \frac{1}{2n} \sum_{i=1}^{n} (\mathbf{w}^\top \mathbf{x}_i - y_i)^2$$

- The downstream $\widehat{y}-$parametrized decision problem:

$$\begin{aligned} \underset{z}{\text{minimize}} \quad & \frac{1}{2}(z - \widehat{y})^2 + \lambda z^2 \\ \text{subject to} \quad & z \geqslant 0 \end{aligned}$$

- **Decision-focused learning:** seeks the linear model by solving

$$\begin{aligned} \underset{\mathbf{w}, \widehat{\mathbf{y}}}{\text{minimize}} \quad & \frac{1}{2n} \sum_{i=1}^{n} (z^\star(\widehat{y}_i) - z^\star(y_i))^2 \\ \text{subject to} \quad & \widehat{y}_i = \mathbf{w}^\top \mathbf{x}_i \\ & z^\star(\widehat{y}_i) = \underset{z}{\text{argmin}} \quad \frac{1}{2}(z - \widehat{y}_i)^2 + \lambda z^2 \\ & \qquad\qquad \text{subject to} \quad z \geqslant 0, \qquad\qquad \forall i = 1, \ldots, n \end{aligned}$$

Illustrative example: Solution via bilevel optimization

- The KKTs of the lower-level problem are

$$
\begin{aligned}
z - \widehat{y} + \lambda - \mu &= 0 && \text{(Lagrange optimality)} \\
z &\geqslant 0 && \text{(primal feasibility)} \\
\mu &\geqslant 0 && \text{(dual feasibility)} \\
z\mu &= 0 && \text{(complementarity slackness)}
\end{aligned}
$$

Illustrative example: Solution via bilevel optimization

- The KKTs of the lower-level problem are

$$
\begin{aligned}
z - \widehat{y} + \lambda - \mu = 0 \quad & \text{(Lagrange optimality)} \\
z \geqslant 0 \quad & \text{(primal feasibility)} \\
\mu \geqslant 0 \quad & \text{(dual feasibility)} \\
z\mu = 0 \quad & \text{(complementarity slackness)}
\end{aligned}
$$

- Big-M reformulation of complementarity slackness:

$$
\begin{cases}
\mu \leqslant Mu \\
z \leqslant M(1-u)
\end{cases}
$$

where $u \in \{0, 1\}$ is an auxiliary binary variable, and M is a large enough constant

Illustrative example: Solution via bilevel optimization

- The KKTs of the lower-level problem are

$$
\begin{aligned}
z - \widehat{y} + \lambda - \mu = 0 \quad & \text{(Lagrange optimality)} \\
z \geqslant 0 \quad & \text{(primal feasibility)} \\
\mu \geqslant 0 \quad & \text{(dual feasibility)} \\
z\mu = 0 \quad & \text{(complementarity slackness)}
\end{aligned}
$$

- Big-M reformulation of complementarity slackness:

$$
\begin{cases}
\mu \leqslant Mu \\
z \leqslant M(1-u)
\end{cases}
$$

  where $u \in \{0,1\}$ is an auxiliary binary variable, and M is a large enough constant

- The bilevel reformulation of the decision-focused learning problem:

$$
\begin{aligned}
\underset{\mathbf{w},\widehat{\mathbf{y}},\mathbf{z},\boldsymbol{\mu},\mathbf{u}}{\text{minimize}} \quad & \frac{1}{2n}\sum_{i=1}^{n}(z^\star(\widehat{y}_i) - z^\star(y_i))^2 \\
\text{subject to} \quad & \widehat{y}_i = \mathbf{w}^\top \widehat{\mathbf{x}}_i, \quad \forall i = 1,\ldots,n \\
& \mathbf{z} - \widehat{\mathbf{y}} + 2\lambda\mathbf{z} - \boldsymbol{\mu} = \mathbf{0} \\
& \mathbf{z} \geqslant \mathbf{0}, \quad \boldsymbol{\mu} \geqslant \mathbf{0}, \quad \mathbf{u} \in \{0,1\}^n \\
& \boldsymbol{\mu} \leqslant M\mathbf{u}, \quad \mathbf{z} \leqslant M(\mathbf{1}-\mathbf{u})
\end{aligned}
$$

Illustrative example: Solution via stochastic gradient descent

$$
\begin{aligned}
\underset{\mathbf{w},\widehat{\mathbf{y}}}{\text{minimize}} \quad & \frac{1}{2n}\sum_{i=1}^{n}(z^\star(\widehat{y}_i) - z^\star(y_i))^2 \\
\text{subject to} \quad & \widehat{y}_i = \mathbf{w}^\top \mathbf{x}_i \\
& z^\star(\widehat{y}_i) = \underset{z}{\text{argmin}} \quad \frac{1}{2}(z - \widehat{y}_i)^2 + \lambda z^2 \\
& \qquad\qquad \text{subject to} \quad z \geqslant 0, \qquad\qquad \forall i = 1, \ldots, n
\end{aligned}
$$

- Closed-form solution for the lower-level problem

$$z^\star(\widehat{y}_i) = \max\left\{0, \frac{\widehat{y}_i}{1+2\lambda}\right\}$$

- The gradient $\partial z^\star(\widehat{y}_i)/\partial \widehat{y}_i$ is obtained using LogSumExp approximation:

$$z^\star(\widehat{y}_i) = \max\left\{0, \frac{\widehat{y}_i}{1+2\lambda}\right\} \approx \log\left(1 + \exp\left(\frac{\widehat{y}_i}{1+2\lambda}\right)\right) \qquad \text{(EECS 559)}$$

$$z^\star(\widehat{y}_i)/\partial \widehat{y}_i \approx \frac{\exp\left(\frac{\widehat{y}_i}{1+2\lambda}\right)}{1+\exp\left(\frac{\widehat{y}_i}{1+2\lambda}\right)}\frac{1}{1+2\lambda}$$

Illustrative example: Solution via stochastic gradient descent

$$\begin{aligned}
\underset{\mathbf{w},\widehat{\mathbf{y}}}{\text{minimize}} \quad & \frac{1}{2n}\sum_{i=1}^{n}(z^\star(\widehat{y}_i) - z^\star(y_i))^2 \\
\text{subject to} \quad & \widehat{y}_i = \mathbf{w}^\top \mathbf{x}_i \\
& z^\star(\widehat{y}_i) = \underset{z}{\text{argmin}} \quad \frac{1}{2}(z - \widehat{y}_i)^2 + \lambda z^2 \\
& \qquad\qquad \text{subject to} \quad z \geqslant 0, \qquad\qquad \forall i = 1, \ldots, n
\end{aligned}$$

- For SGD, we draw one sample i from a dataset and minimize

$$\mathcal{C}_i = (z^\star(\widehat{y}_i) - z^\star(y_i))^2$$

- By the chain rule, the gradient is

$$\begin{aligned}
\frac{\partial \mathcal{C}_i}{\partial \mathbf{w}} &= (z^\star(\widehat{y}_i) - z^\star(y_i)) \cdot \frac{\partial z^\star}{\partial y_i} \cdot \frac{\partial y_i}{\partial \mathbf{w}} \\
&= \left(\underbrace{\max\left\{0, \frac{\widehat{y}_i}{1+2\lambda}\right\}}_{\text{closed form for } z^\star} - \underbrace{z^\star(y_i)}_{\text{precomputed offline}} \right) \cdot \underbrace{\frac{\exp\left(\frac{\widehat{y}_i}{1+2\lambda}\right)}{1+\exp\left(\frac{\widehat{y}_i}{1+2\lambda}\right)} \frac{1}{1+2\lambda}}_{\partial z^\star(\widehat{y}_i)/\partial \widehat{y}_i} \cdot \mathbf{x}_i
\end{aligned}$$

- The SGD perform the following update:

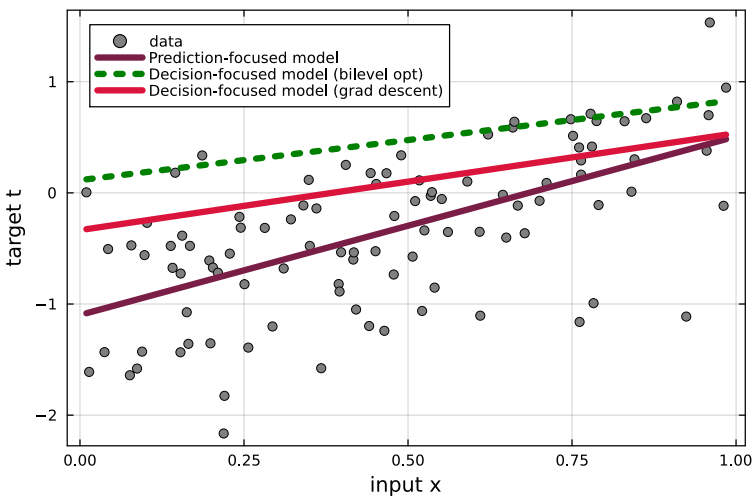
$$\mathbf{w} \leftarrow \mathbf{w} - \alpha \frac{\partial \mathcal{C}_i}{\partial \mathbf{w}}$$

with some step size $\alpha > 0$.

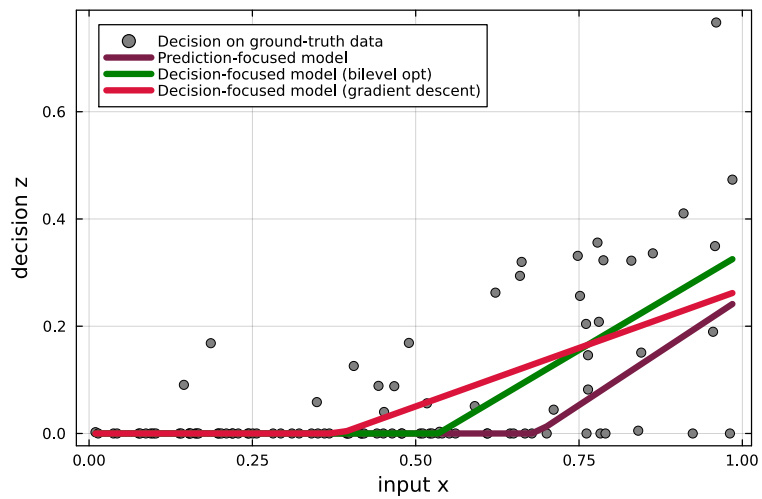
Illustrative example: Results

- Experiments on synthetic data (100 samples)
- **Prediction space:** displays how the model explains the underlying data
- **Decision space:** displays how the model prescribes decisions

prediction space



decision space



- Prediction-focused model is better in predictions, decision-focused – in decisions
- Bilevel optimization provides the **global** optimal solution
- Gradient descent-based solution does not have global guarantees

Tutorial time

- Look around you and form teams of 2 people (1 min)
- Pick one person to code; the other one guides
- Work in pairs for the whole tutorial session

Goal: replicate the results of the illustrative example

Multiparametric programming to ease computational burden

Challenge: Large-scale downstream optimization

- Wind power forecast $\widehat{\mathbf{w}}$−parametrized DC-OPF problem

$$
\begin{aligned}
\underset{\mathbf{p}}{\text{minimize}} \quad & \frac{1}{2}\mathbf{p}^\top \mathbf{C}\mathbf{p} + \mathbf{c}^\top \mathbf{p} && \textit{generator dispatch cost} \\
\text{subject to} \quad & \underline{\mathbf{p}} \leqslant \mathbf{p} \leqslant \overline{\mathbf{p}} && \textit{generation limits} \\
& \mathbf{1}^\top(\mathbf{p} + \widehat{\mathbf{w}} - \mathbf{d}) = 0 && \textit{power balance condition} \\
& |\mathbf{F}(\mathbf{p} + \widehat{\mathbf{w}} - \mathbf{d})| \leqslant \overline{\mathbf{f}} && \textit{power flow limits}
\end{aligned}
$$

- For practical-sized systems, there are thousands of variables and constraints
- Solving DC-OPF at every SGD update significantly slows convergence

Multiparametric programming to ease computational burden

Challenge: Large-scale downstream optimization

- Wind power forecast $\widehat{\mathbf{w}}$−parametrized DC-OPF problem

$$
\begin{aligned}
\underset{\mathbf{p}}{\text{minimize}} \quad & \frac{1}{2}\mathbf{p}^\top \mathbf{C}\mathbf{p} + \mathbf{c}^\top \mathbf{p} && \textit{generator dispatch cost} \\
\text{subject to} \quad & \underline{\mathbf{p}} \leqslant \mathbf{p} \leqslant \overline{\mathbf{p}} && \textit{generation limits} \\
& \mathbf{1}^\top(\mathbf{p} + \widehat{\mathbf{w}} - \mathbf{d}) = 0 && \textit{power balance condition} \\
& |\mathbf{F}(\mathbf{p} + \widehat{\mathbf{w}} - \mathbf{d})| \leqslant \overline{\mathbf{f}} && \textit{power flow limits}
\end{aligned}
$$

- For practical-sized systems, there are thousands of variables and constraints
- Solving DC-OPF at every SGD update significantly slows convergence

Solution: Multiparametric programming (MPP)

- Studies the solutions of an optimization problem as a function of its parameters
- Let $\widehat{\mathbf{w}}$ belong to the union $\mathcal{W} \triangleq \bigcup_{i=1}^{k} [0,1]_i$ of normalized output ranges of k wind farms. A sampled wind power output $\widehat{\mathbf{w}}$ is seen as a random draw from $\mathcal{W}$
- For convex quadratic programs, set $\mathcal{W}$ can be portioned into polyhedral regions, each having a closed-form solution for primal and dual variables $\Rightarrow$ speed up!

Compact DC-OPF formulation for MPP

$$\begin{aligned} \underset{\mathbf{p}}{\text{minimize}} \quad & \frac{1}{2}\mathbf{p}^\top \mathbf{C}\mathbf{p} + \mathbf{c}^\top \mathbf{p} \\ \text{subject to} \quad & \mathbf{A}\mathbf{p} \leqslant \mathbf{B}\widehat{\mathbf{w}} + \mathbf{b} \quad : \boldsymbol{\lambda} \end{aligned}$$

where problem data is

$$\mathbf{A} = \begin{bmatrix} \mathbf{I} \\ -\mathbf{I} \\ -\mathbf{1}^\top \\ \mathbf{F} \\ -\mathbf{F} \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1}^\top \\ \mathbf{F} \\ -\mathbf{F} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \overline{\mathbf{p}} \\ -\underline{\mathbf{p}} \\ -\mathbf{1}^\top \underline{\mathbf{d}} \\ \overline{\mathbf{f}} \\ \overline{\mathbf{f}} \end{bmatrix}$$

Q. Why can we replace the power balance constraint with inequality?

Compact DC-OPF formulation for MPP

$$\begin{aligned} \underset{\mathbf{p}}{\text{minimize}} \quad & \frac{1}{2}\mathbf{p}^\top \mathbf{C}\mathbf{p} + \mathbf{c}^\top \mathbf{p} \\ \text{subject to} \quad & \mathbf{A}\mathbf{p} \leqslant \mathbf{B}\widehat{\mathbf{w}} + \mathbf{b} \quad : \boldsymbol{\lambda} \end{aligned}$$

where problem data is

$$\mathbf{A} = \begin{bmatrix} \mathbf{I} \\ -\mathbf{I} \\ -\mathbf{1}^\top \\ \mathbf{F} \\ -\mathbf{F} \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1}^\top \\ \mathbf{F} \\ -\mathbf{F} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} \overline{\mathbf{p}} \\ -\underline{\mathbf{p}} \\ -\mathbf{1}^\top \underline{\mathbf{d}} \\ \overline{\mathbf{f}} \\ \overline{\mathbf{f}} \end{bmatrix}$$

Q. Why can we replace the power balance constraint with inequality?

- A multiparametric programming problem (MPP) dependent on $\widehat{\mathbf{w}}$
- For some realization $\widehat{\mathbf{w}}$, MPP expresses the primal $\mathbf{p}$ and dual $\boldsymbol{\lambda}$ solutions as affine functions of some $\widehat{\mathbf{w}}$ in the *neighborhood* of $\widehat{\mathbf{w}}$

Closed-form DC-OPF solutions from MPP

- Lagrangian function of the compact DC-OPF:

$$\mathcal{L}(\mathbf{p}, \boldsymbol{\lambda}) = \frac{1}{2}\mathbf{p}^\top \mathbf{C}\mathbf{p} + \mathbf{c}^\top \mathbf{p} + \boldsymbol{\lambda}^\top (\mathbf{A}\mathbf{p} - \mathbf{B}\widehat{\mathbf{w}} - \mathbf{b})$$

- Let $\tilde{\mathbf{A}}, \tilde{\mathbf{B}}, \tilde{\mathbf{b}}$ be sub-matrices/vectors containing only those entries of $\mathbf{A}, \mathbf{B}, \mathbf{b}$ that correspond to the *active* (binding) constraints ($\lambda > 0$)

- $\overline{\mathbf{A}}, \overline{\mathbf{B}}, \overline{\mathbf{b}}$ collect the remaining entries corresponding to *inactive* constraints ($\lambda = 0$).

Closed-form DC-OPF solutions from MPP

- Lagrangian function of the compact DC-OPF:

$$\mathcal{L}(\mathbf{p}, \boldsymbol{\lambda}) = \frac{1}{2}\mathbf{p}^\top \mathbf{C}\mathbf{p} + \mathbf{c}^\top \mathbf{p} + \boldsymbol{\lambda}^\top (\mathbf{A}\mathbf{p} - \mathbf{B}\widehat{\mathbf{w}} - \mathbf{b})$$

- Let $\tilde{\mathbf{A}}, \tilde{\mathbf{B}}, \tilde{\mathbf{b}}$ be sub-matrices/vectors containing only those entries of $\mathbf{A}, \mathbf{B}, \mathbf{b}$ that correspond to the *active* (binding) constraints ($\lambda > 0$)

- $\overline{\mathbf{A}}, \overline{\mathbf{B}}, \overline{\mathbf{b}}$ collect the remaining entries corresponding to *inactive* constraints ($\lambda = 0$).

- From the Karush–Kuhn–Tucker (KKT) conditions we have a system of equations:

$$\mathbf{C}\mathbf{p} + \mathbf{c} + \tilde{\mathbf{A}}^\top \tilde{\boldsymbol{\lambda}} = \mathbf{0}$$
$$\tilde{\mathbf{A}}\mathbf{p} - \tilde{\mathbf{B}}\widehat{\mathbf{w}} - \tilde{\mathbf{b}} = \mathbf{0}$$

Q. From which KKT conditions do these equation come from?

- Manipulating these equations, we get the closed-form solutions

Closed-form DC-OPF solutions from MPP (cont'd)

$$\mathbf{C}\mathbf{p} + \mathbf{c} + \tilde{\mathbf{A}}^\top \tilde{\boldsymbol{\lambda}} = \mathbf{0} \tag{1a}$$

$$\tilde{\mathbf{A}}\mathbf{p} - \tilde{\mathbf{B}}\widehat{\mathbf{w}} - \tilde{\mathbf{b}} = \mathbf{0} \tag{1b}$$

From (1a) we express $\mathbf{p}$ (assuming $\mathbf{C}$ is invertible)

$$\mathbf{p} = -\mathbf{C}^{-1}\left(\mathbf{c} + \tilde{\mathbf{A}}^\top \tilde{\boldsymbol{\lambda}}\right) \tag{2}$$

and substitute it into (1b):

$$-\tilde{\mathbf{A}}\mathbf{C}^{-1}\mathbf{c} - \tilde{\mathbf{A}}\mathbf{C}^{-1}\tilde{\mathbf{A}}^\top \tilde{\boldsymbol{\lambda}} - \tilde{\mathbf{B}}\widehat{\mathbf{w}} - \tilde{\mathbf{b}} = \mathbf{0}$$

$$\Leftrightarrow \quad \tilde{\boldsymbol{\lambda}} = \underbrace{-\left(\tilde{\mathbf{A}}\mathbf{C}^{-1}\tilde{\mathbf{A}}^\top\right)^{-1}\tilde{\mathbf{B}}}_{\mathbf{D}}\,\widehat{\mathbf{w}}\,\underbrace{-\left(\tilde{\mathbf{A}}\mathbf{C}^{-1}\tilde{\mathbf{A}}^\top\right)^{-1}\left(\tilde{\mathbf{b}} + \tilde{\mathbf{A}}\mathbf{C}^{-1}\mathbf{c}\right)}_{\mathbf{d}}$$

$$\Leftrightarrow \quad \tilde{\boldsymbol{\lambda}} = \mathbf{D}\widehat{\mathbf{w}} + \mathbf{d} \quad \text{(closed-form solution for duals)} \tag{3}$$

Substituting (3) into (2), we have the solution for $\mathbf{p}$:

$$\mathbf{p} = -\mathbf{C}^{-1}\left(\mathbf{c} + \tilde{\mathbf{A}}^\top\left(\mathbf{D}\widehat{\mathbf{w}} + \mathbf{d}\right)\right) = \underbrace{-\mathbf{C}^{-1}\tilde{\mathbf{A}}^\top\mathbf{D}}_{\boldsymbol{\Pi}}\,\widehat{\mathbf{w}}\,\underbrace{-\mathbf{C}^{-1}\mathbf{c} - \mathbf{C}^{-1}\tilde{\mathbf{A}}^\top\mathbf{d}}_{\boldsymbol{\pi}}$$

$$\Leftrightarrow \quad \mathbf{p} = \boldsymbol{\Pi}\widehat{\mathbf{w}} + \boldsymbol{\pi} \quad \text{(closed-form solution for primals)} \tag{4}$$

Closed-form DC-OPF solutions from MPP (cont'd)

- The primal-dual solution depends linearly on $\widehat{\mathbf{w}}$:

$$\begin{bmatrix} \mathbf{p} \\ \tilde{\boldsymbol{\lambda}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Pi} \\ \mathbf{D} \end{bmatrix} \widehat{\mathbf{w}} + \begin{bmatrix} \boldsymbol{\pi} \\ \mathbf{d} \end{bmatrix} \tag{5}$$

Q. How do we obtain matrices and vectors $\boldsymbol{\Pi}, \mathbf{D}, \boldsymbol{\pi}$ and $\mathbf{d}$?

Closed-form DC-OPF solutions from MPP (cont'd)

- The primal-dual solution depends linearly on $\widehat{\mathbf{w}}$:

$$\begin{bmatrix} \mathbf{p} \\ \tilde{\boldsymbol{\lambda}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Pi} \\ \mathbf{D} \end{bmatrix} \widehat{\mathbf{w}} + \begin{bmatrix} \boldsymbol{\pi} \\ \mathbf{d} \end{bmatrix} \tag{5}$$

Q. How do we obtain matrices and vectors $\boldsymbol{\Pi}, \mathbf{D}, \boldsymbol{\pi}$ and $\mathbf{d}$?

- For a given $\widehat{\mathbf{w}}$, we obtain them by solving an optimization problem (**bad** news)
- Eq. (5) valid not only for $\widehat{\mathbf{w}}$ but for all $\widehat{\mathbf{w}}$ in the neighborhood of $\widehat{\mathbf{w}}$ (**good** news)

Closed-form DC-OPF solutions from MPP (cont'd)

- The primal-dual solution depends linearly on $\widehat{\mathbf{w}}$:

$$\begin{bmatrix} \mathbf{p} \\ \tilde{\boldsymbol{\lambda}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Pi} \\ \mathbf{D} \end{bmatrix} \widehat{\mathbf{w}} + \begin{bmatrix} \boldsymbol{\pi} \\ \mathbf{d} \end{bmatrix} \tag{5}$$

Q. How do we obtain matrices and vectors $\boldsymbol{\Pi}, \mathbf{D}, \boldsymbol{\pi}$ and $\mathbf{d}$?

- For a given $\widehat{\mathbf{w}}$, we obtain them by solving an optimization problem (**bad** news)
- Eq. (5) valid not only for $\widehat{\mathbf{w}}$ but for all $\widehat{\mathbf{w}}$ in the neighborhood of $\widehat{\mathbf{w}}$ (**good** news)
- The neighborhood originates from two KKT conditions:

$$(\overline{\mathbf{A}}\boldsymbol{\Pi} - \overline{\mathbf{B}})\widehat{\mathbf{w}} < \overline{\mathbf{b}} - \overline{\mathbf{A}}\boldsymbol{\pi} \qquad \text{(primal feasibility of inactive constraints)} \tag{6a}$$

$$\mathbf{D}\widehat{\mathbf{w}} + \mathbf{d} \geqslant \mathbf{0} \qquad \text{(dual feasibility)} \tag{6b}$$

- We term the neighborhood of $\widehat{\mathbf{w}}$ a *critical region (CR)*, formally defined as:

$$\mathcal{R} = \left\{ \widehat{\mathbf{w}} \in \mathcal{W} \mid (\overline{\mathbf{A}}\boldsymbol{\Pi} - \overline{\mathbf{B}})\widehat{\mathbf{w}} < \overline{\mathbf{b}} - \overline{\mathbf{A}}\boldsymbol{\pi},\ \mathbf{D}\widehat{\mathbf{w}} + \mathbf{d} \geqslant \mathbf{0} \right\} \tag{7}$$

- In fact, there are many CRs. The task is to identify the CR corresponding to a particular realization of wind power, and then compute solution from (5).

Identifying critical regions

- 5-Bus PJM testbed with 3 variable net loads
- Initialize the set of critical regions $\mathcal{CR} = \{\varnothing\}$
- To identify critical regions, perform the following iterations

for $i = 1, \ldots, 1000$ **do**
  Sample $\widehat{\mathbf{w}}$ from $\mathcal{W}$
  if $\widehat{\mathbf{w}}$ does not belong to any $\mathcal{R}$ in $\mathcal{CR}$ **then**
    Solve DC-OPF on $\widehat{\mathbf{w}}$, compute $\mathcal{R}$ and add to $\mathcal{CR}$
  end if
end for

Q1. What is the complexity of this algorithm?

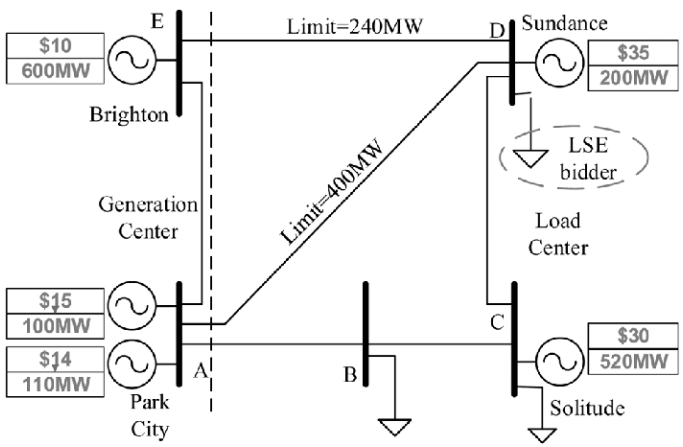
Identifying critical regions

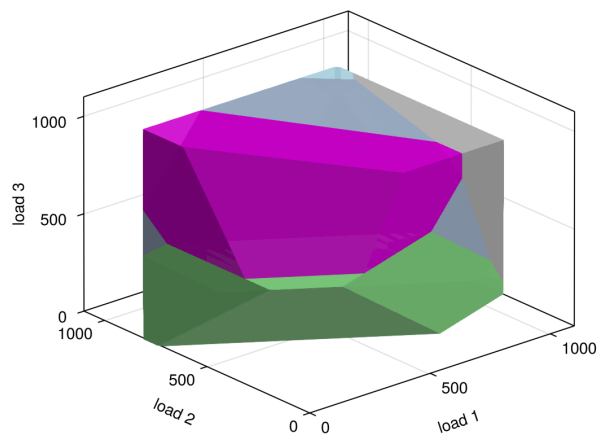
- 5-Bus PJM testbed with 3 variable net loads
- Initialize the set of critical regions $\mathcal{CR} = \{\varnothing\}$
- To identify critical regions, perform the following iterations

for $i = 1, \ldots, 1000$ **do**
  Sample $\widehat{\mathbf{w}}$ from $\mathcal{W}$
  if $\widehat{\mathbf{w}}$ does not belong to any $\mathcal{R}$ in $\mathcal{CR}$ **then**
    Solve DC-OPF on $\widehat{\mathbf{w}}$, compute $\mathcal{R}$ and add to $\mathcal{CR}$
  end if
end for

Q1. What is the complexity of this algorithm?

- 7 distinct critical regions, each having a fixed set of active constraints





Q2. How many DC-OPF problems did we solve in this case?

MPP-Powered SGD for decision-focused learning

- Machine learning model $\widehat{\mathbf{w}} = \mathbb{W}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{\varphi}})$
- For a given $\widehat{\mathbf{w}}$, the primal solution and the gradient are

$$\mathbf{p} = \boldsymbol{\Pi}\widehat{\mathbf{w}} + \boldsymbol{\pi} \qquad \frac{\partial \mathbf{p}^\star}{\partial \widehat{\mathbf{w}}} = \boldsymbol{\Pi}^\top$$

- Decision-focused problem minimizing the primal decision error

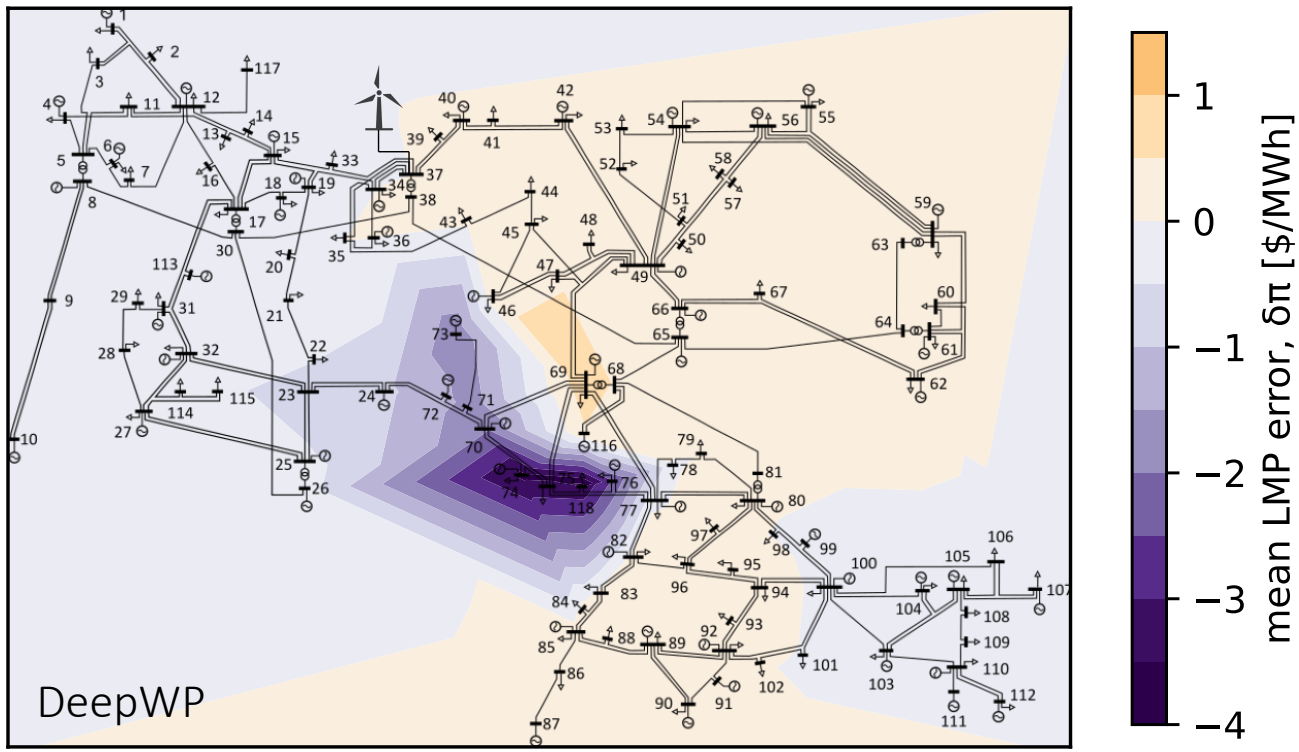
$$\underset{\boldsymbol{\theta}}{\text{minimize}} \quad \mathcal{R} \triangleq \frac{1}{2k} \sum_{i=1}^{k} \|\mathbf{p}^\star(\mathbb{W}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{\varphi}}_i)) - \mathbf{p}_i^\star\|_2^2$$

- The gradient update in SGD for a particular sample i takes the form:

$$\begin{aligned}\frac{\partial \mathcal{R}}{\partial \boldsymbol{\theta}} &= (\mathbf{p}^\star(\mathbb{W}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{\varphi}}_i)) - \mathbf{p}_i^\star) \cdot \frac{\partial \mathbf{p}^\star}{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)} \cdot \frac{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)}{\partial \boldsymbol{\theta}} \\ &= (\boldsymbol{\Pi}\mathbb{W}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{\varphi}}_i) + \boldsymbol{\pi} - \mathbf{p}_i^\star) \cdot \boldsymbol{\Pi}^\top \cdot \frac{\partial \mathbb{W}_{\boldsymbol{\theta}}(\boldsymbol{\varphi}_i)}{\partial \boldsymbol{\theta}}\end{aligned}$$

- No need to solve optimization if we know the critical region of $\mathbb{W}_{\boldsymbol{\theta}}(\widehat{\boldsymbol{\varphi}}_i)$!
- Significant time saving after we explore a sufficient number of critical regions
- By analogy, we can minimize the dual decision error

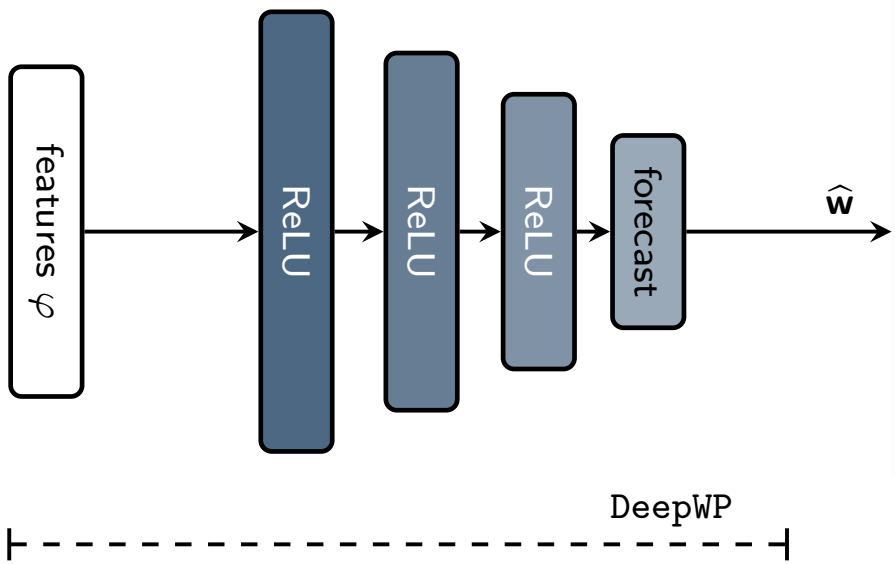
Example (cont'd): Wind power forecast & electricity pricing



Example (cont'd): Price-awareness for wind power forecast

- Dataset $\{(\boldsymbol{\varphi}_1, \mathbf{w}_1), \ldots, (\boldsymbol{\varphi}_k, \mathbf{w}_k)\}$ of wind power records
- Two deep learning architectures `DeepWP` and `DeepWP+` for forecasting[2]

loss function: $\|\widehat{\mathbf{w}} - \mathbf{w}\|$

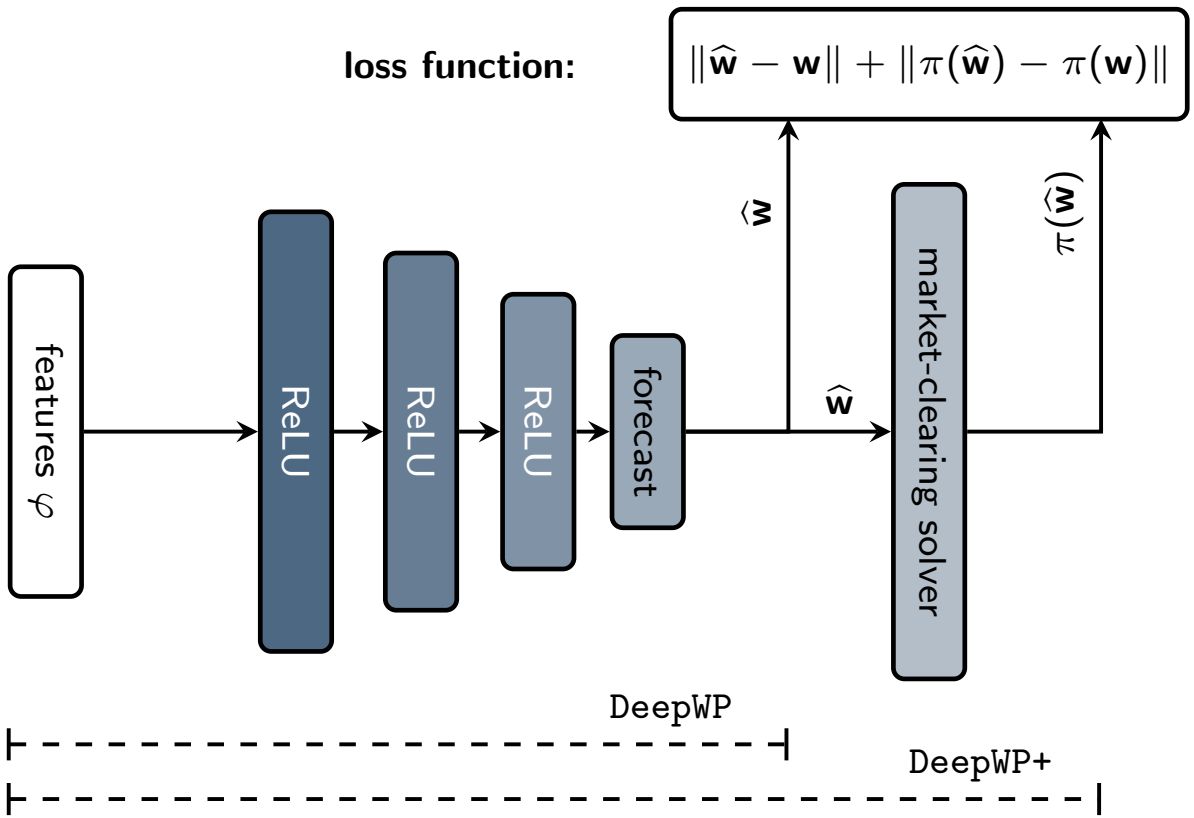


- DeepWP+ informs wind power predictions about the downstream pricing errors

[2] Dvorkin V., & Fioretto F. Price-aware deep learning for electricity markets. 2023

Example (cont'd): Price-awareness for wind power forecast

- Dataset $\{(\boldsymbol{\varphi}_1, \mathbf{w}_1), \ldots, (\boldsymbol{\varphi}_k, \mathbf{w}_k)\}$ of wind power records
- Two deep learning architectures `DeepWP` and `DeepWP+` for forecasting[2]

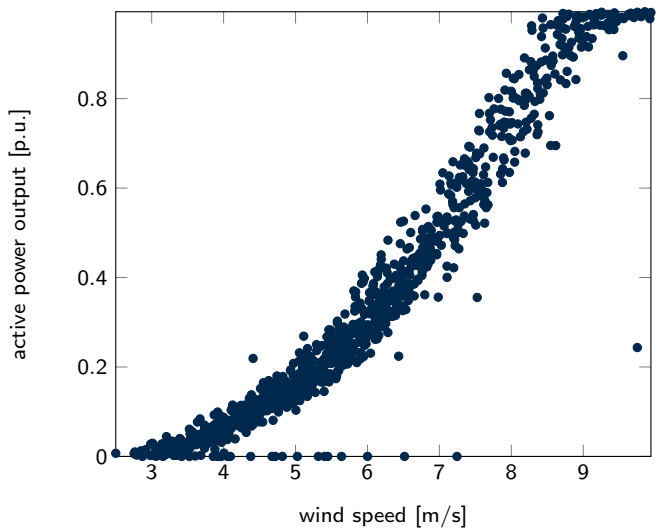


- `DeepWP+` informs wind power predictions about the downstream pricing errors

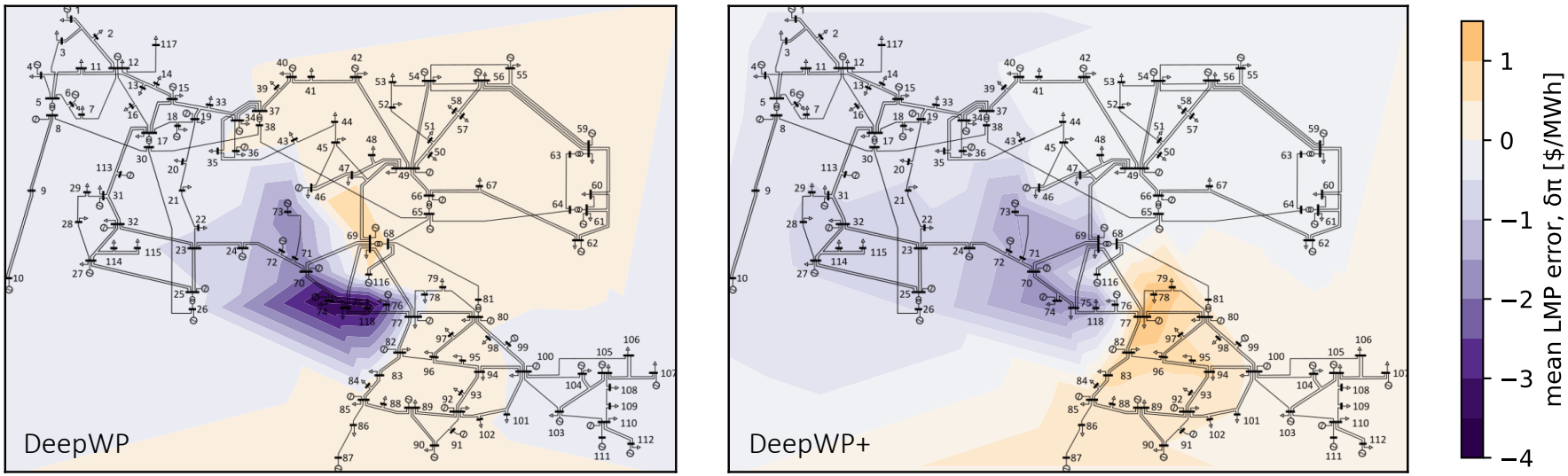
[2] Dvorkin V., & Fioretto F. Price-aware deep learning for electricity markets. 2023

Example (cont'd): Numerical experiments

- Standard `PowerModels.jl` test cases
- 1,000 wind power records
  - Active power output
  - Wind speed and direction
  - Blade pitch angle
- `DeepWP` has 4 hidden layers with 30 neurons each. `DeepWP+` additionally includes an opt. layer
- ADAM optimizer with varying learning rate



Example (cont'd): IEEE 118-bus system



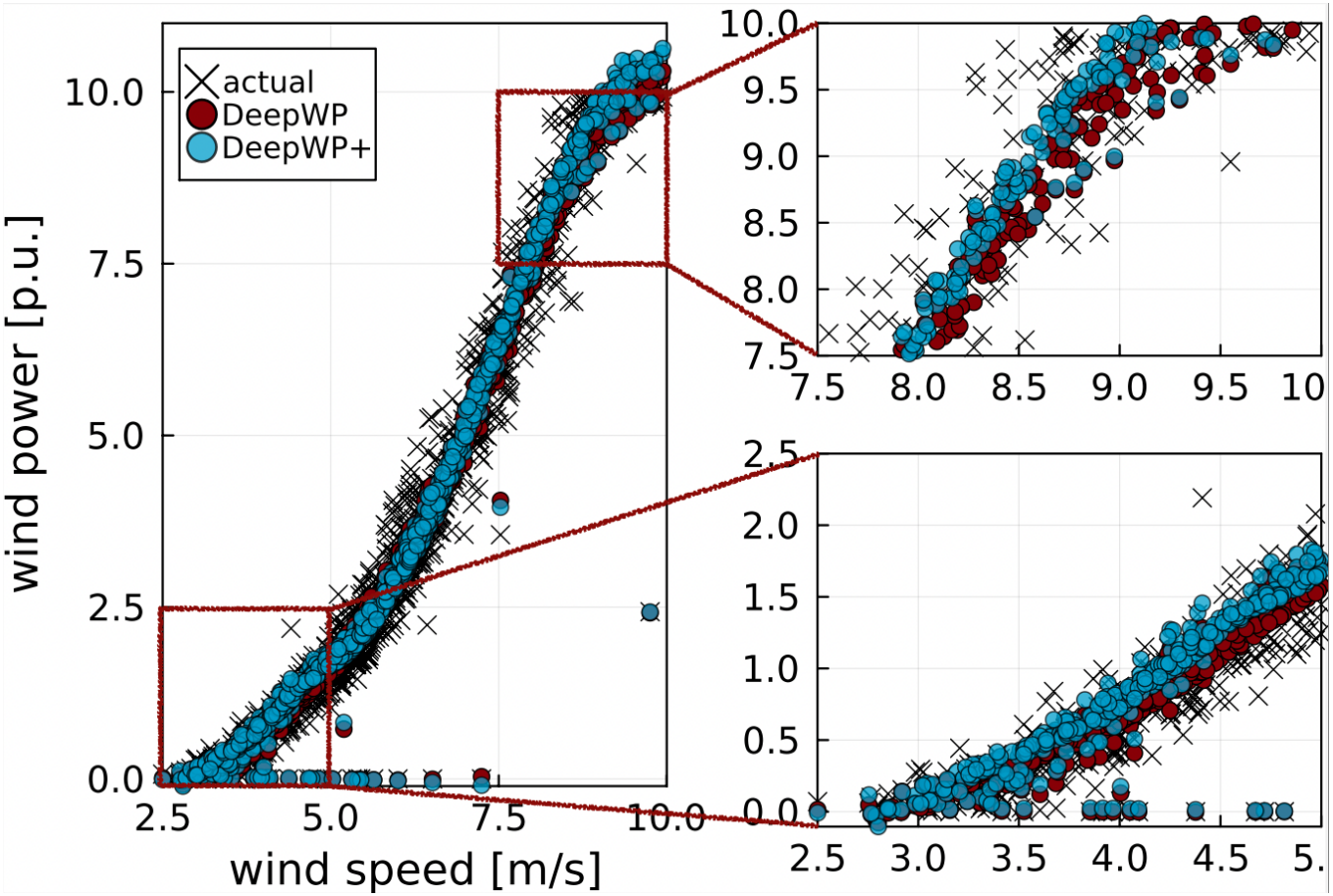
`DeepWP:` Forecast error minimization yields $\delta\pi \in [-4, 1]$ \$/MWh

`DeepWP+:` Price error minimization yields $\delta\pi \in [-1, 1]$ \$/MWh

Q1. Which prediction is fairer?

Q2. What prevents us from completely eliminating the price error?

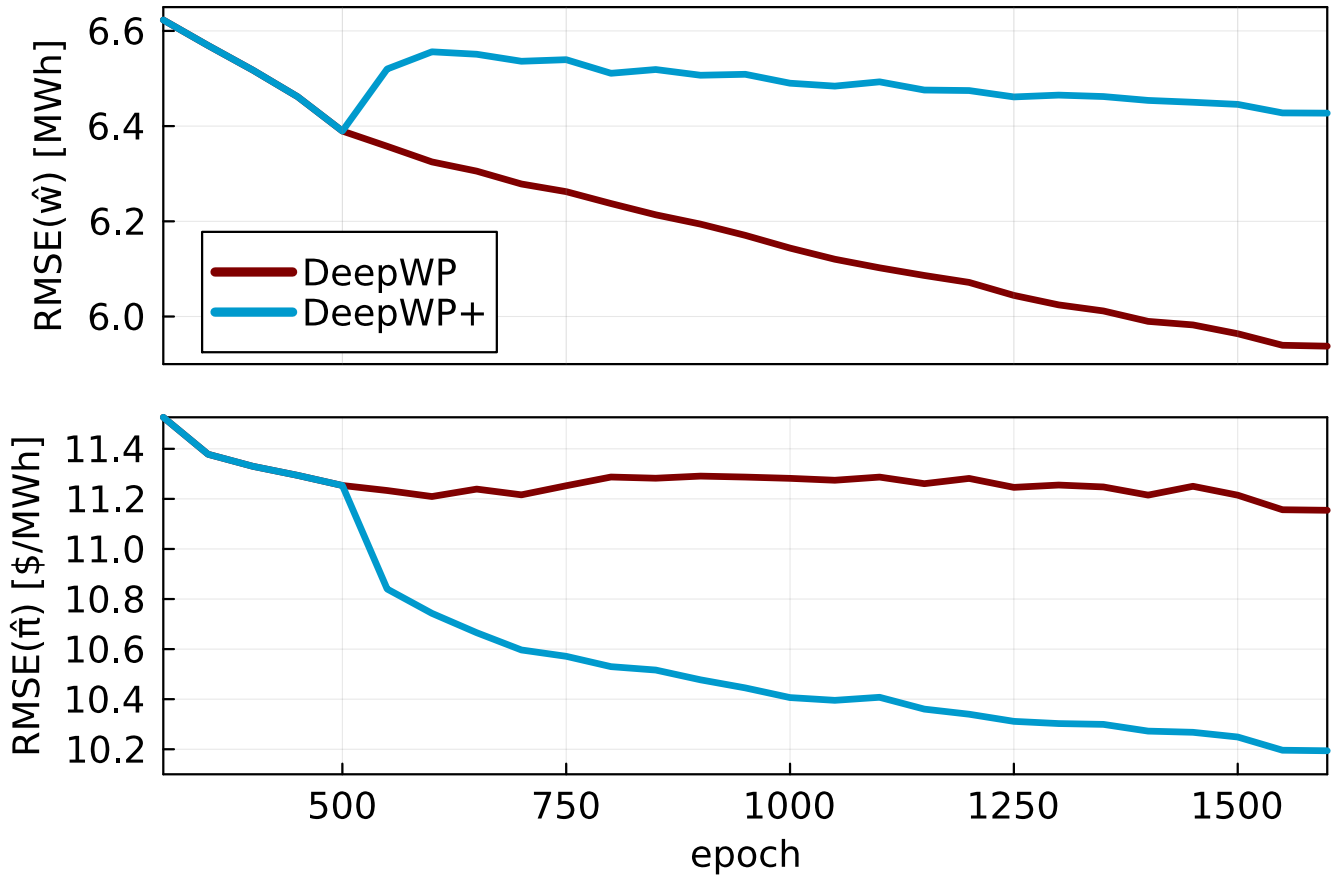
Example (cont'd): Wind power forecasts



`DeepWP:` Minimizes the average forecast deviation

`DeepWP+:` Intentionally over-predicts in certain range of wind speeds

Example (cont'd): Bias of DeepWP+ model



- DeepWP+ training starts at iteration 500 using a pre-trained DeepWP model
- RMSE($\widehat{w}$) and RMSE($\widehat{\pi}$) are conflicting objectives which are kept in balance

Example (cont'd): Forecast and price error trade-off

case	DeepWP				DeepWP+							
	RMSE($\widehat{w}$)	RMSE($\widehat{\pi}$)	CVaR($\widehat{\pi}$)	$\alpha-$value	RMSE($\widehat{w}$)		RMSE($\widehat{\pi}$)		CVaR($\widehat{\pi}$)		$\alpha-$value	
	MWh	$/MWh	$/MWh	$/MWh	MWh	gain	$/MWh	gain	$/MWh	gain	$/MWh	gain
14_ieee	0.35	0.62	1.52	0	0.35	+0.6%	0.61	**−0.6%**	1.50	−0.8%	0	—
57_ieee	2.31	11.03	34.64	32.08	2.60	+11.2%	10.72	**−2.9%**	33.59	−3.1%	30.92	−3.8%
24_ieee	4.08	8.62	37.70	27.48	4.51	+9.6%	8.33	**−3.5%**	36.35	−3.7%	26.26	−4.6%
39_epri	5.94	11.15	31.21	17.53	6.43	+7.6%	10.19	**−9.4%**	28.02	−11.4%	15.84	−10.7%
73_ieee	4.02	5.12	16.21	32.83	5.51	+26.9%	4.24	**−20.8%**	13.41	−20.9%	26.63	−23.3%
118_ieee	2.29	3.59	11.32	17.91	2.60	+12.1%	2.88	**−24.7%**	9.06	−25.0%	14.09	−27.2%

- Worst-case improvement exceeds that of the average case
- Price error reduction and fairness improves with the size of the network

Resources

- Mandi, J. *et al.* (2024). Decision-focused learning: Foundations, state of the art, benchmark and future opportunities. Journal of Artificial Intelligence Research, 80, 1623-1701.

- Amos, B., & Kolter, J. Z. (2017, July). Optnet: Differentiable optimization as a layer in neural networks. In International conference on machine learning (pp. 136-145).

- Tøndel, P., Johansen, T. A., & Bemporad, A. (2003). An algorithm for multi-parametric quadratic programming and explicit MPC solutions. Automatica, 39(3), 489-497.

- Dvorkin, V., & Fioretto, F. (2023). Price-aware deep learning for electricity markets. arXiv preprint arXiv:2308.01436.