

Diagnosing and Mitigating Infeasibilities in Power Grid Optimization

Using Machine Learning and Counterfactual Explanations

Presenter: Dr. Kyri Baker

Work led by PhD students Mostafa Mohammadian and Anna Van Boven



University of Colorado **Boulder**



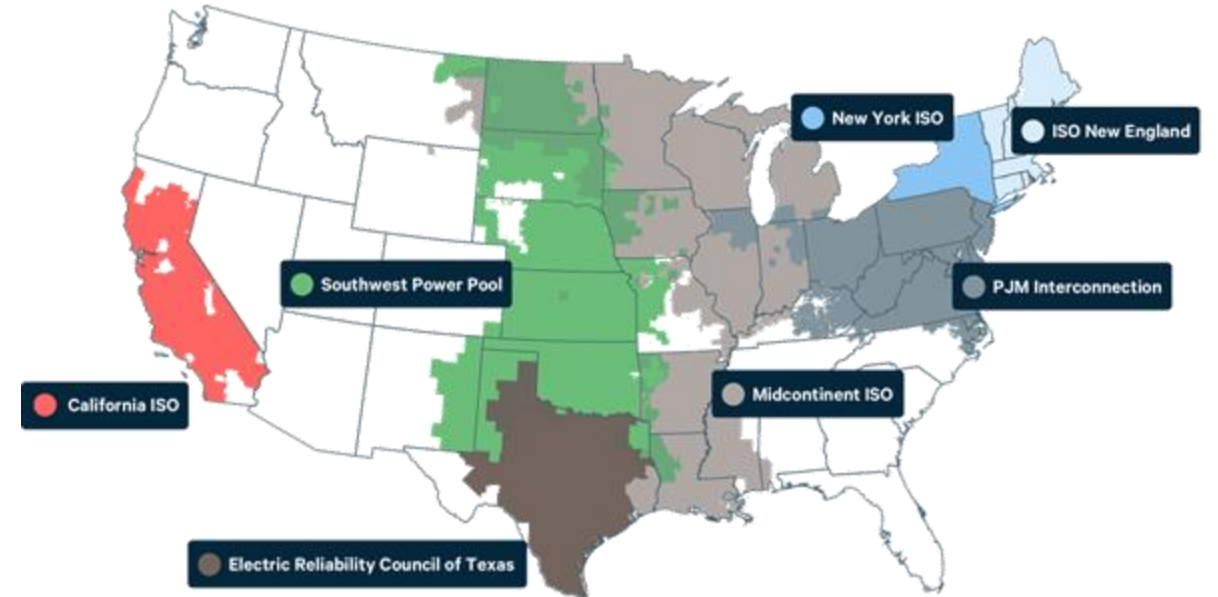
The optimization at the core of the grid

1960's: Carpentier formalized the idea of “optimal power flow (**OPF**)” in electricity grids.

1990's: Deregulation of power grid operations began in the United States.

Varying levels of deregulation and market participation exist throughout the country.

Seven main **independent system operators (ISOs)** run competitive wholesale power markets which **run an optimization problem** to match supply and demand economically and reliably by solving **OPF**



Source: Homeland Infrastructure Foundation-Level Data (2019)

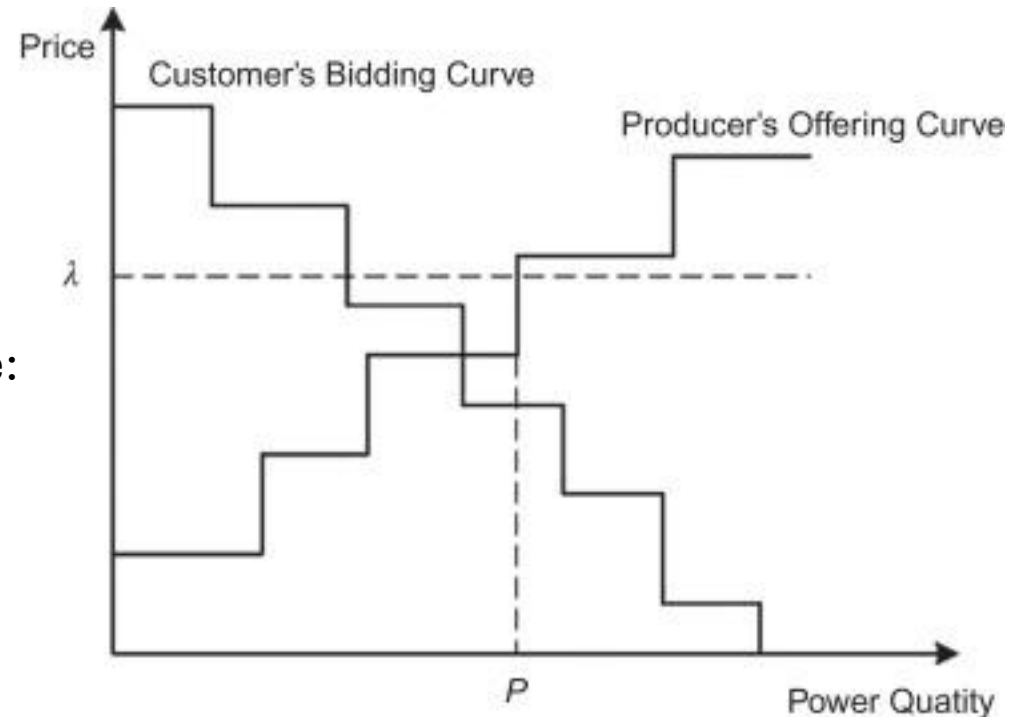
The US National Academy of Engineering ranks electric power networks as **the number 1 greatest engineering achievement of the 20th century.**

If only it were this simple...

The market is “cleared” by solving a (linear) optimization problem which minimizes the cost of power generation.

This is different than normal “supply and demand” clearing in economics – here, we need a full optimization problem with **constraints** to adhere to the physical limits of the grid.

If only it were this simple:

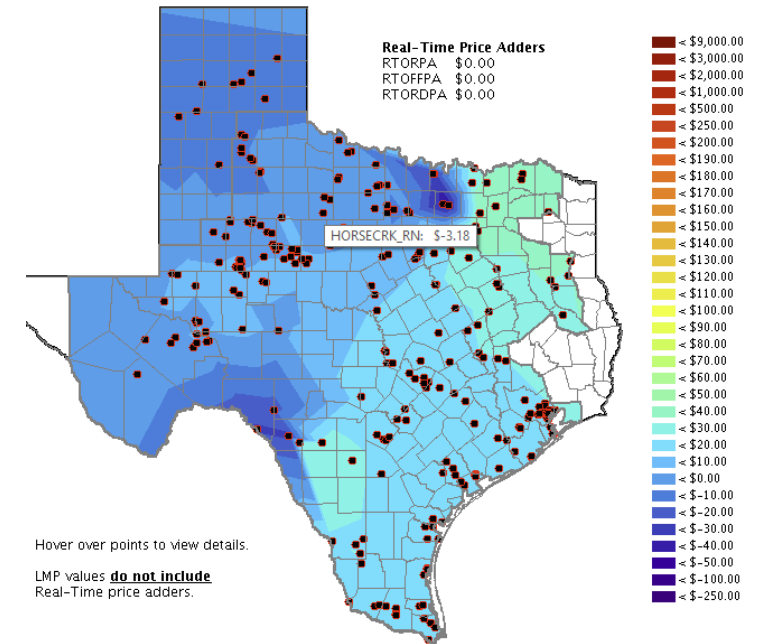


Power grids are uniquely weird

Power grids have the only markets where **prices go negative every day**

This has happened in some other situations in history..

Like with future prices for oil for the first time in history in 2020



US oil prices turn negative

Price per barrel of WTI



Source: Bloomberg, 20 April 2020, 20:15 GMT

BBC

Onions in 1956



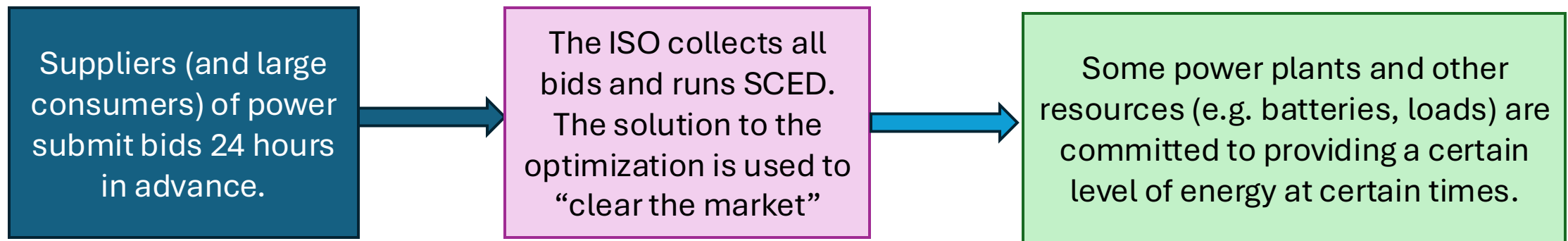
“Security Constrained Economic Dispatch”

Optimal power flow (OPF) is sometimes called by this name (“SCED” for short).

Basic Economic Dispatch doesn’t include grid constraints and was solved by hand in the 1930’s²

OPF’s first “full formulation” was the one by Carpentier (1962).

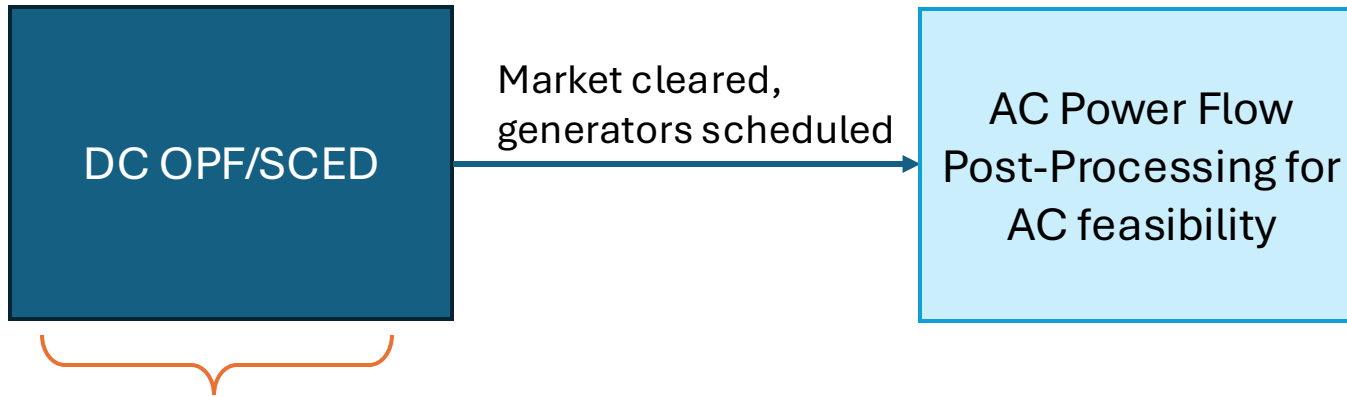
(Day Ahead Markets)



² M. Cain, R. O’Neill, A. Castillo, *History of Optimal Power Flow and Formulations*, FERC Staff Paper, 2012.

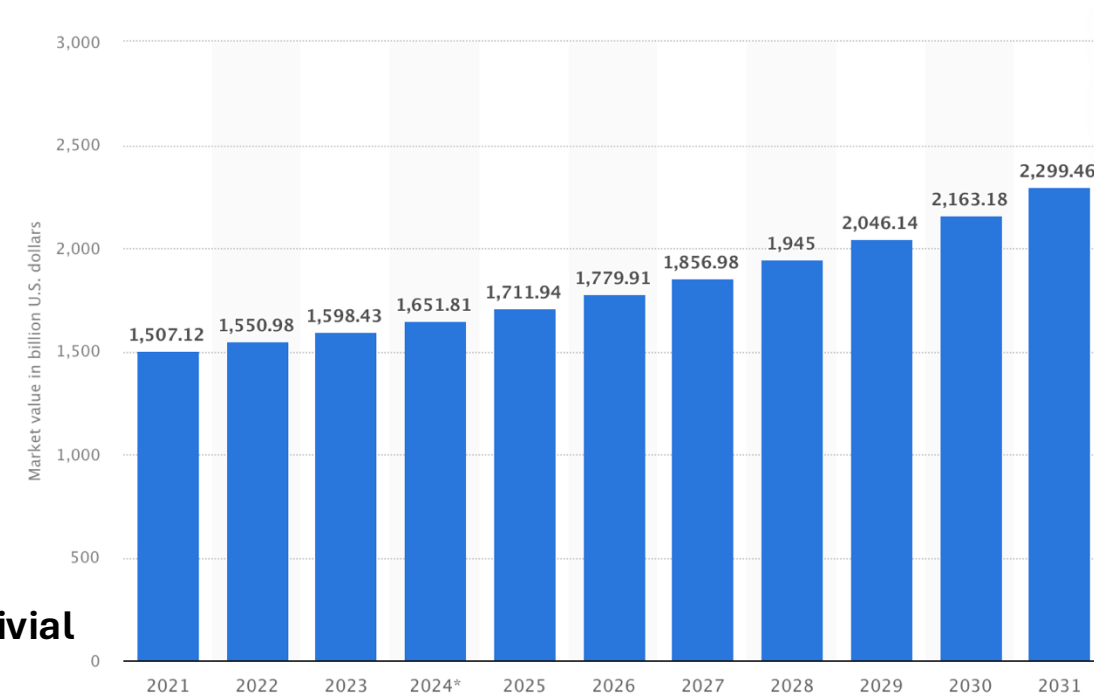
The multi-trillion dollar power market

This optimization problem, and its solution, impacts a **multi-trillion dollar market** in the U.S.



How do we make sure this actually solves?

Even though DC OPF is a convex optimization, for real grids, it's still nontrivial



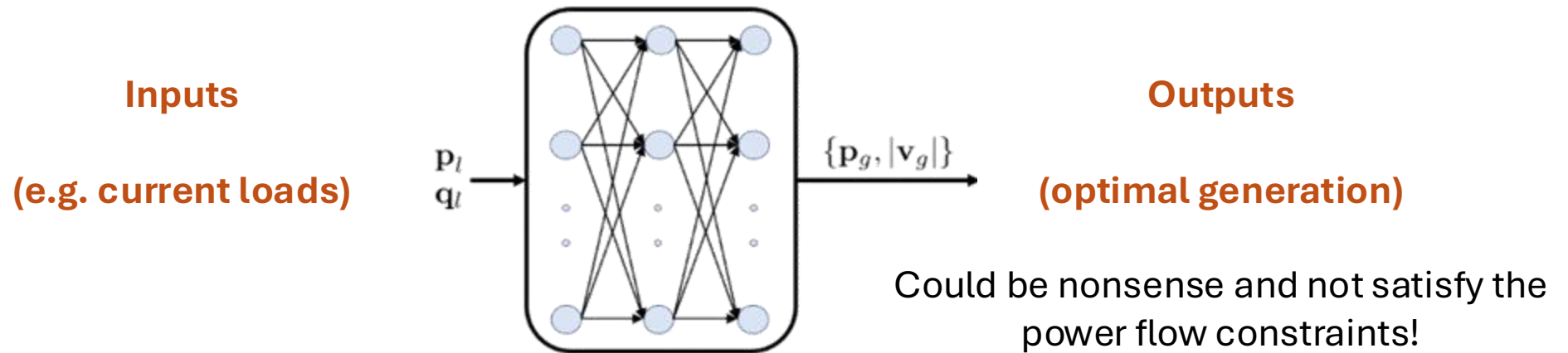
Market value of electricity generation worldwide

Source: Statista

ML for OPF

There have been a lot of works (ours included) looking at how we can solve OPF on very fast timescales

But none of these models (to my knowledge) tell you when the **inputs are infeasible**



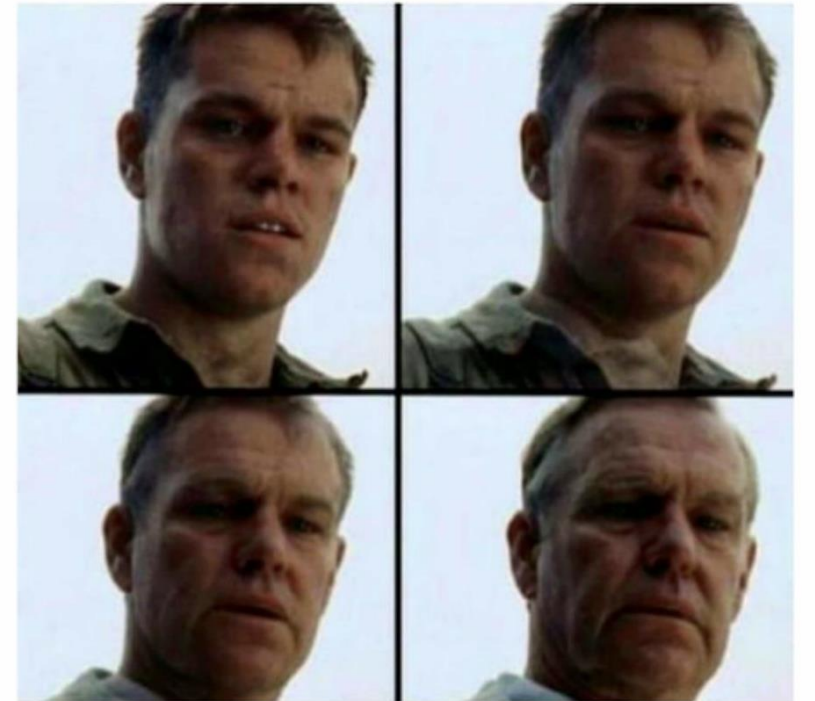
We need models which can a) tell us when OPF is infeasible and b) helps us fix it

a) Tell us when the OPF is infeasible

Wait! Can't you use an existing solver (e.g. Gurobi) to determine if an optimization problem is infeasible?

Sure, if you have ***all the time in the world*** , but who does, in this economy

waiting for Newton-Raphson
to reach its max iteration count
before concluding my OPF is infeasible



a) Tell us when the OPF is infeasible

Wait! Can't you use an existing solver (e.g. Gurobi) to determine if an optimization problem is infeasible?

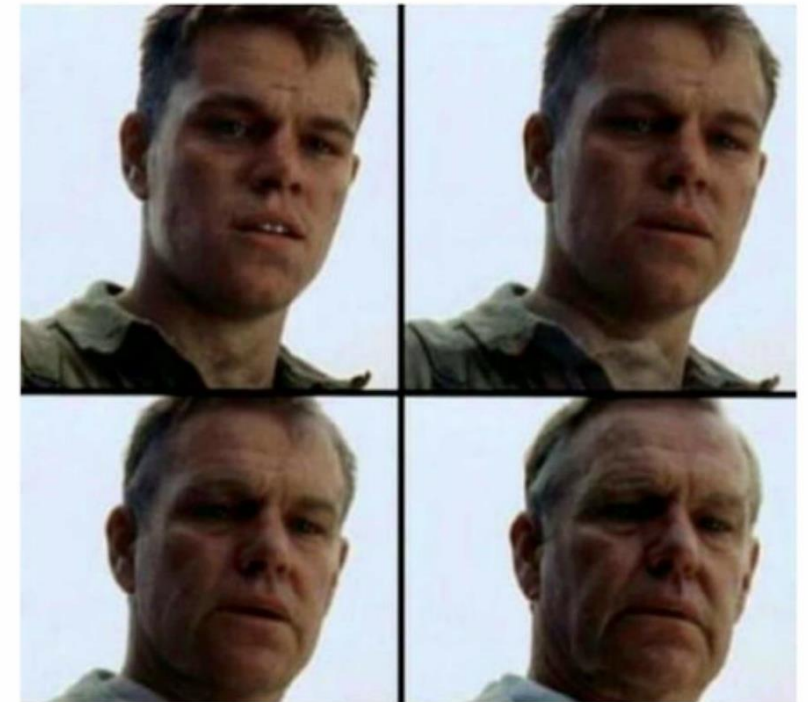
Sure, if you have ***all the time in the world*** , but who does, in this economy

Especially with AC OPF, which is a highly complex, nonconvex optimization, it may take a long amount of time, unsuitable for real-time operation, to fail solving

And these aforementioned ML-for-OPF regression models can solve AC OPF blazingly fast but will always produce a solution – with no **guarantee** of feasibility!

It would be nice to know if a problem formulation is infeasible beforehand without needlessly sending it to a solver

waiting for Newton-Raphson
to reach its max iteration count
before concluding my OPF is infeasible



b) Help us fix the infeasibility

```
RHS range      [2e-03, 1e+04]  
Presolve time: 0.02s  
  
Explored 0 nodes (0 simplex iterations) in 0.06 seconds  
Thread count was 1 (of 8 available processors)  
  
Solution count 0  
  
Model is infeasible
```

Ok...now what?

b) Help us fix the infeasibility

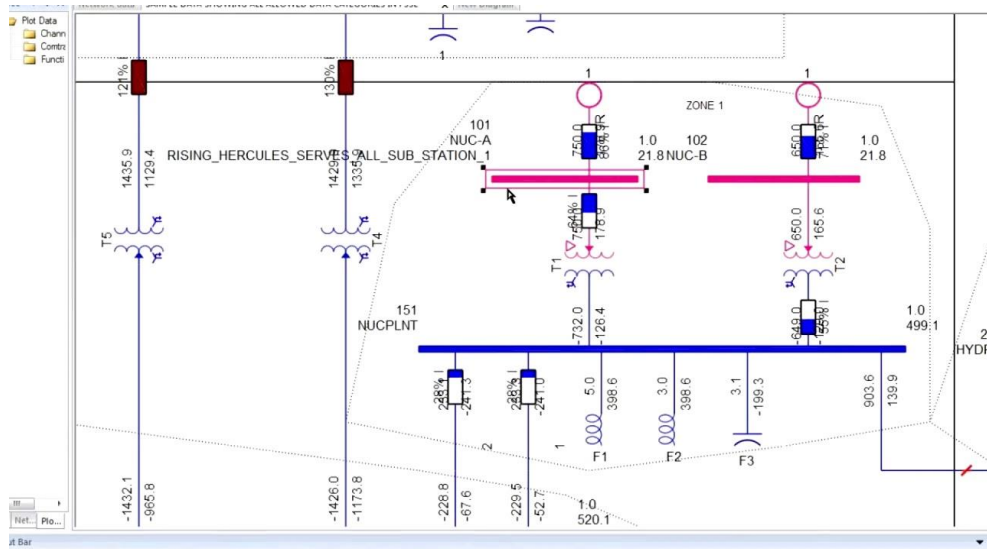
```
RHS range      [2e-03, 1e+04]  
Presolve time: 0.02s  
  
Explored 0 nodes (0 simplex iterations) in 0.06 seconds  
Thread count was 1 (of 8 available processors)  
  
Solution count 0  
  
Model is infeasible
```

Ok...now what?

- We could try to use our intuition to adjust some constraints? (e.g. change line limits to soft constraints)
- We could add slack variables to the problem and resolve?
- We could use a built-in function from an optimizer like Gurobi's **feasrelax**?

Using intuition to adjust some constraints

- Great if you are a seasoned grid operator and know where the likely source of the infeasibility is
- Not as straightforward for everyone. And is the human picking the best “fix”?
- What if there’s an infeasibility that requires load shed or demand response? We can choose this manually (raising issues of equity) or through solving *another* optimization problem?



Maybe I'll change that transformer tap setting? That worked last time..

Using slack variables or functions like **feasrelax**

Slack variables are famously **slow** – we have to add a variable for every constraint in the problem, and more constraints

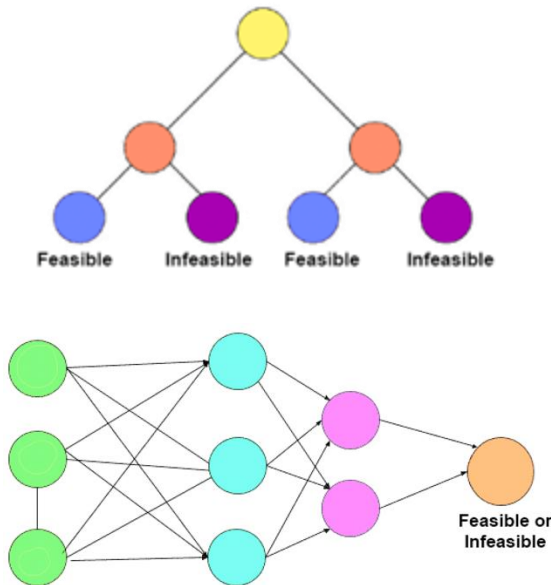
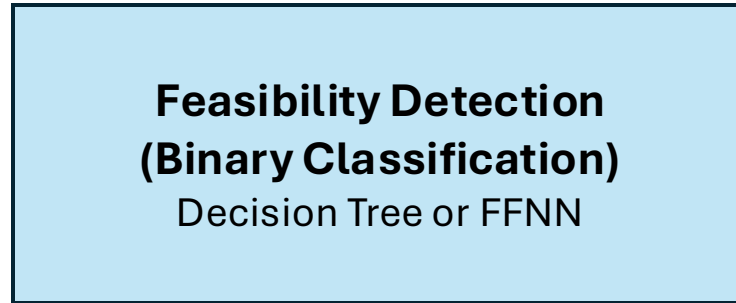
Feasrelax and other feasibility “fixing” techniques often use slack variables.

Even limiting the number of these variables can add significant model complexity

$$\begin{array}{ccc} \min_{\mathbf{x}} f(\mathbf{x}) & & \min_{\mathbf{x}, \mathbf{s}} f(\mathbf{x}) + g(\mathbf{s}) \\ h(\mathbf{x}) \leq \mathbf{0} & \xrightarrow{\hspace{1cm}} & \begin{array}{l} h(\mathbf{x}) \leq \mathbf{s} \\ \mathbf{s} \geq \mathbf{0} \end{array} \end{array}$$

Also, these techniques usually only yield **one** set of slack variables – or one option for making the system feasible

Our solution : Detect and provide fixes



If Infeasible (0)



Counterfactual Analysis
(If some system settings or inputs
would have been different, could
that flip the 0 to a 1?)



**Call on generation reserves,
perform demand response,
load shed (worst case)**

A suite of options that can push us to feasibility

Wait. Isn't there an “optimal” counterfactual?

$$\min_{\mathbf{x}, \mathbf{s}} f(\mathbf{x}) + g(\mathbf{s})$$

$$h(\mathbf{x}) \leq \mathbf{s}$$

$$\mathbf{s} \geq \mathbf{0}$$

Isn't the 'optimal' decision the one which minimizes the amount of slack added to the original problem?

Not necessarily.

Wait. Isn't there an “optimal” counterfactual?

$$\min_{\mathbf{x}, \mathbf{s}} f(\mathbf{x}) + g(\mathbf{s})$$

$$h(\mathbf{x}) \leq \mathbf{s}$$

$$\mathbf{s} \geq \mathbf{0}$$

Isn't the 'optimal' decision the one which minimizes the amount of slack added to the original problem?

Not necessarily.

The “optimal” way to load shed, for example, may hurt more communities more than others

Mathematical optimality \neq Social optimality



SHARE



This is an archived article and the information in the article may be outdated. Please look at the time stamp on the story to see when it was last updated.

When the nation's largest utility warned customers that it would cut power to nearly 2 million people across Northern California, many rushed out to buy portable generators, knowing the investment could help sustain them during blackouts.

(Also, for large systems, adding hundreds or thousands of new variables/constraints is not ideal!)

First step: DC OPF – what operators currently use



Infeasibilities here are due to **line congestion**
Or power lines reaching their thermal limits

Can also have infeasibilities due to too much
load in the system – but this is trivial to predict
(just check if $\sum load > \sum gen$)

Cost function: Minimize cost of power generation

$$\min_{p_g} \sum_{j \in \mathcal{G}} a_j p_{g,j}^2 + b_j p_{g,j} + c_j$$

Power balance at each bus

$$\text{s.t.} : p_{l,i} - \sum_{k \in \mathcal{G}_i} p_{g,k} = \sum_{m \in \mathcal{N}} B_{im} \theta_{im}, \forall i \in \mathcal{N}$$

$$-F_{im} \leq B_{im} \theta_{im} \leq F_{im}, \forall im \in \mathcal{L}$$

Line flow limits

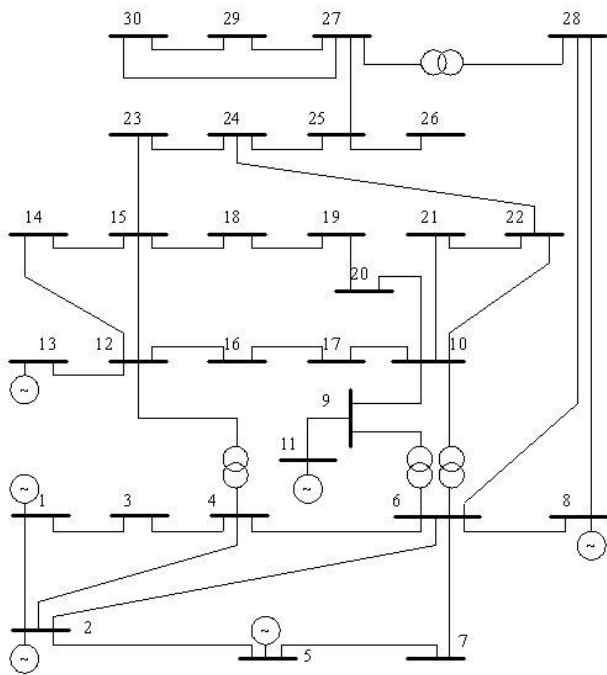
+ generator limits
+ angle constraint

Set of buses

Set of lines

Dataset Creation

First looking at DC OPF, and will move to AC OPF (significantly harder) in future work



30 and 300-bus systems

10,000 load scenarios



Input Data: {load, feasible_flag}

50/50 Class Balance

80/20 Train/Test split

This data is used to train the classifiers

The counterfactual model can be modeled
as an unconstrained optimization or
another ML model

The Counterfactual Model

What modifications to the load vector will result in the classifier prediction changing from a 0 to a 1?

Set of k counterfactuals
(load/gen modification vectors)



$$C(\mathbf{x}) = \arg \min_{\mathbf{c}_1, \dots, \mathbf{c}_k} \left(\frac{1}{k} \sum_{i=1}^k \text{yloss}(f(\mathbf{c}_i), y) \right. \\ \left. + \frac{\lambda_1}{k} \sum_{i=1}^k \text{dist}(\mathbf{c}_i, \mathbf{x}) \right. \\ \left. - \lambda_2 \text{dpp_diversity}(\mathbf{c}_1, \dots, \mathbf{c}_k) \right)$$


We can generate a set of possible counterfactuals that an operator can choose from

(should I shed load at bus 34? Should I bring online additional generation at bus 12?)

The Counterfactual Model

What modifications to the load vector will result in the classifier prediction changing from a 0 to a 1?

Set of k counterfactuals
(load/gen modification vectors)

$$C(\mathbf{x}) = \arg \min_{\mathbf{c}_1, \dots, \mathbf{c}_k} \left(\frac{1}{k} \sum_{i=1}^k \text{yloss}(f(\mathbf{c}_i), y) + \frac{\lambda_1}{k} \sum_{i=1}^k \text{dist}(\mathbf{c}_i, \mathbf{x}) - \lambda_2 \text{dpp_diversity}(\mathbf{c}_1, \dots, \mathbf{c}_k) \right)$$



The loss function here describes how effectively the counterfactuals change the classification from a 0 to a 1

f is a differentiable model (e.g. our classifier)

The Counterfactual Model

What modifications to the load vector will result in the classifier prediction changing from a 0 to a 1?

Set of k counterfactuals
(load/gen modification vectors)

$$C(\mathbf{x}) = \arg \min_{\mathbf{c}_1, \dots, \mathbf{c}_k} \left(\frac{1}{k} \sum_{i=1}^k \text{yloss}(f(\mathbf{c}_i), y) + \frac{\lambda_1}{k} \sum_{i=1}^k \text{dist}(\mathbf{c}_i, \mathbf{x}) - \lambda_2 \text{dpp_diversity}(\mathbf{c}_1, \dots, \mathbf{c}_k) \right)$$


A distance metric that penalizes too much change from our original feature vector \mathbf{x}

We don't want to change the system more than we have to!

The Counterfactual Model

What modifications to the load vector will result in the classifier prediction changing from a 0 to a 1?

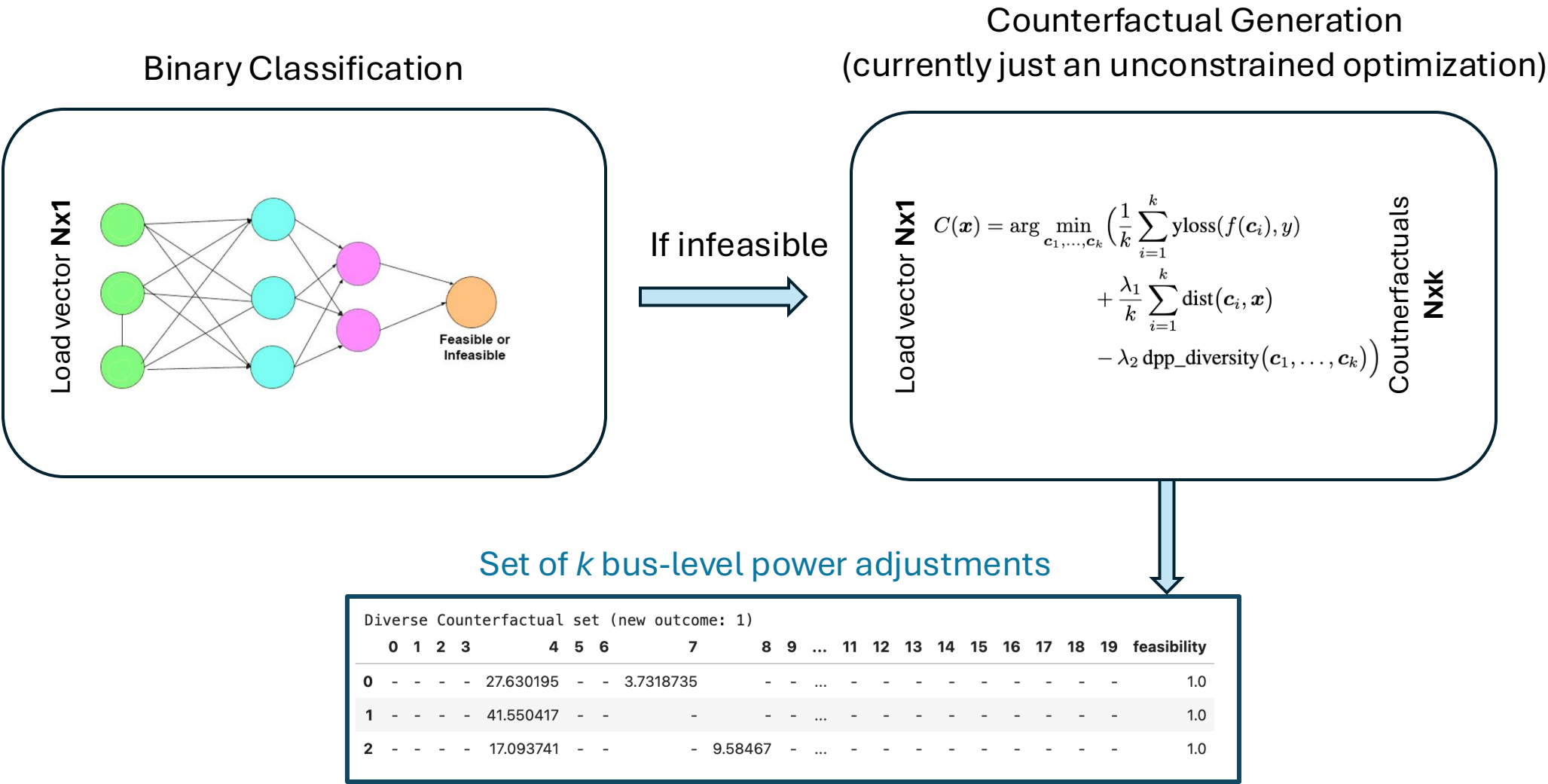
Set of k counterfactuals
(load/gen modification vectors)

$$C(\mathbf{x}) = \arg \min_{\mathbf{c}_1, \dots, \mathbf{c}_k} \left(\frac{1}{k} \sum_{i=1}^k \text{yloss}(f(\mathbf{c}_i), y) + \frac{\lambda_1}{k} \sum_{i=1}^k \text{dist}(\mathbf{c}_i, \mathbf{x}) - \lambda_2 \text{dpp_diversity}(\mathbf{c}_1, \dots, \mathbf{c}_k) \right) \longrightarrow$$

A diversity metric that promotes a wider *variety* of counterfactuals to yield diverse options (rather than a bunch of counterfactuals that are slight perturbations of each other)

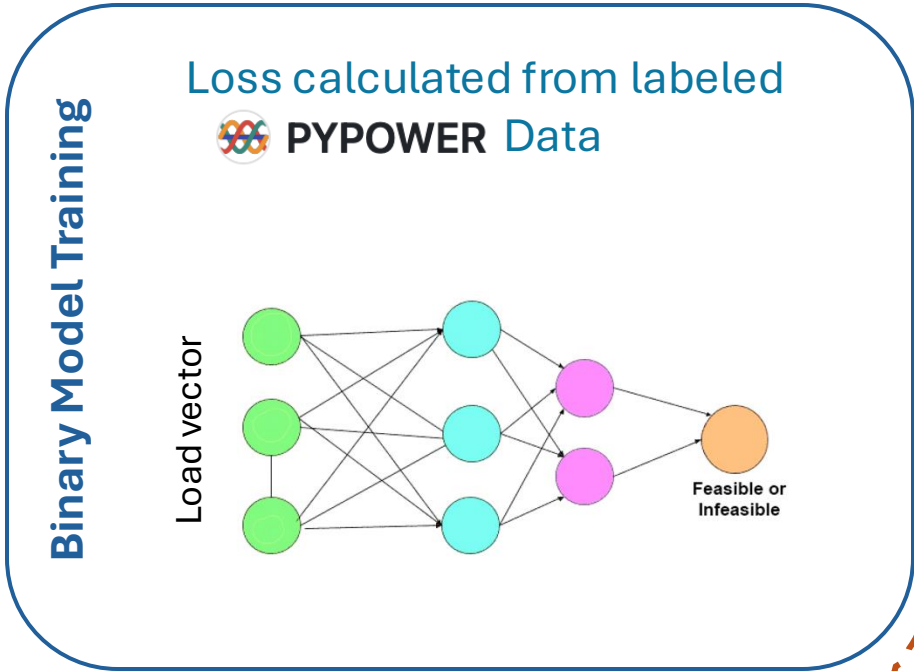
$$\text{dpp_diversity} = \det(\mathbf{K}) \quad \text{where} \quad \mathbf{K}_{i,j} = \frac{1}{1 + \text{dist}(\mathbf{c}_i, \mathbf{c}_j)}$$

Forward Pass

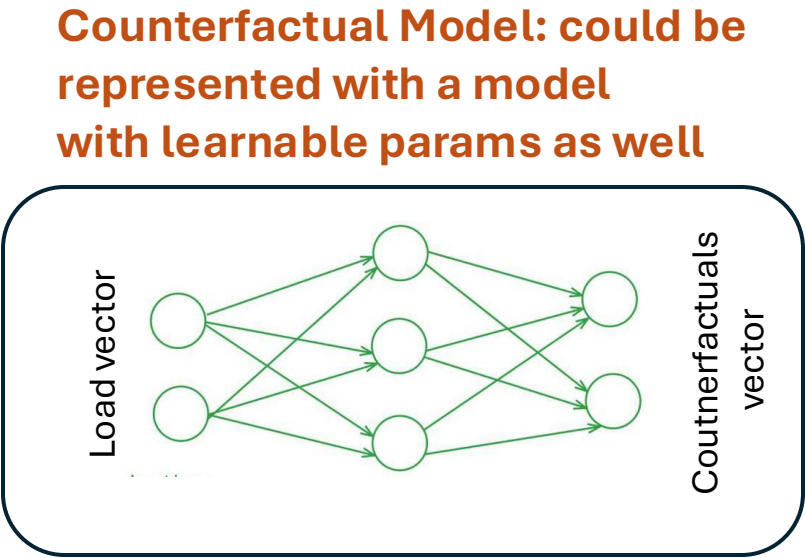


Backward Pass

Loss calculated from original load vector vs. updated,
+ updated feasibility status from classifier model



Updated
feasibility
status



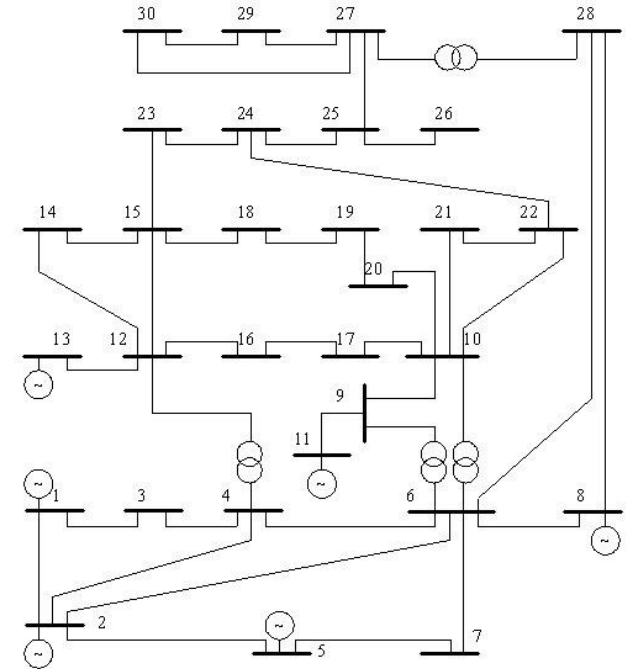
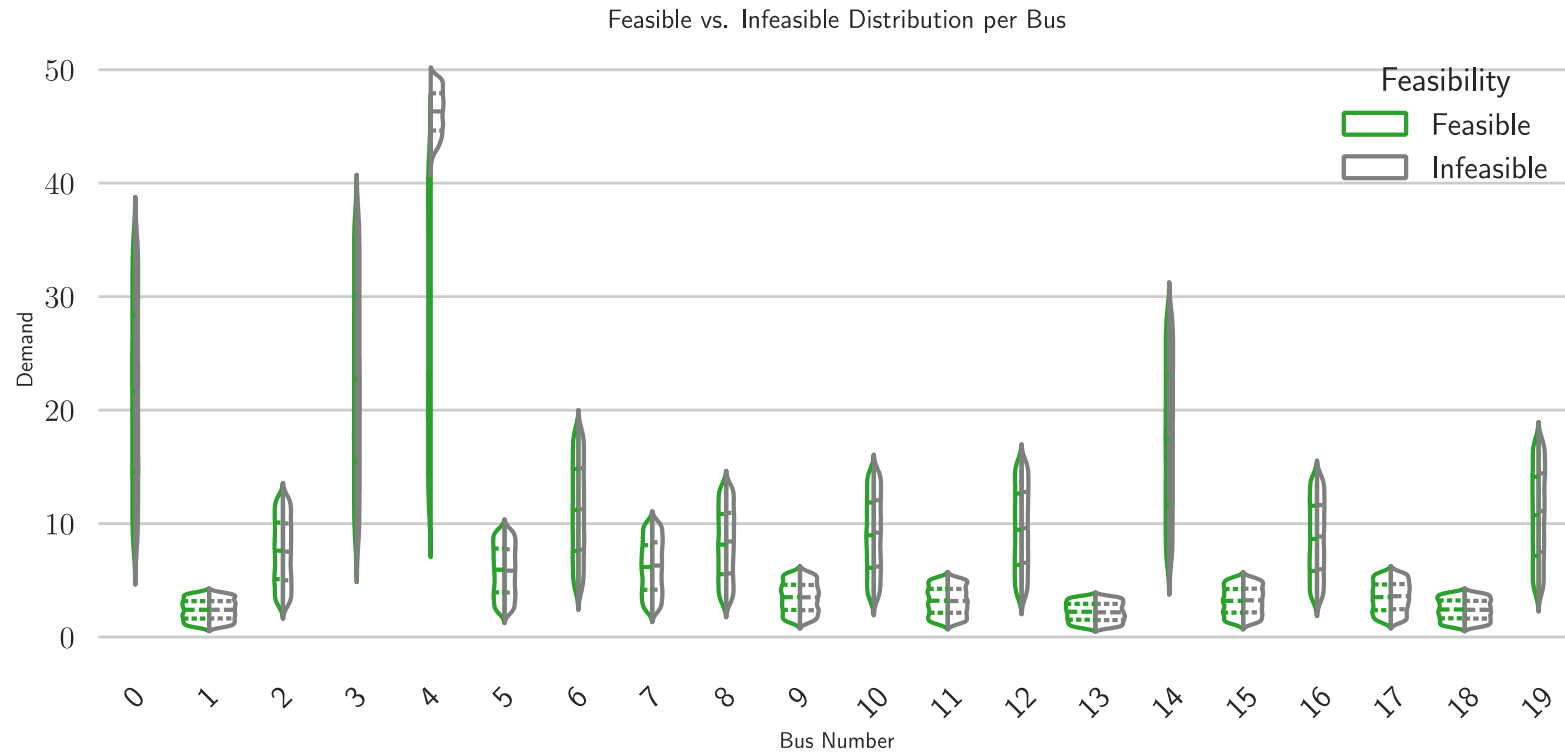
New load vectors
from CF model

Set of k bus-level power adjustments

Diverse Counterfactual set (new outcome: 1)																					
	0	1	2	3	4	5	6	7	8	9	...	11	12	13	14	15	16	17	18	19	feasibility
0	-	-	-	-	27.630195	-	-	3.7318735	-	-	...	-	-	-	-	-	-	-	-	-	1.0
1	-	-	-	-	41.550417	-	-	-	-	-	...	-	-	-	-	-	-	-	-	-	1.0
2	-	-	-	-	17.093741	-	-	-	9.58467	-	...	-	-	-	-	-	-	-	-	-	1.0

Feasible/Infeasible

Nominal loading was perturbed $\pm 65\%$ (30 bus) and $\pm 50\%$ (300 bus) for dataset generation

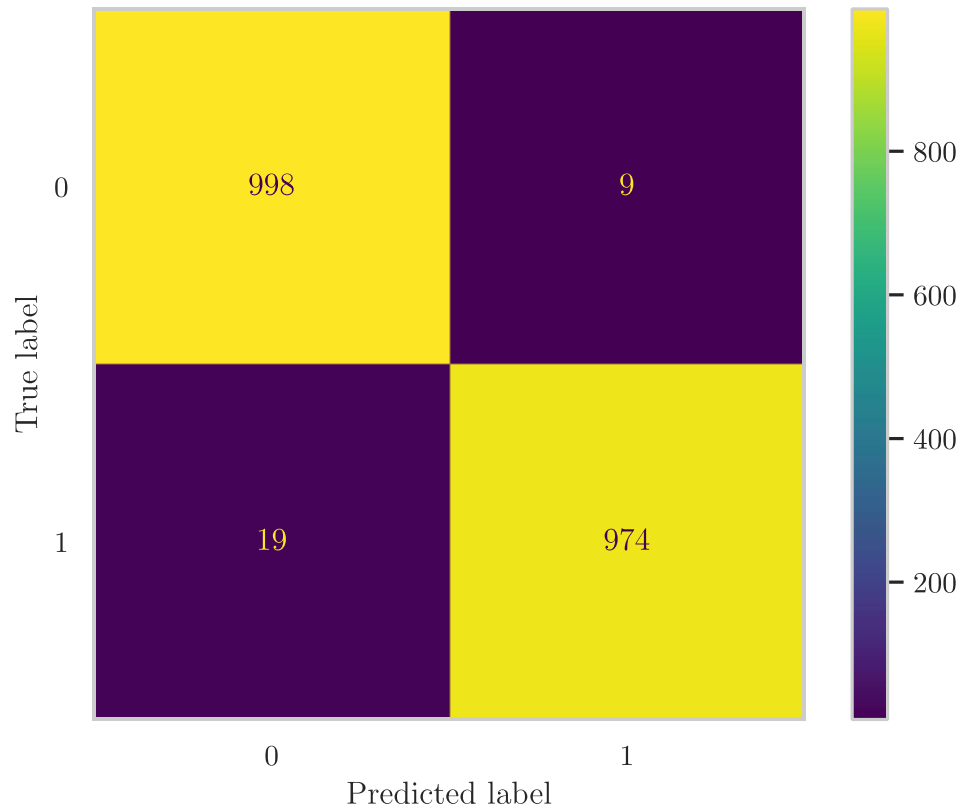


Distribution showing the relationship between load at each bus and overall problem feasibility status – an even distribution of feasible/infeasible except at bus 4

(more complex networks do not have as straightforward of a relationship, but instead feasibility is often a function of the load of multiple buses)

Classification Model Accuracy

Confusion matrix for (30-bus) decision tree:



Method	IEEE 30-bus	IEEE 300-bus
FFNN	99.15%	88.05%
DT	99.0%	84.0%

Decision tree is actually pretty good!


Accuracy decreases as system complexity increases

Accuracy could be improved even further by more rigorous hyperparameter tuning

Feasibility restoration rates per model

Method	IEEE 30-bus	IEEE 300-bus
CF_{FFNN}	99%	100%
CF_{DT}	98%	100%
Gurobi	100%	100%

“Ground-truth” solution which uses
slack variables to guarantee feasibility



These preliminary results indicate that ML models + Counterfactual generation can be used to restore the majority of infeasible DC OPF problems (on the 30 and 300-bus systems) back to feasible, avoiding the use of computationally expensive slack variables or expensive commercial solvers like Gurobi

But for DC OPF on systems this small, there's little to no benefit regarding computation time!

Is there *any* benefit to the ML+CF based approach for small systems?

A suite of feasibility fixes

Adding slack variables to the problem directly (like Gurobi's `feasrelax` does) *can* yield different fixes by changing the objective function to penalize the slack variables accordingly:

$$||x||_1 = \sum_{i=1}^n |x_i|,$$

L-1 norm promotes sparsity
(give me a counterfactual that
modifies as few loads as possible)



We'd ideally want an L-0 norm (not a real norm)
which measures the number of nonzero components
in a vector, but this is not continuous or differentiable

$$||x||_2 = \sqrt{\sum_{i=1}^n |x_i|^2}$$

L-2 norm promotes smallest change
(give me a counterfactual that
is as close to the original solution as possible
in terms of Euclidean distance)

Counterfactuals : Creating Diversity

$$C(\mathbf{x}) = \arg \min_{\mathbf{c}_1, \dots, \mathbf{c}_k} \left(\frac{1}{k} \sum_{i=1}^k \text{yloss}(f(\mathbf{c}_i), y) + \frac{\lambda_1}{k} \sum_{i=1}^k \text{dist}(\mathbf{c}_i, \mathbf{x}) - \lambda_2 \text{dpp_diversity}(\mathbf{c}_1, \dots, \mathbf{c}_k) \right)$$

Using a counterfactual model, we can generate **k different options** for fixing a feasibility, and a human can determine which they'd like (for example)

Lights, Camera, Reaction: Lit-Up Downtown Skylines Are Enraging Powerless Texans

Texans on social media have kept warm by burning the fuel of white-hot rage.



By Dan Solomon

February 17, 2021

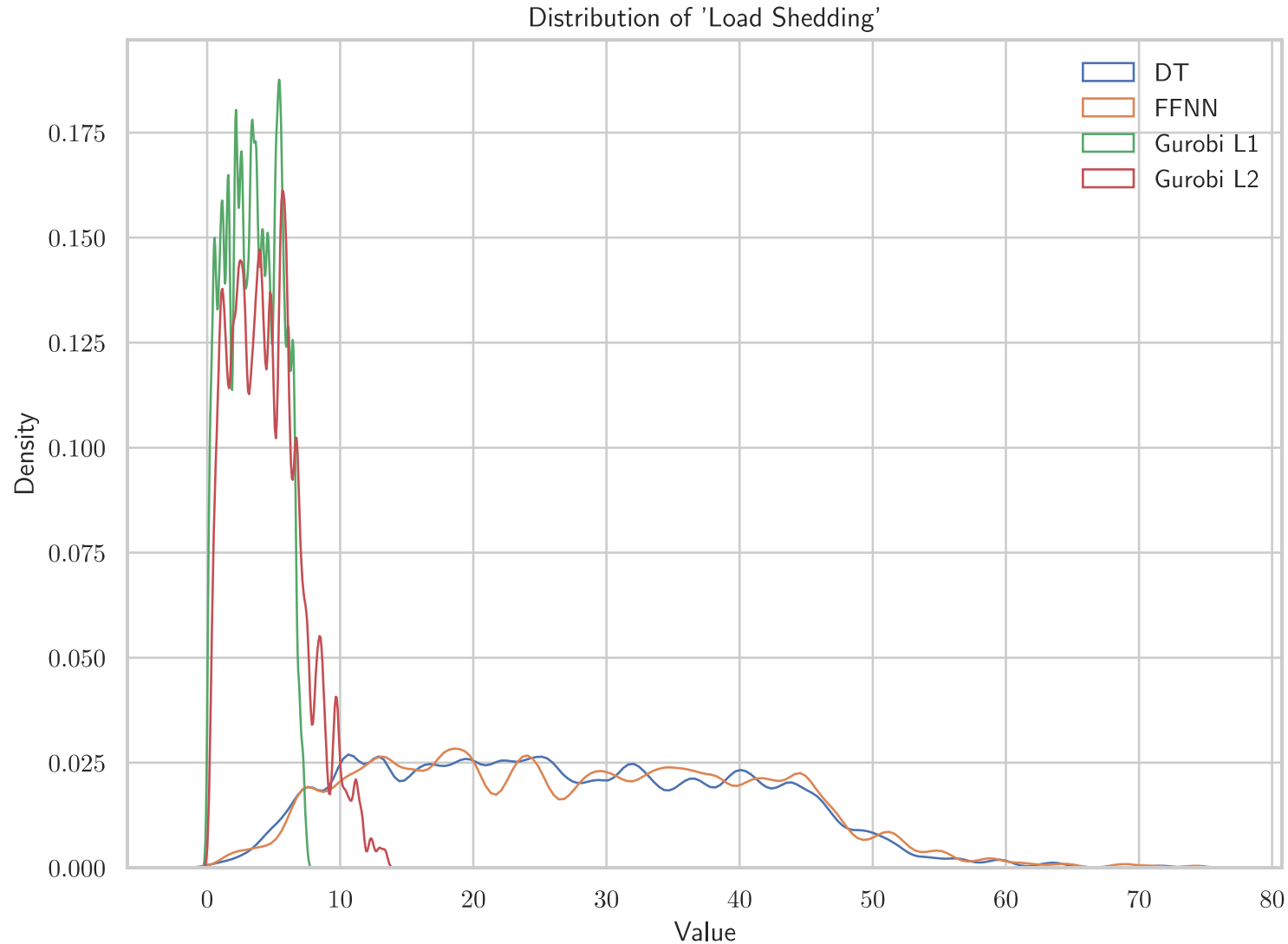


2



(load shedding isn't always perfectly equitable or granular, but you can imagine situations where rotating outages is preferable to keeping the same community out, even if it means a greater amount of load shed)

Counterfactuals : Creating Diversity



Gurobi tends to generate a smaller range of possible bus-level power adjustments, focusing on minimizing the amount rather than creating options

But for smaller networks (like the 30-bus), many of these solutions may be undesirable

Example scenario

	0	1	2	3	4	5	6	7	8	9	Total load reduction or gen increase
CF Option 1	-	-	-	-	27.630195	-	-	3.7318735	-	-	31.36 MW
CF Option 2	-	-	-	-	41.550417	-	-	-	-	-	41.55 MW
CF Option 3	-	-	-	-	17.093741	-	-	-	9.58467	-	26.67 MW

Hmm...option 3 results in the lowest change, but customers at bus 8 have experienced a lot of outages lately.

I could call on a lot of demand response for the industrial customer at bus 4, or use CF Option 1 and call on a smaller amount of DR for customer 4 and use some storage reserves at bus 7.



Any of these options will help keep the lights on!

Future Work

- **Wider ranges of feasibility adjustments**

→ Incorporation of other system changes (e.g. transformer tap settings) rather than just load/generation adjustments

- **Extension to AC OPF**

→ Fixing feasibilities in AC OPF is extremely hard

→ Feasrelax can run for hours just to fix one instance of AC OPF

- **End-to-End learning**

→ Train both the classifier and the counterfactual model in one end-to-end training pipeline

Thank you!

Questions?



www.kyrib.com
kyri@colorado.edu