European Control Conference 2020
Workshop WeWC3

From Decision Trees and Neural Networks to MILP: Power System Optimization Considering Dynamic Stability Constraints

Spyros Chatzivasileiadis
Associate Professor

Main takeaway

Intractable/Non-linear Optimization Problem

$$\min_{\mathbf{x}} f(\mathbf{x})$$

linear bounds $\mathbf{x}_{\min} \leq \mathbf{x} \leq \mathbf{x}_{\max}$

linear constraints $\mathbf{A}\mathbf{x} = \mathbf{b}$

non-linear inequality constraints $\mathbf{g}(\mathbf{x}) \leq \mathbf{0}$

Intractable constraints e.g. based on differential equations, dynamic stability, etc $\phi(\mathbf{x}) \in \mathbf{S}$

Main takeaway

DT: Decision Tree
NN: Neural Network

Intractable/Non-linear Optimization Problem

$$\min_{\mathbf{x}} f(\mathbf{x})$$

linear bounds $\mathbf{x}_{\min} \leq \mathbf{x} \leq \mathbf{x}_{\max}$

linear constraints $\mathbf{A}\mathbf{x} = \mathbf{b}$

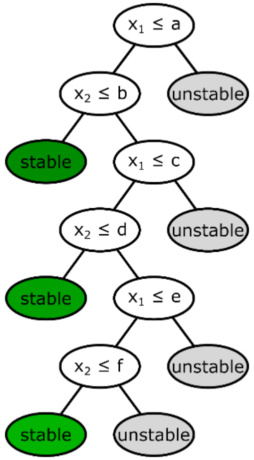
non-linear inequality constraints $\mathbf{g}(\mathbf{x}) \leq \mathbf{0}$

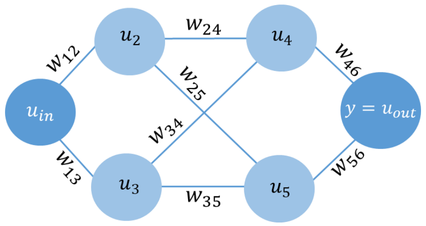
Intractable constraints $\phi(\mathbf{x}) \in \mathbf{S}$

e.g. based on differential equations, dynamic stability, etc

Encode the feasible space to a DT or NN

Classify: feasible/infeasible





Main takeaway

DT: Decision Tree
NN: Neural Network
MILP: Mixed-Integer Linear Program

Intractable/Non-linear Optimization Problem

$$\min_{\mathbf{x}} f(\mathbf{x})$$

linear bounds $\mathbf{x}_{\min} \leq \mathbf{x} \leq \mathbf{x}_{\max}$

linear constraints $\mathbf{A}\mathbf{x} = \mathbf{b}$

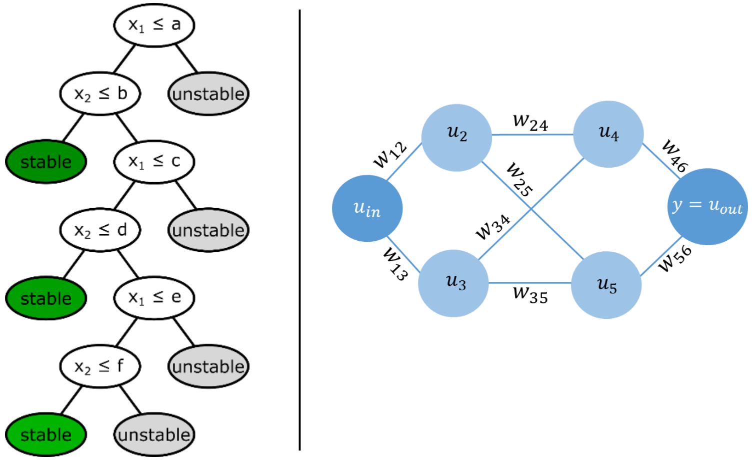
non-linear inequality constraints $\mathbf{g}(\mathbf{x}) \leq \mathbf{0}$

Intractable constraints
e.g. based on differential equations, dynamic stability, etc

$\phi(\mathbf{x}) \in \mathbf{S}$

Encode the feasible space to a DT or NN

Classify: feasible/infeasible



Exact Transformation: Convert DT or NN to a MILP

$$\min_{\mathbf{x}} f(\mathbf{x})$$

$$\mathbf{x}_{\min} \leq \mathbf{x} \leq \mathbf{x}_{\max}$$

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$

MILP Constraints
(Exact transformation)

Capture previously intractable constraints and solve a MILP

Outline

- Guiding Example: Dynamic Stability Constrained Optimal Power Flow
- From Decision Trees to MILP
- From Neural Networks to MILP

What is the problem?

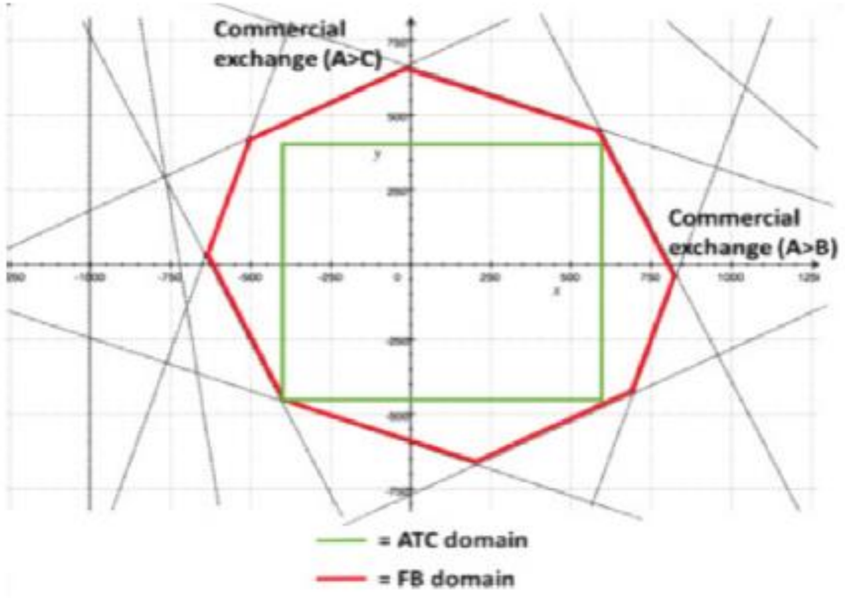
- **Power system optimization is primarily used in electricity markets,** and more recently for loss minimization and other functions

- The actual feasible region is non-convex (and very complex to identify it)

- **Electricity markets** consider the largest **convex** feasible region **and solve a MILP** (due to block offers, etc.)

- We are **missing** parts of the feasible region that can contain **"more optimal" points**

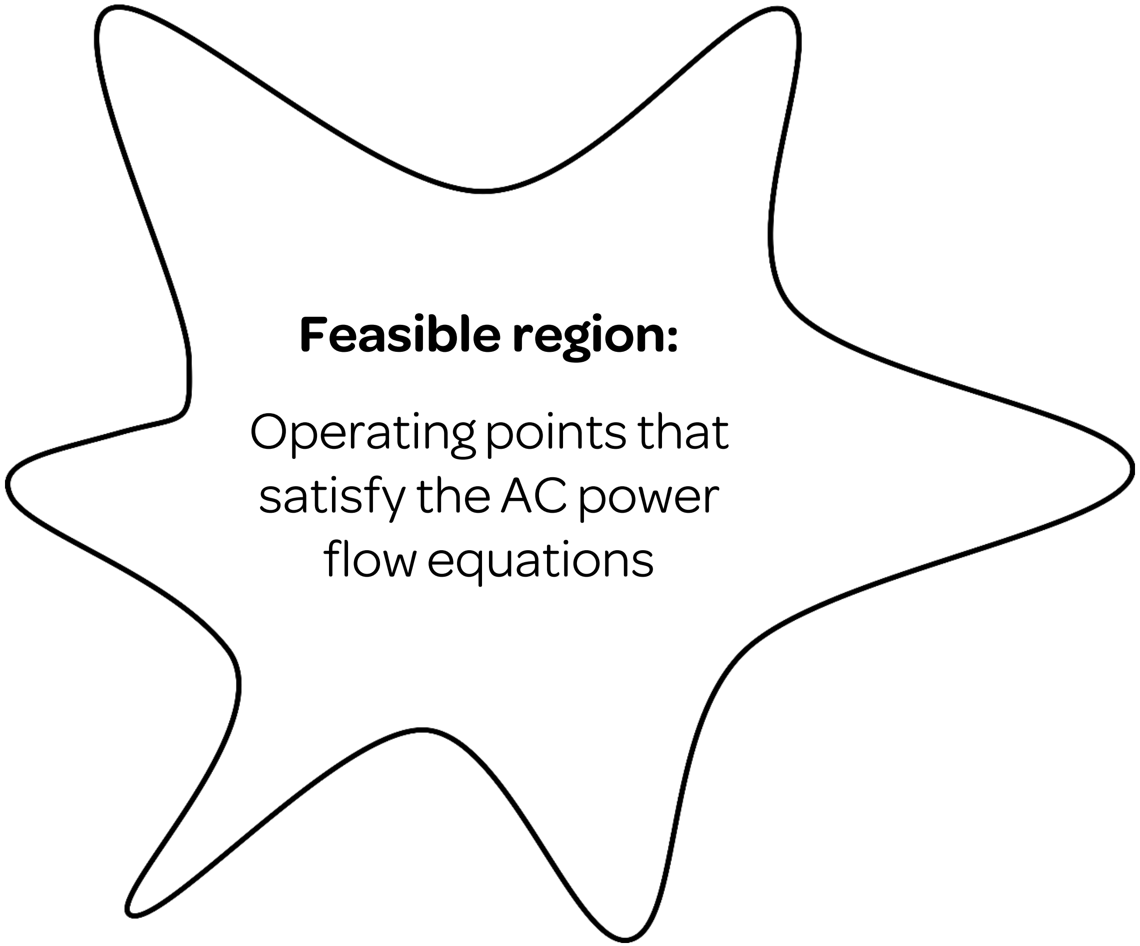
- **Goal:** find a **computationally tractable** way to consider the **actual feasible region in a MILP**



Largest Convex Region of the Feasible Space for Optimal Power Flow, for two types of Electricity Markets*

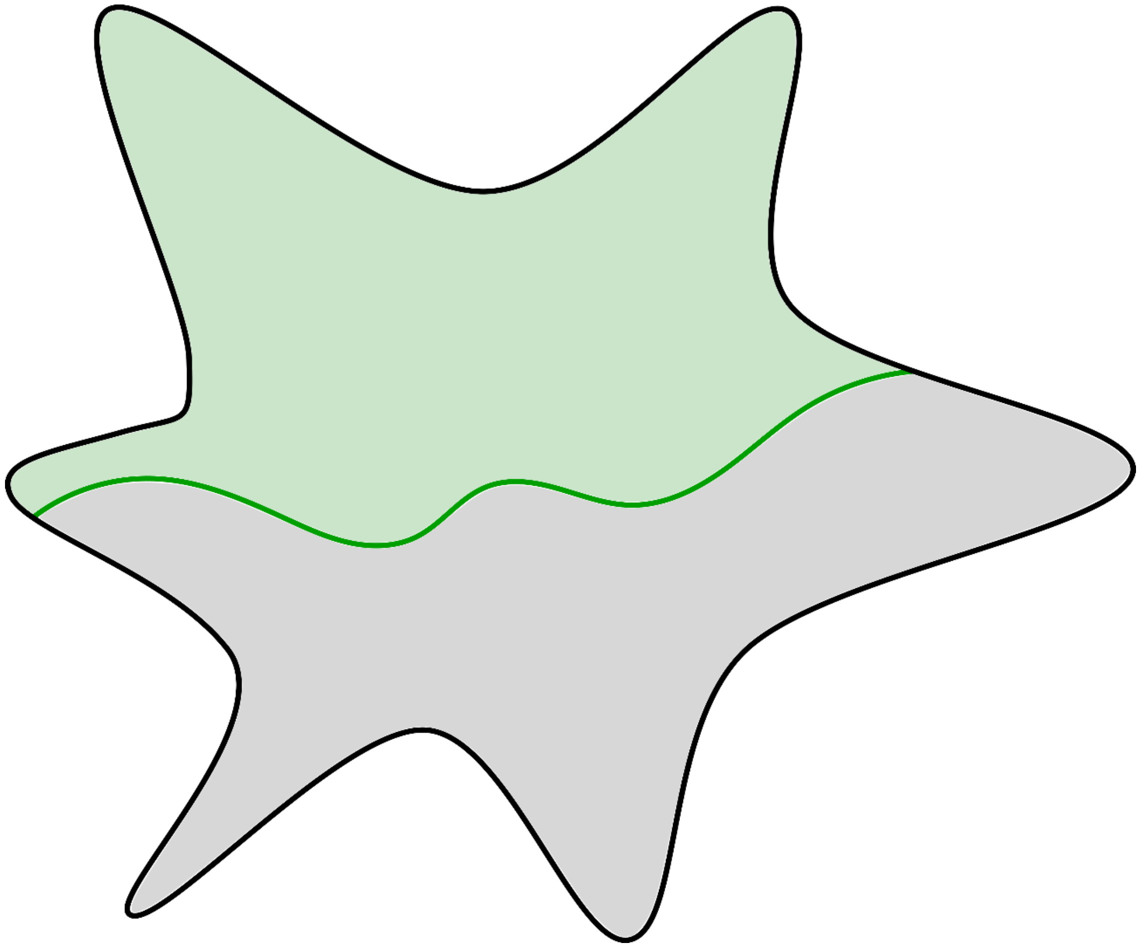
*KU Leuven Energy Institute, "EI Fact Sheet: Cross-border Electrcity Trading: Towards Flow-based Market Coupling," 2015. [Online]. Available: http://set.kuleuven.be/ei/factsheets

The safe operating region of power system operations



- Non-linear and non-convex AC power flow equations
- Component limits

The feasible space of power system operations

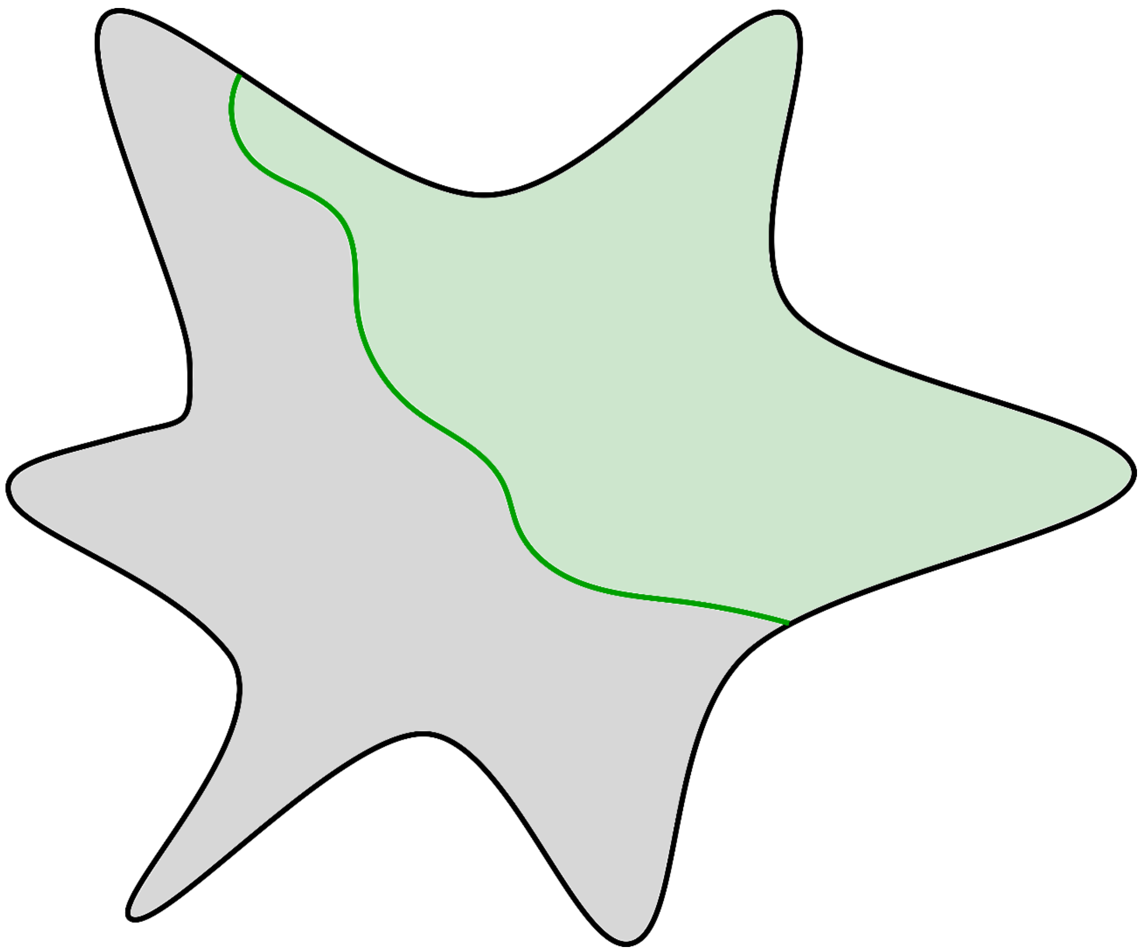


- Non-linear and non-convex AC power flow equations
- Component limits

+ N-1 security criterion

(non-linear algebraic inequality constraints)

The feasible space of power system operations

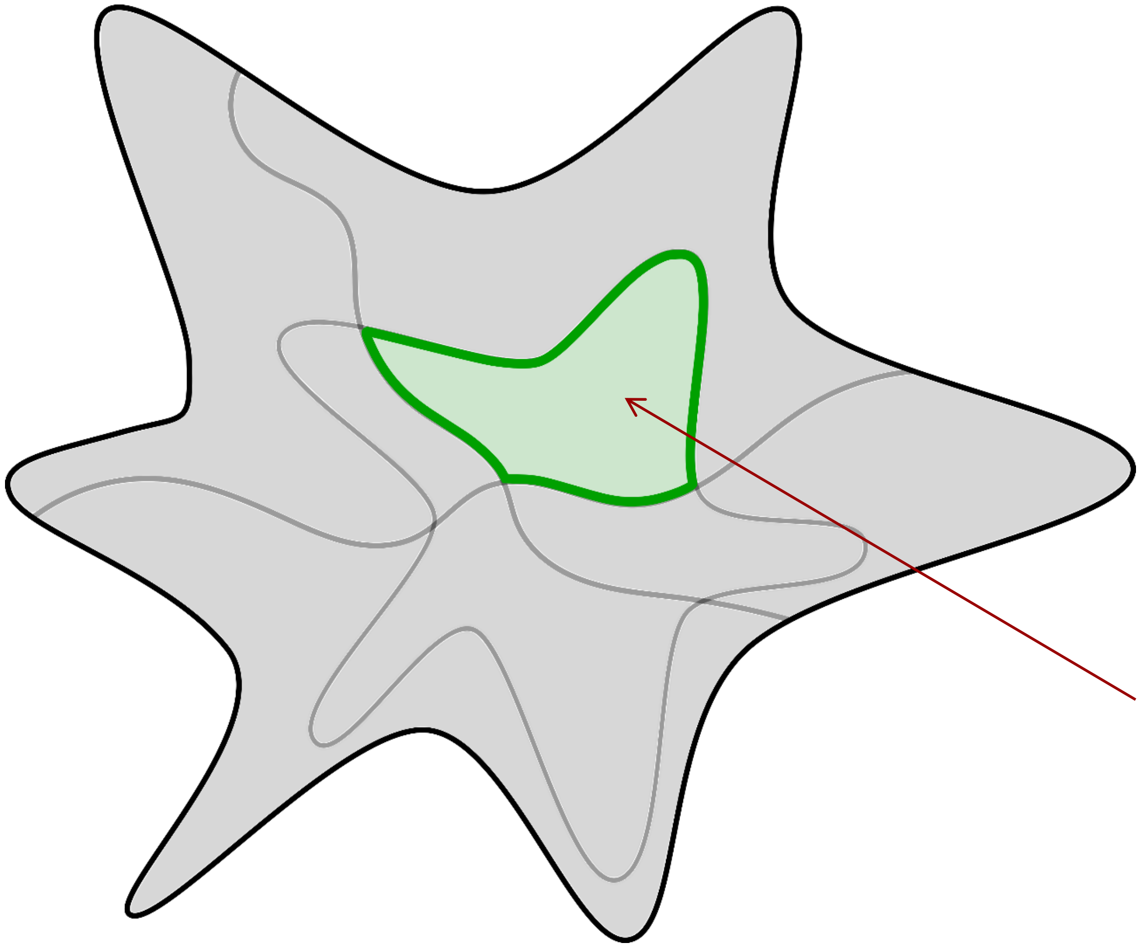


- Non-linear and non-convex AC power flow equations

- Component limits

+ Stability Limits (inequality constraints based on differential equations)

The feasible space of power system operations



- Non-linear and non-convex AC power flow equations

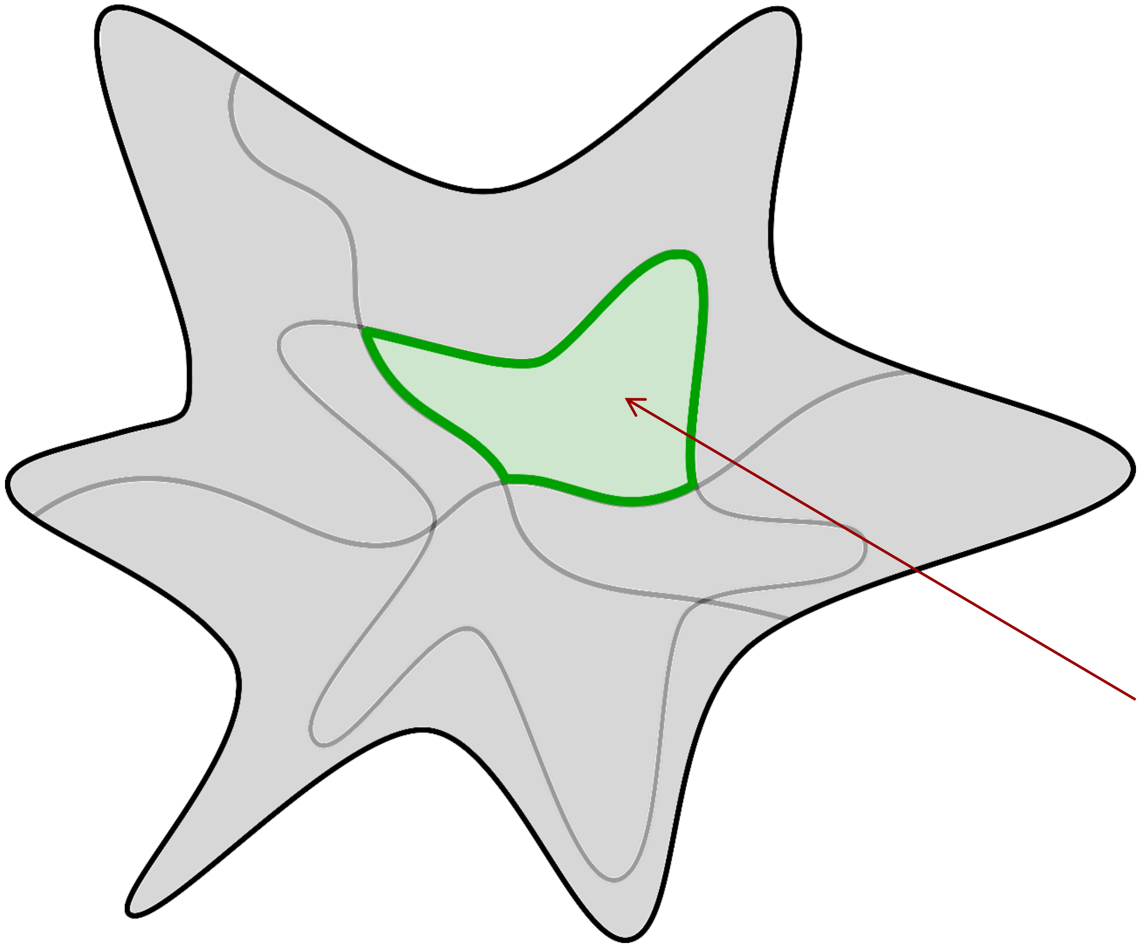
- Component limits

+ N-1 security criterion (non-linear algebraic)

+ Stability Limits (differential equations)

Intersection of all security/stability criteria: Non-linear and non-convex security region

The feasible space of power system operations



Optimization constraints should represent this area

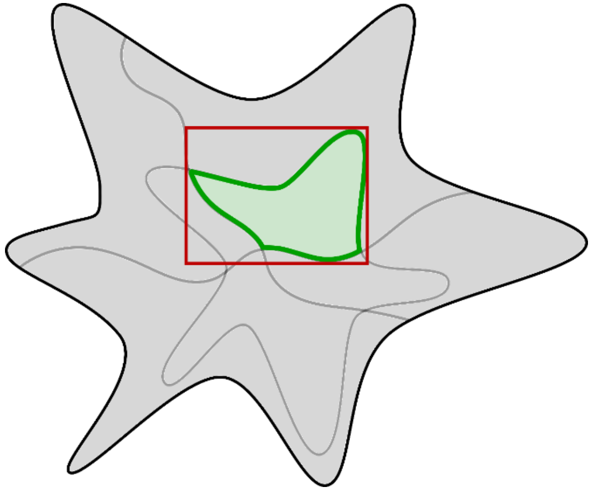
Impossible → differential and non-linear algebraic equations

Intersection of all security/stability criteria: Non-linear and non-convex security region

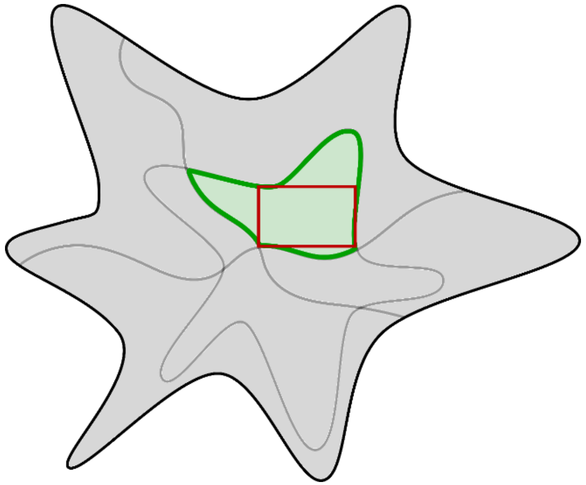
What do TSOs and market operators do?
Linear approximations

Net Transfer Capacity[1]

Inaccurate

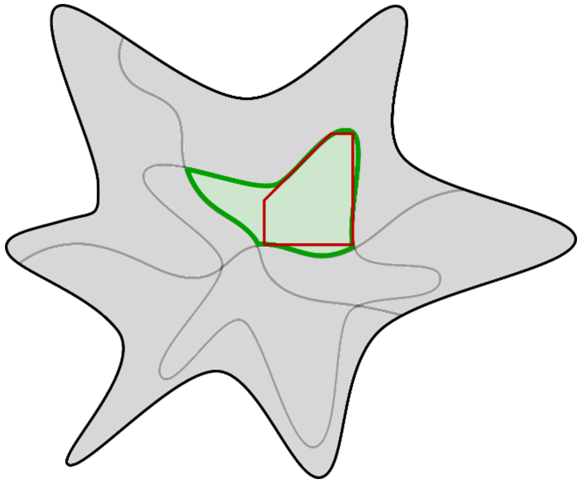


Too conservative



Flow-based market coupling[2]

Single convex region



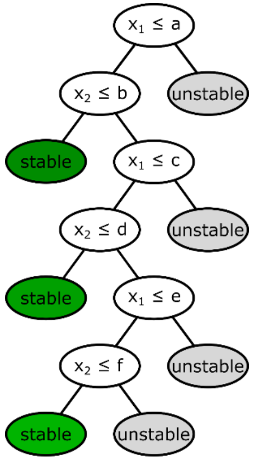
[1]e.g. Nordic Electricity Market

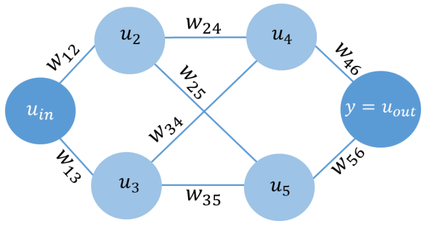
[2]e.g. Central European Market

Our proposal: Data-driven Security Constrained OPF
How does it work?

Database of secure and insecure operating points

{P,Q,V,θ,ζ}





$$\mathbf{PTDF} \cdot \left(\mathbf{P_G} - \mathbf{P_D}\right) \le \mathbf{F_{L,p}^{max}} y_p + \mathbf{F_L^{max}}(1 - y_p)$$

$$\mathbf{PTDF} \cdot \left(\mathbf{P_G} - \mathbf{P_D}\right) \ge \mathbf{F_{L,p}^{min}} y_p - \mathbf{F_L^{max}}(1 - y_p)$$

Operating points provided by the TSOs through simulated and real data

Train a decision tree or neurla network to classify secure and insecure regions

Exact reformulation to MILP

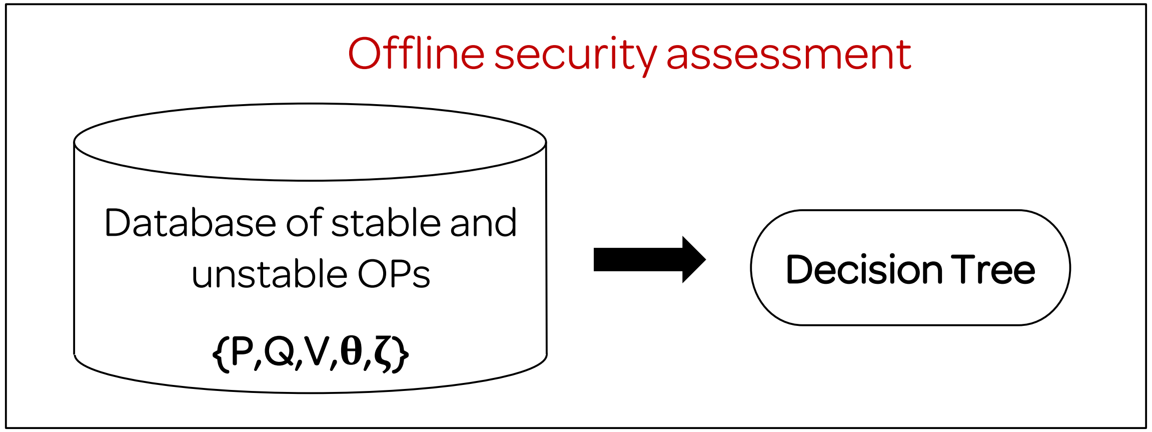
A. Venzke, D. T. Viola, J. Mermet-Guyennet, G. S. Misyris, S. Chatzivasileiadis. Neural Networks for Encoding Dynamic Security-Constrained Optimal Power Flow to Mixed-Integer Linear Programs. 2020. https://arxiv.org/pdf/2003.07939.pdf

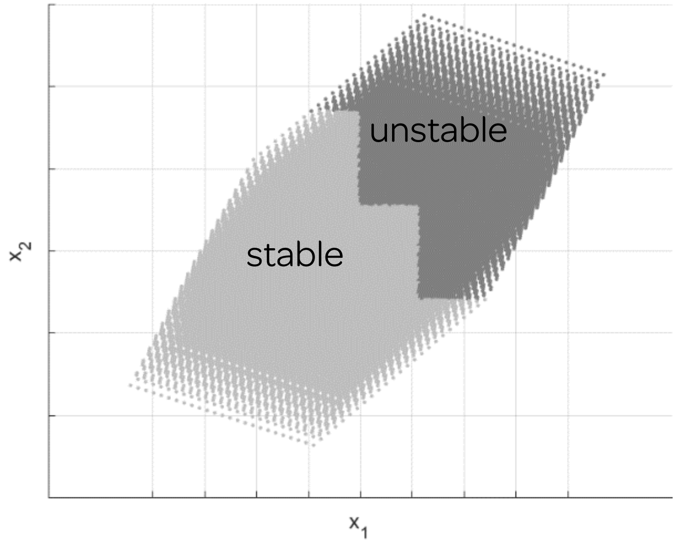
L. Halilbašić, F. Thams, A. Venzke, S. Chatzivasileiadis, and P. Pinson, "Data-driven security-constrained AC-OPF for operations and markets," *PSCC* 2018. [.pdf]

F. Thams, L. Halilbašić, P. Pinson, S. Chatzivasileiadis, and R. Eriksson, "Data-driven security-constrained OPF," *IREP* 2017. [.pdf]

From decision trees to mixed integer programming

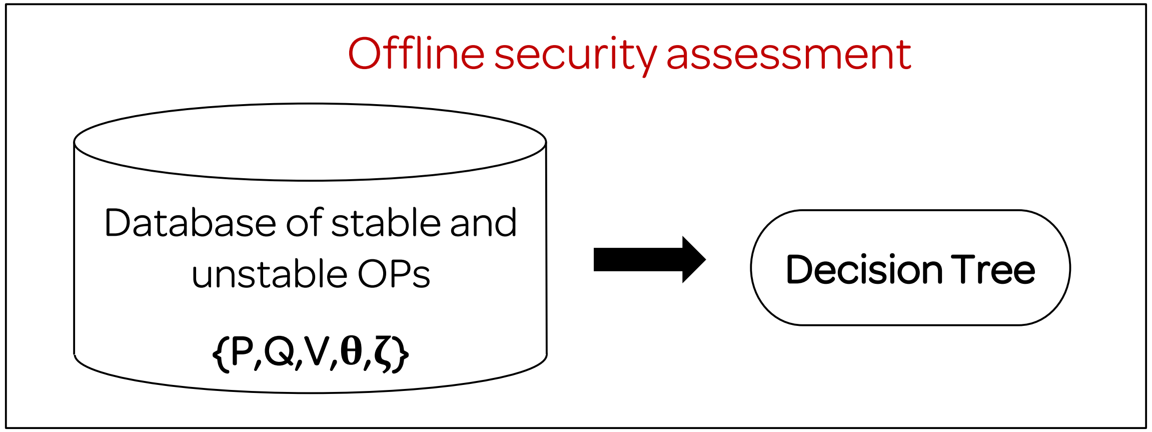
Data-driven security-constrained OPF



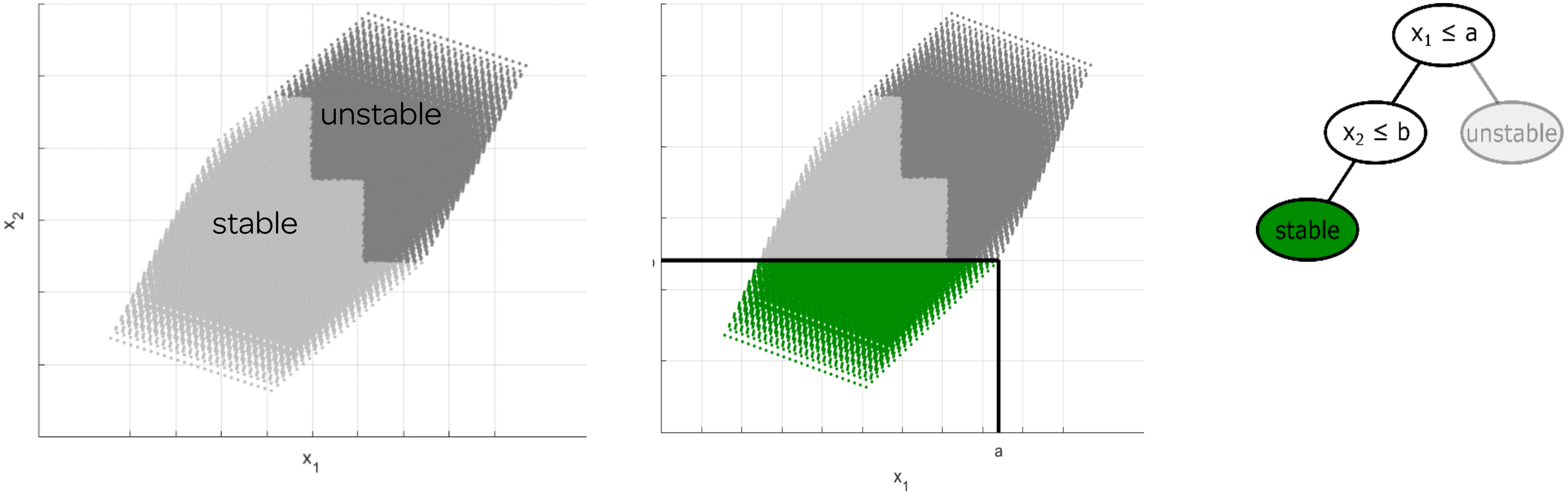


Thams et al IREP 2017, Halilbasic et al PSCC 2018

Data-driven security-constrained OPF

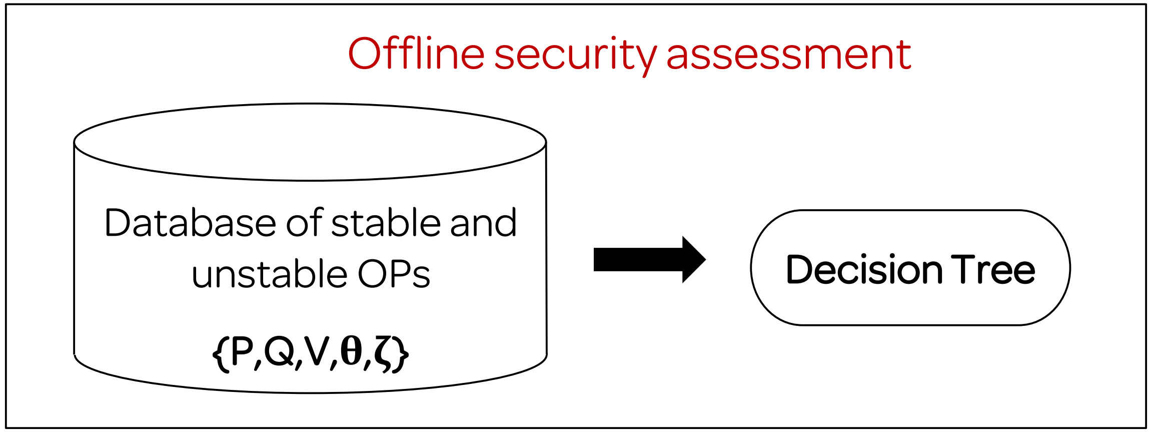


Partitioning the secure operating region

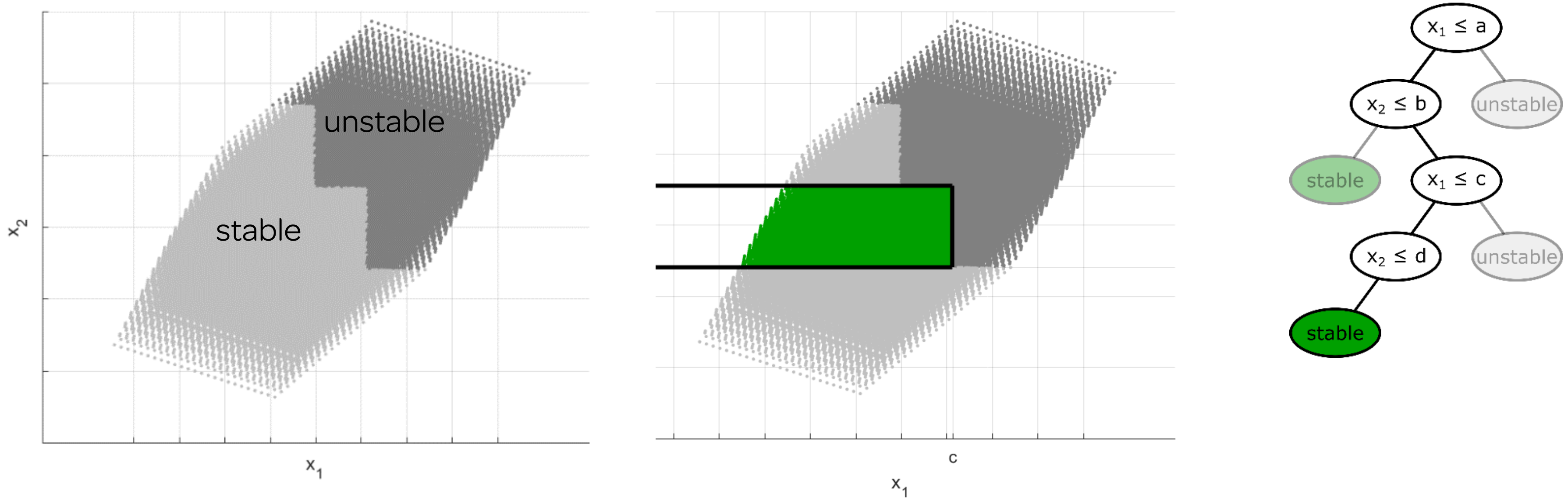


Thams et al IREP 2017,
Halilbasic et al PSCC 2018

Data-driven security-constrained OPF

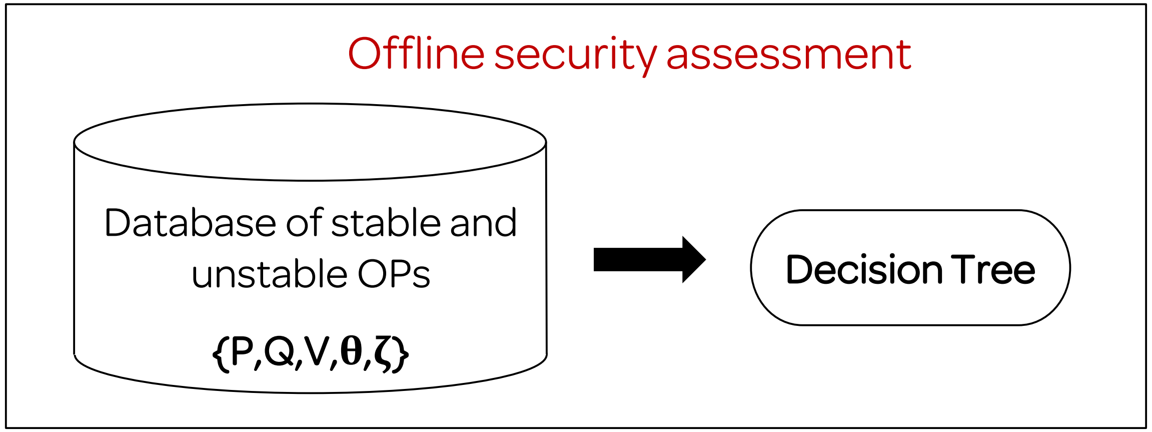


Partitioning the secure operating region

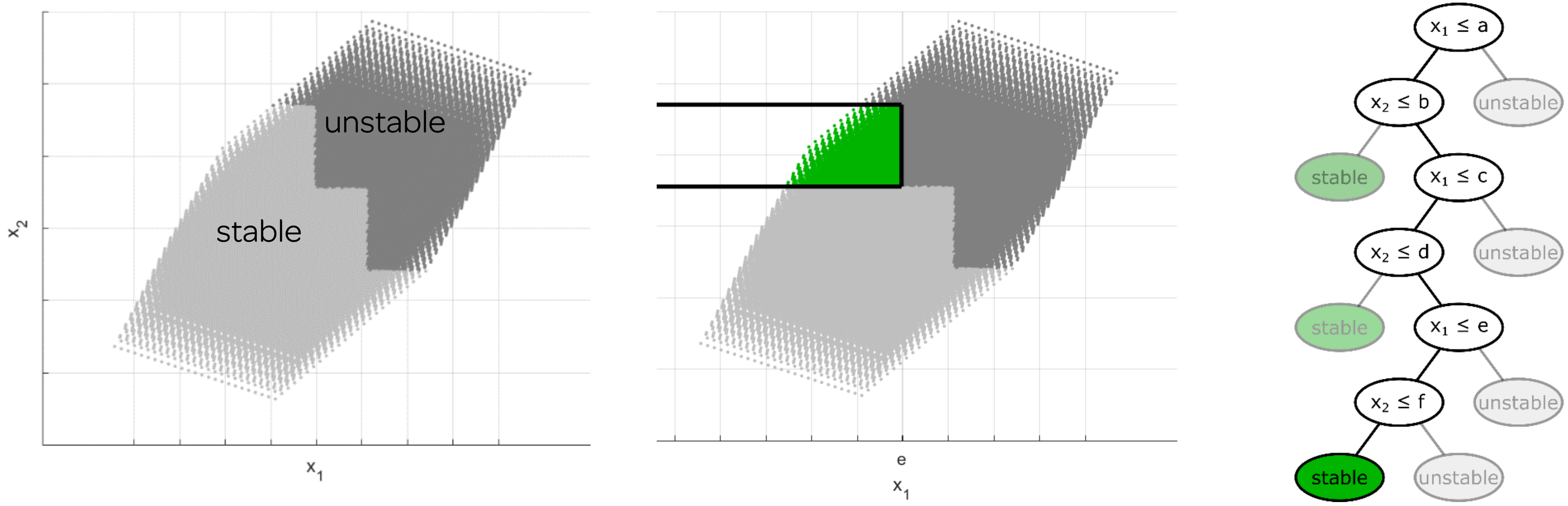


Thams et al IREP 2017,
Halilbasic et al PSCC 2018

Data-driven security-constrained OPF



Partitioning the secure operating region



Thams et al IREP 2017,
Halilbasic et al PSCC 2018

Data-driven security-constrained OPF

Offline security assessment

Database of stable and unstable OPs

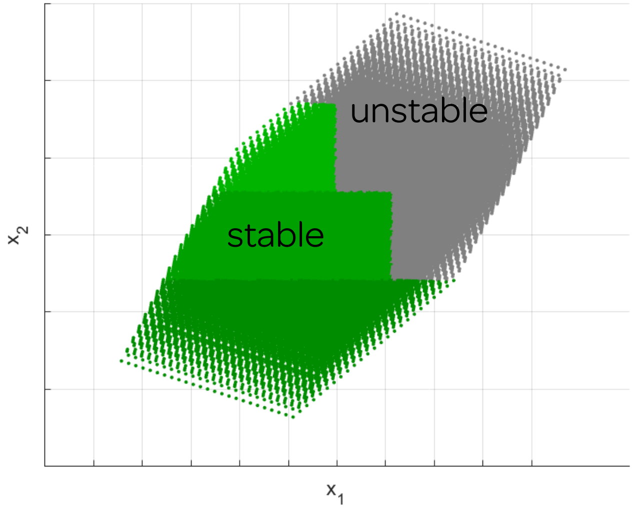
{P,Q,V,θ,ζ}

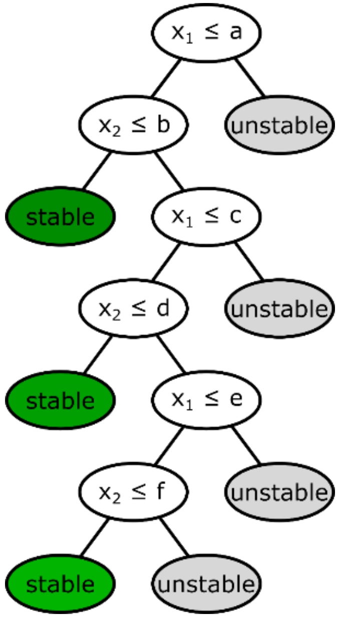
→ Decision Tree

Optimization

Integer Programming to incorporate partitions (DT)

- DC-OPF (MILP)
- AC-OPF (MINLP)
- Relaxation (MIQCP, MISOCP)

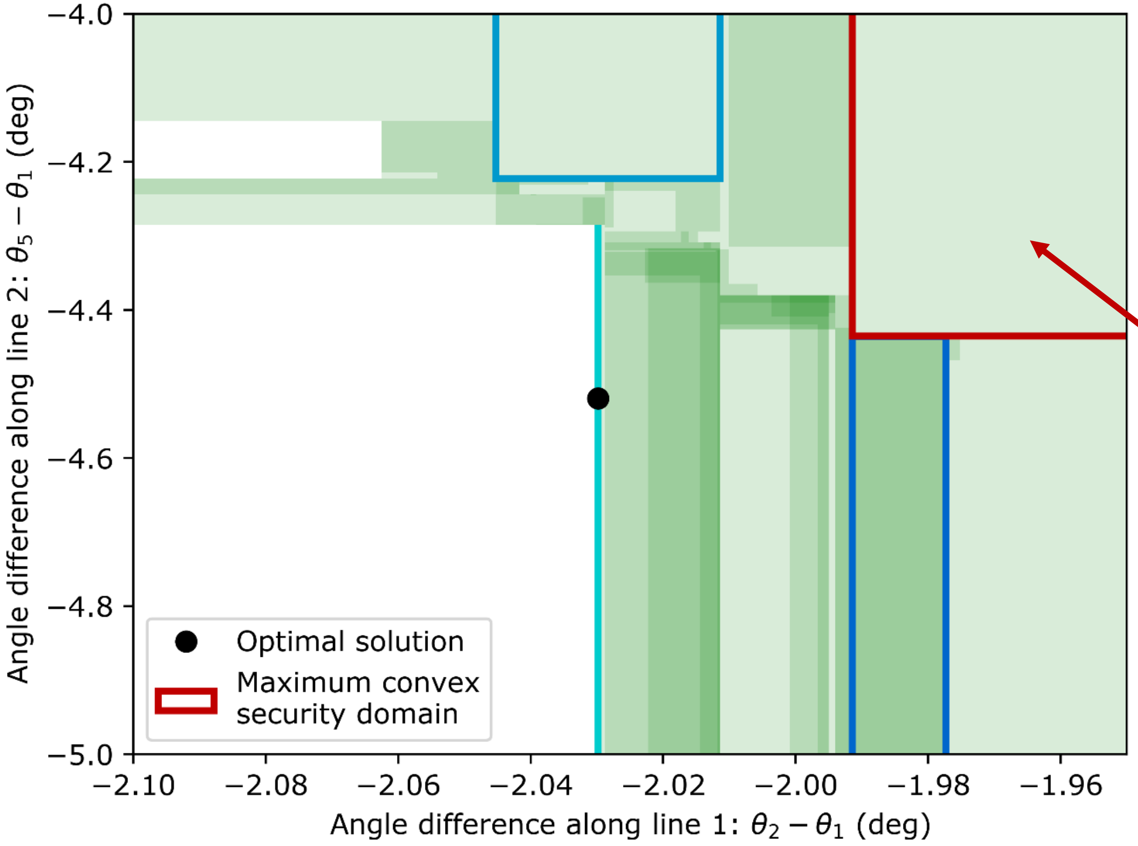




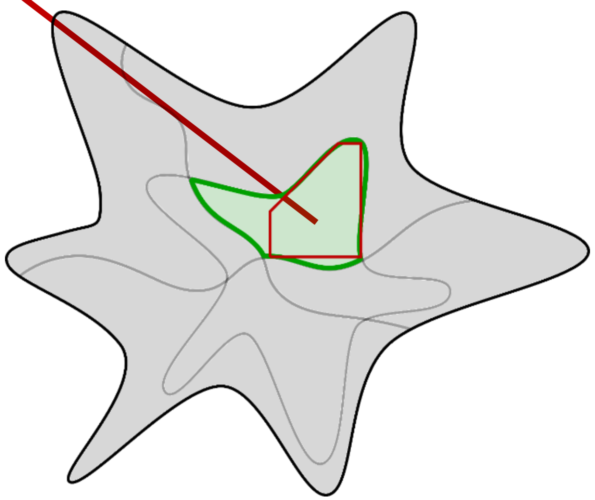
- Each leaf is a convex region
- Flow-based market coupling corresponds to the leaf that maps the largest convex region

Thams et al IREP 2017, Halilbasic et al PSCC 2018

We gain ~22% of the feasible space using data and Mixed Integer Programming

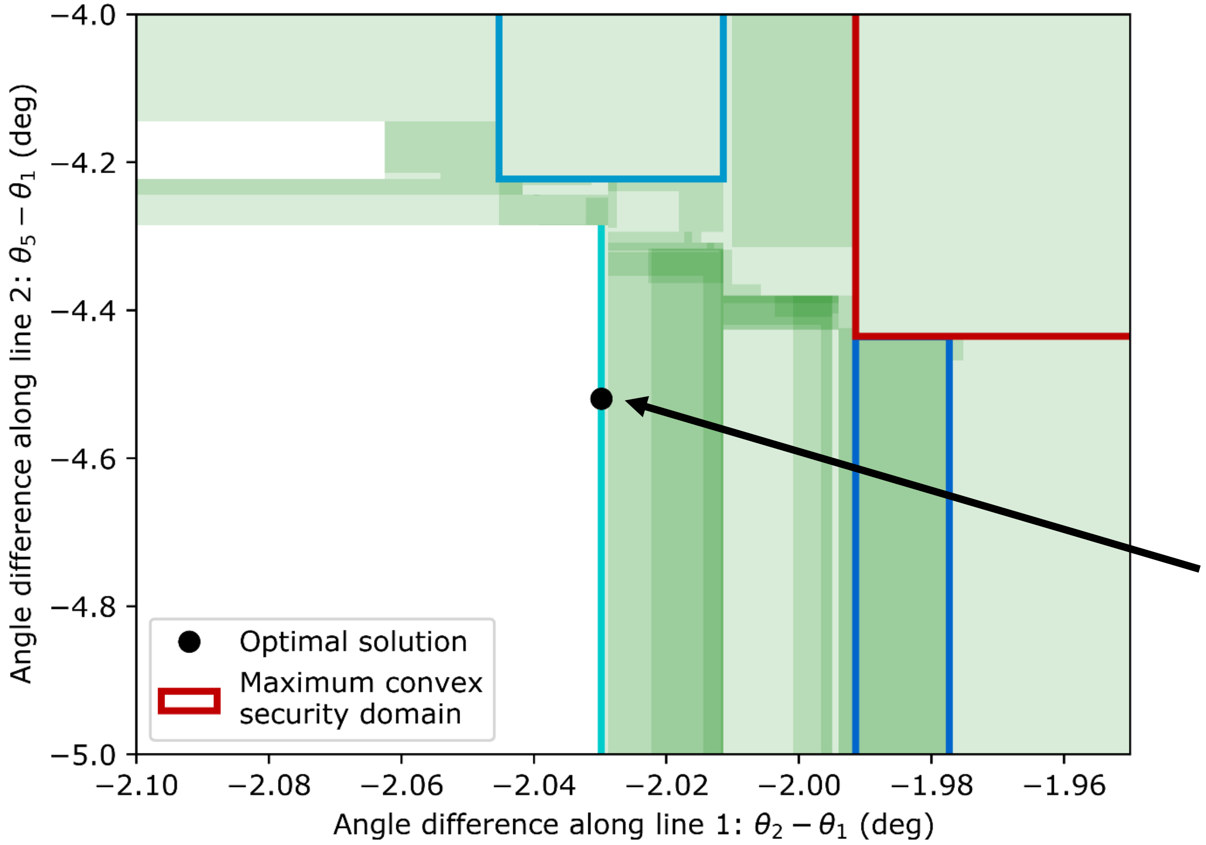


Largest convex region covers ~78%



L. Halilbašić, F. Thams, A. Venzke, S. Chatzivasileiadis, and P. Pinson, "Data-driven security-constrained AC-OPF for operations and markets," *PSCC* 2018. [.pdf]

MISOCP finds better solutions than nonconvex problem!

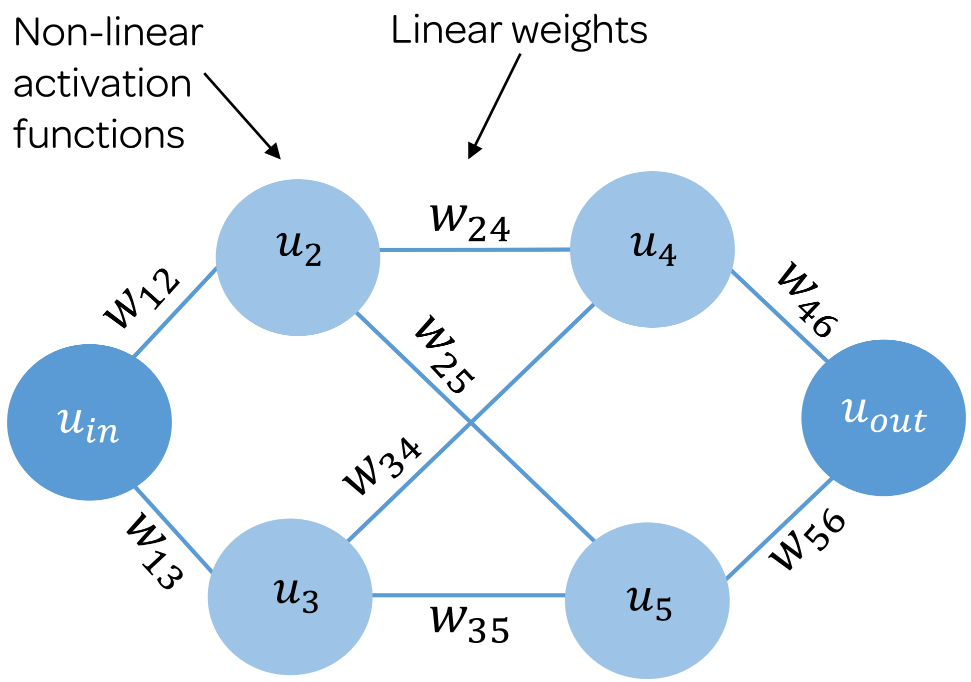


Optimum located at boundary of considered security region

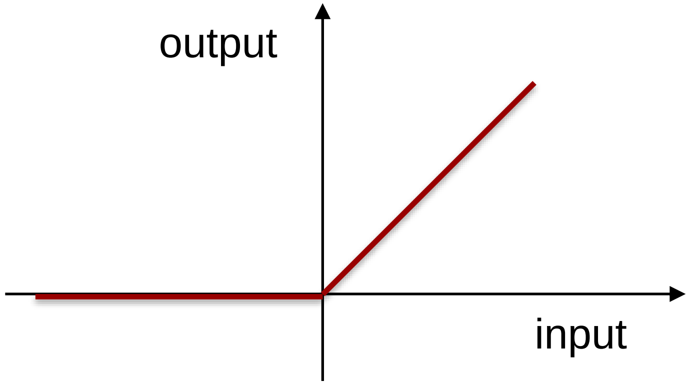
L. Halilbašić, F. Thams, A. Venzke, S. Chatzivasileiadis, and P. Pinson, "Data-driven security-constrained AC-OPF for operations and markets," *PSCC* 2018. [.pdf]

From Neural Networks to Mixed Integer Linear Programming

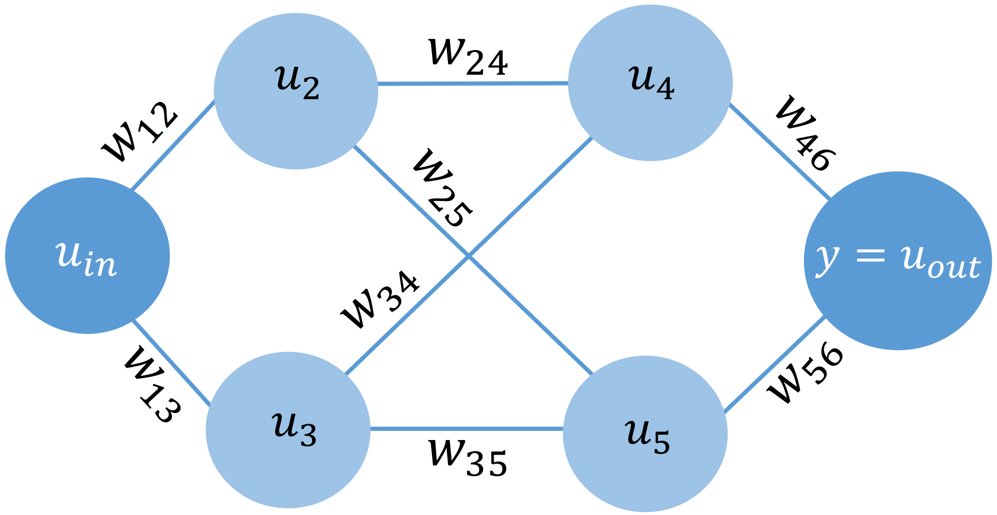
From Neural Networks to Mixed-Integer Linear Programming



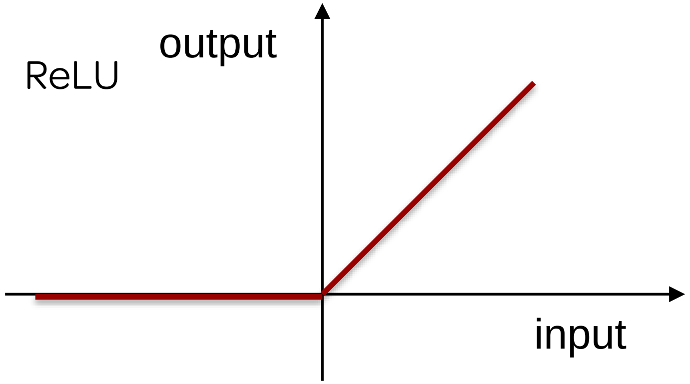
- Most usual activation function: ReLU
- ReLU: Rectifier Linear Unit



From Neural Networks to Mixed-Integer Linear Programming

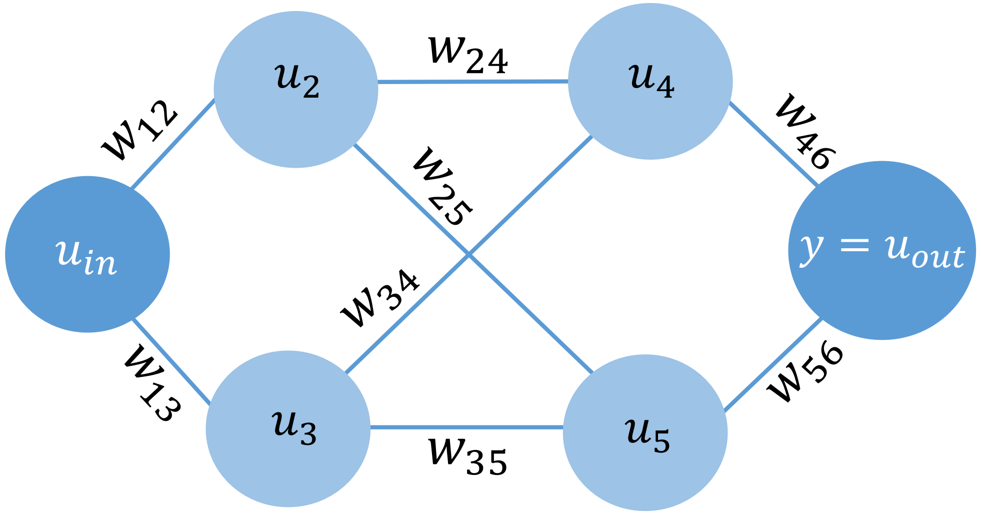


- Linear weights
- On every node: a non-linear activation function
  - ReLU: $\boldsymbol{u_j = \max(0, w_{ij}u_i + b_i)}$
- But ReLU can be transformed to a piecewise linear function with binaries



MILP

From Neural Networks to Mixed-Integer Linear Programming

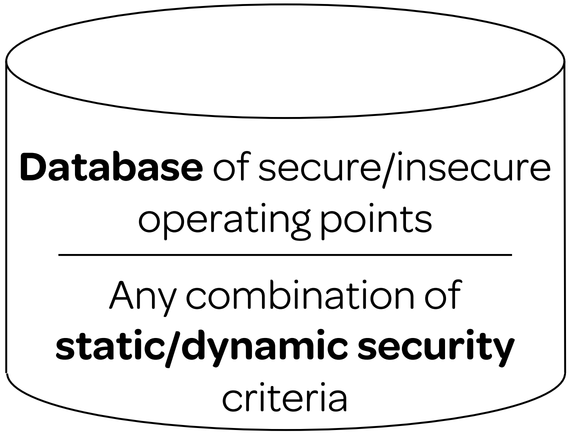


- Output
  - Binary classification: feasible/infeasible
  - ReLU is the most common activation function for Deep Neural Networks
  - Output vector y with two elements:

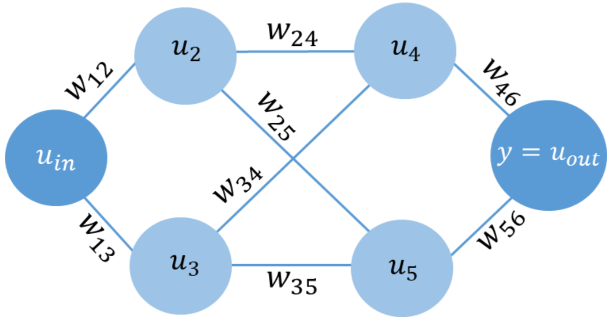
$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

- $y_1 \geq y_2$: safe
- $y_2 \geq y_1$: unsafe

Data-driven Security Constrained OPF
How does it work?



e.g. N-1 & Small-signal stability (Small-Signal Stab. up to now impossible to *directly* include in an OPF)



Train a neural network → "encode" all information about secure and insecure regions

Tractable small-signal stability-constrained OPF

$$\min f(\mathbf{p_g})$$

s.t.

$$\mathbf{p_g}^{\min} \leq \mathbf{p_g} \leq \mathbf{p_g}^{\max}$$

$$\mathbf{v_g}^{\min} \leq \mathbf{v_g} \leq \mathbf{v_g}^{\max}$$

$$\mathbf{s}_{\text{balance}}(\mathbf{p}^0, \mathbf{q}^0, \mathbf{v}^0, \boldsymbol{\theta}^0) = \mathbf{0}$$

NN → MILP

$$\hat{\mathbf{u}}_{\mathbf{k}} = \mathbf{W}_k \mathbf{u}_{\mathbf{k-1}} + \mathbf{b}_{\mathbf{k}}$$

$$\mathbf{u}_k = \max(\hat{\mathbf{u}}_k, 0) \Rightarrow \begin{cases} y_k \leq \hat{u}_k - \hat{u}_k^{\min}(1 - b_k) \\ u_k \geq \hat{u}_k \\ u_k \leq \hat{u}_k^{\max} b_k \\ u_k \geq 0 \\ b_k \in \{0,1\}^{N_k} \end{cases}$$

$$\mathbf{y} = \mathbf{u}_{out}$$

$$y_1 \geq y_2$$

Exact reformulation to MILP

A. Venzke, D. T. Viola, J. Mermet-Guyennet, G. S. Misyris, S. Chatzivasileiadis. Neural Networks for Encoding Dynamic Security-Constrained Optimal Power Flow to Mixed-Integer Linear Programs. 2020. https://arxiv.org/pdf/2003.07939.pdf

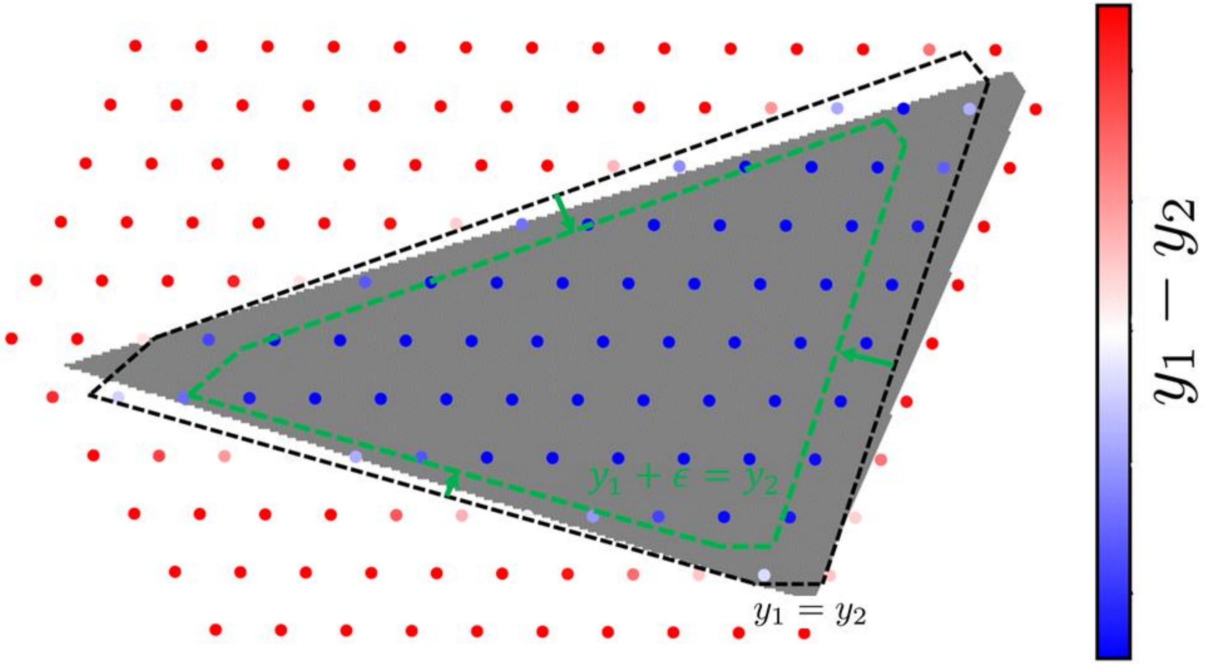
Code available: https://gitlab.com/violatimon/power_system_database_generation

Challenges

1. How do you ensure that the feasible region is captured accurately by the NN?

2. How do you handle the non-linear **equality** constraints?

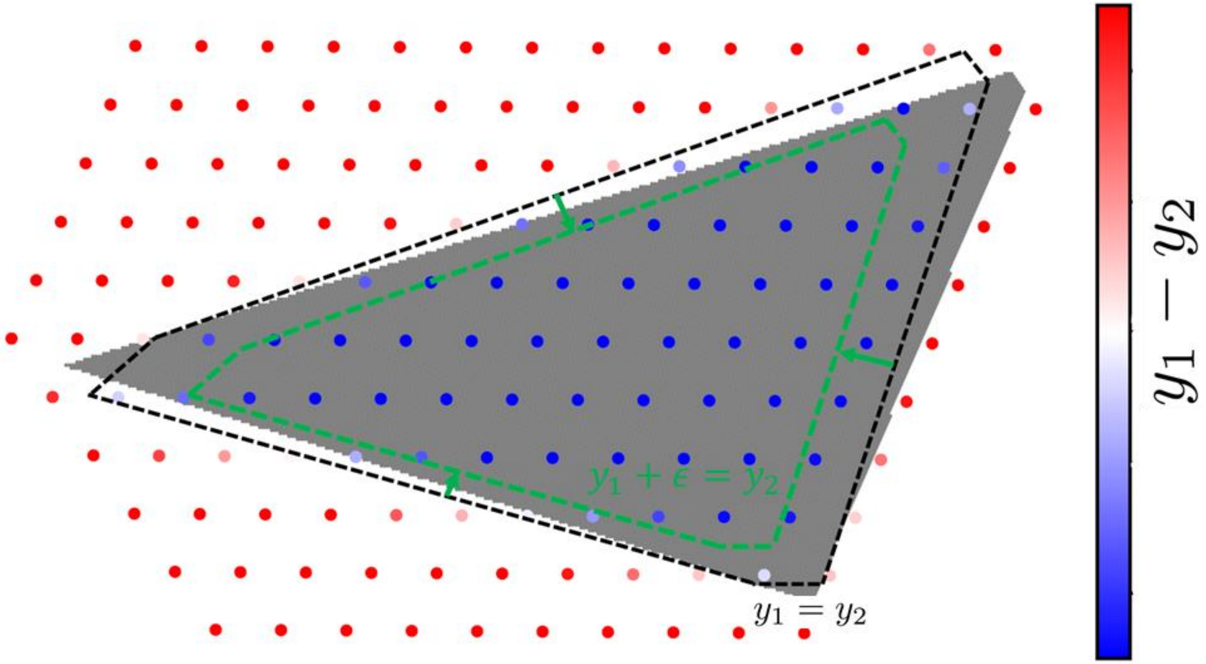
Challenge #1: Guiding NN to accurately capture the (previously intractable) feasible region



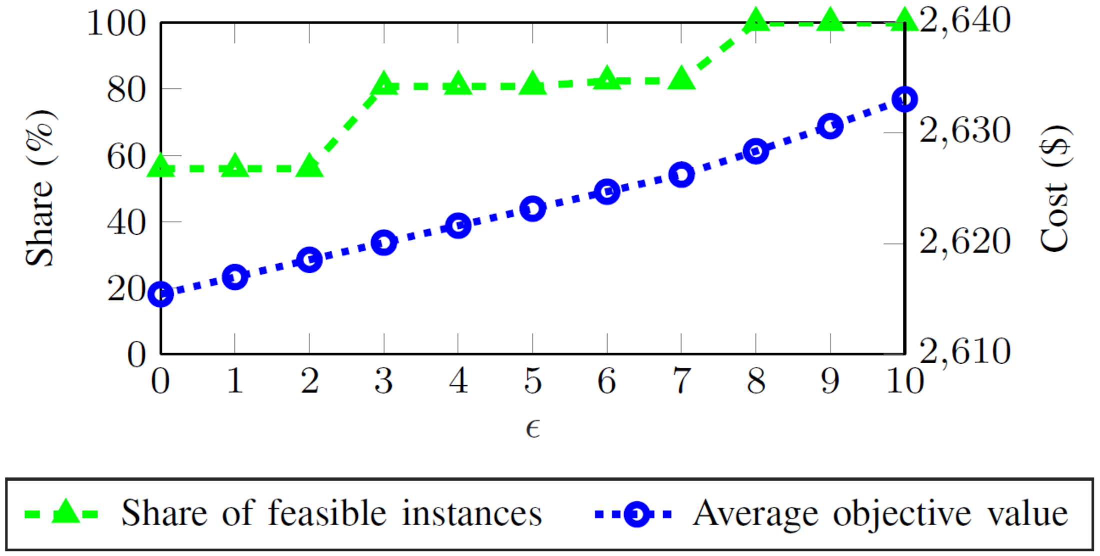
- Increase conservativeness: Replace $y_1 \geq y_2$ with $y_1 \geq y_2 + \epsilon$

Grey area: True feasible region
Black dashed line: Original NN estimate
Green dashed line: ϵ-conservativeness

Challenge #1: Guiding NN to accurately capture the (previously intractable) feasible region



- Increase conservativeness: Replace $y_1 \geq y_2$ with $y_1 \geq y_2 + \epsilon$



Grey area: True feasible region
Black dashed line: Original NN estimate
Green dashed line: ϵ-conservativeness

Challenge #2: Handling the non-linear equality constraints

- The problem: $\mathbf{s}_{\text{balance}}(\mathbf{p}^0, \mathbf{q}^0, \mathbf{v}^0, \boldsymbol{\theta}^0) = \mathbf{0}$ ← quadratic constraints (#constraints = #nodes)

- Three solution options to avoid solving a MINLP:
  a. Train a **Regression Neural Network** to estimate q^0, θ^0 from p^0, v^0 and insert it as a list of mixed-integer linear constraints to the problem
  b. **Convexify**, if possible, and solve a MISOCP; recover feasible (global?) optimal
  c. **Linearize** the non-linear equations and solve iteratively the MILP

Challenge #2: Handling the non-linear equality constraints

- The problem: $\mathbf{s}_{\text{balance}}(\mathbf{p}^0, \mathbf{q}^0, \mathbf{v}^0, \boldsymbol{\theta}^0) = \mathbf{0}$ ← quadratic constraints (#constraints = #nodes)

- Three solution options to avoid solving a MINLP:
  a. Train a **Regression Neural Network** to estimate q^0, θ^0 from p^0, v^0 and insert it as a list of mixed-integer linear constraints to the problem
  b. **Convexify**, if possible, and solve a MISOCP; recover feasible (global?) optimal
  c. **Linearize** the non-linear equations and solve iteratively the MILP

- **Here: linearization**
  – **Replace N constraints with 1 linearized constraint** of the total active power nodal balance
  – **Iterative MILP converges very fast : 1.04 iterations on average in 125 instances**

$$\sum_{\mathcal{G}} \mathbf{p}_{\mathbf{g}}^{\mathbf{0}} + \sum_{\mathcal{N}} \mathbf{p}_{\mathbf{d}}^{\mathbf{0}} + p_{\text{losses}}|_i + \frac{\delta p_{\text{losses}}}{\delta \mathbf{p}_{\mathbf{g}}^{\mathbf{0}}}|_i(\mathbf{p}_{\mathbf{g}}^{\mathbf{0}} - \mathbf{p}_{\mathbf{g}}^{\mathbf{0}}|_i) + \frac{\delta p_{\text{losses}}}{\delta \mathbf{v}_{\mathbf{g}}^{\mathbf{0}}}|_i(\mathbf{v}_{\mathbf{g}}^{\mathbf{0}} - \mathbf{v}_{\mathbf{g}}^{\mathbf{0}}|_i) = 0$$

Results: average over 125 instances

	Problem formulation	Problem type	Generation cost ($)	Solver time (s)	Share of feasible instances (%)
Conventional non-linear program (interior-point)	AC-OPF	NLP	2425.94 (0.0%)	0.04	35.2
	+ N-1 security	NLP	2565.13 (5.7%)	0.15	35.2
Neural Networks → MILP	+ small-signal stability ($\epsilon = 0$)	MILP	2615.43 (7.8%)	0.22	56.0
	+ small-signal stability ($\epsilon = 8$)	MILP	2628.37 (8.3%)	0.12	100.0

Results: average over 125 instances

	Problem formulation	Problem type	Generation cost ($)	Solver time (s)	Share of feasible instances (%)
Conventional non-linear program (interior-point)	AC-OPF	NLP	2425.94 (0.0%)	0.04	35.2
	+ N-1 security	NLP	2565.13 (5.7%)	0.15	35.2
Neural Networks → MILP	+ small-signal stability ($\epsilon = 0$)	MILP	2615.43 (7.8%)	0.22	56.0
	+ small-signal stability ($\epsilon = 8$)	MILP	2628.37 (8.3%)	0.12	100.0

MILP **solution time similar** to NLP; **and MILP contains constraints that are intractable for the NLP**

*ACOPF is faster because it contains less than 15% of the constraints of the other problems

Results: average over 125 instances

	Problem formulation	Problem type	Generation cost ($)	Solver time (s)	Share of feasible instances (%)
Conventional non-linear program (interior-point)	AC-OPF	NLP	2425.94 (0.0%)	0.04	35.2
	+ N-1 security	NLP	2565.13 (5.7%)	0.15	35.2
Neural Networks → MILP	+ small-signal stability ($\epsilon = 0$)	MILP	2615.43 (7.8%)	0.22	56.0
	+ small-signal stability ($\epsilon = 8$)	MILP	2628.37 (8.3%)	0.12	100.0

Conservativeness ($\epsilon = 8$) helps obtain feasible solutions without increasing substantially the objective function

Wrap-up

- From decision trees and neural networks to Mixed Integer Linear Programs (MILP)
  - **Exact** transformation
  - **Capture efficiently constraints that were previously intractable** (e.g. based on differential equations)
  - Note: MILPs are also used for neural network verification*

- Challenges
  - Handling of non-linear equality constraints: SOCP or linearization appear effective
  - Accurately capturing the feasible space: **introducing ϵ-conservativeness**

- The framework can apply to any general non-linear program with intractable constraints

*A. Venzke, S. Chatzivasileiadis. Verification of Neural Network Behaviour: Formal Guarantees for Power System Applications. Under Review. 2019. https://arxiv.org/pdf/1910.01624.pdf

Thank you!

Spyros Chatzivasileiadis

Associate Professor, PhD

www.chatziva.com

spchatz@elektro.dtu.dk

Neural Networks to MILP:

A. Venzke, D. T. Viola, J. Mermet-Guyennet, G. S. Misyris, S. Chatzivasileiadis. Neural Networks for Encoding Dynamic Security-Constrained Optimal Power Flow to Mixed-Integer Linear Programs. 2020. https://arxiv.org/pdf/2003.07939.pdf

Decision Trees to MILP:

L. Halilbašić, F. Thams, A. Venzke, S. Chatzivasileiadis, and P. Pinson, "Data-driven security-constrained AC-OPF for operations and markets," *PSCC* 2018. [.pdf]

F. Thams, L. Halilbašić, P. Pinson, S. Chatzivasileiadis, and R. Eriksson, "Data-driven security-constrained OPF," in 10th IREP Symposium – Bulk Power Systems Dynamics and Control, 2017. [.pdf]

Neural Network Verification:

A. Venzke, S. Chatzivasileiadis. Verification of Neural Network Behaviour: Formal Guarantees for Power System Applications. Under Review. 2019. https://arxiv.org/pdf/1910.01624.pdf

Some code available at:

www.chatziva.com/downloads.html