1st Adelaide Power Systems Summer School (APSSS), Australia

Machine learning for power systems

Spyros Chatzivasileiadis
Associate Professor

Energy in DTU

DTU Photonics

Solar PV systems

DTU Wind Energy

Thermal systems

DTU Mechanical Engineering

DTU Electrical Engineering

DTU Energy Conversion

DTU Compute

Dynamic systems and smart cities

DTU Civil Engineering

Building energy, Solar thermal and District heating

DTU Management Engineering

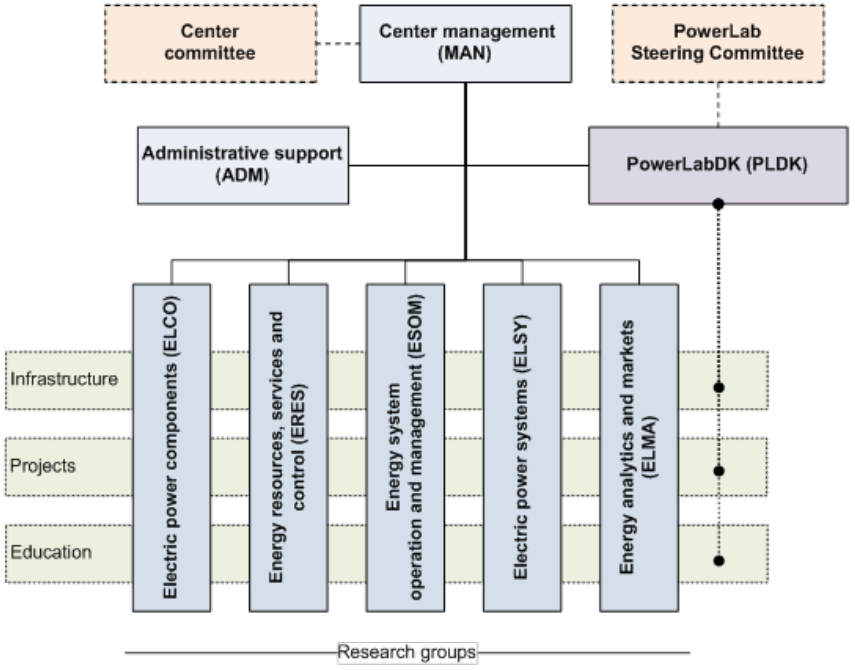
Energy system analysis and policy

Center for Electric Power and Energy (CEE)

- **Established 15 August 2012** by merging two existing units (Lynbgy + Risø)
  - Among the strongest university centers in Europe with approx. 110 employees and 12 faculty members
- **Bachelor and Master programs:** Sustainable Energy Design, Electrical Engineering, Wind Energy, Sustainable Energy
- **Direct support from:**

ENERGINET Ørsted

SIEMENS Danfoss



DTU consistently ranks among the top 10 universities of the world in Energy Science and Engineering (Shanghai ranking, 2016, 2017, 2018)

Strong National and International Collaboration

Selected collaboration partners

Academic partners:

ETH Zürich (CH)
EPFL ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE (CH)
NANYANG TECHNOLOGICAL UNIVERSITY (SG)
(US)
(DK)
(CN)
University of Strathclyde Glasgow (UK)
Berkeley UNIVERSITY OF CALIFORNIA (US)
UNIVERSITY OF DELAWARE (US)
THE HONG KONG POLYTECHNIC UNIVERSITY 香港理工大學 (HK)

NORDIC FIVE TECH
DTU (DK)
NTNU (NO) Norwegian University of Science and Technology
Aalto University School of Science and Technology (FI)
CHALMERS (SE)
KTH (SE)

>80 formal partners

Commercial and industrial partners:

ENERGINET (DK)
Ørsted (DK)
Radius (DK)
SIEMENS (DE+DK)
elia Powering a world in progress (BE)
ABB (SE/DK)
BORNHOLMS ENERGI & FORSYNING (DK)
nationalgrid (UK)
IBM (CH)
Vestas (DK)
Pacific Northwest NATIONAL LABORATORY (US)
Danfoss (DK)
KØBENHAVNS KOMMUNE
seas-nve (DK)
SCOTTISHPOWER (UK)
DANSK ENERGI (DK)
NKT (DK)

>100 formal partners

Smart Energy Networks (DK) Research, Development and Demonstration
MISSION INNOVATION Accelerating the Clean Energy Revolution (Global)
EERA European Energy Research Alliance (EU)
iea Energy Technology Network (Global)
ISGAN International smart grid action network
IEEE PES Power & Energy Society™ (Global)
EES-UETP (EU) Electric Energy Systems University Enterprise Training Partnership

Our (my) research topics – 11 researchers – 8 nationalities

Zero-inertia systems

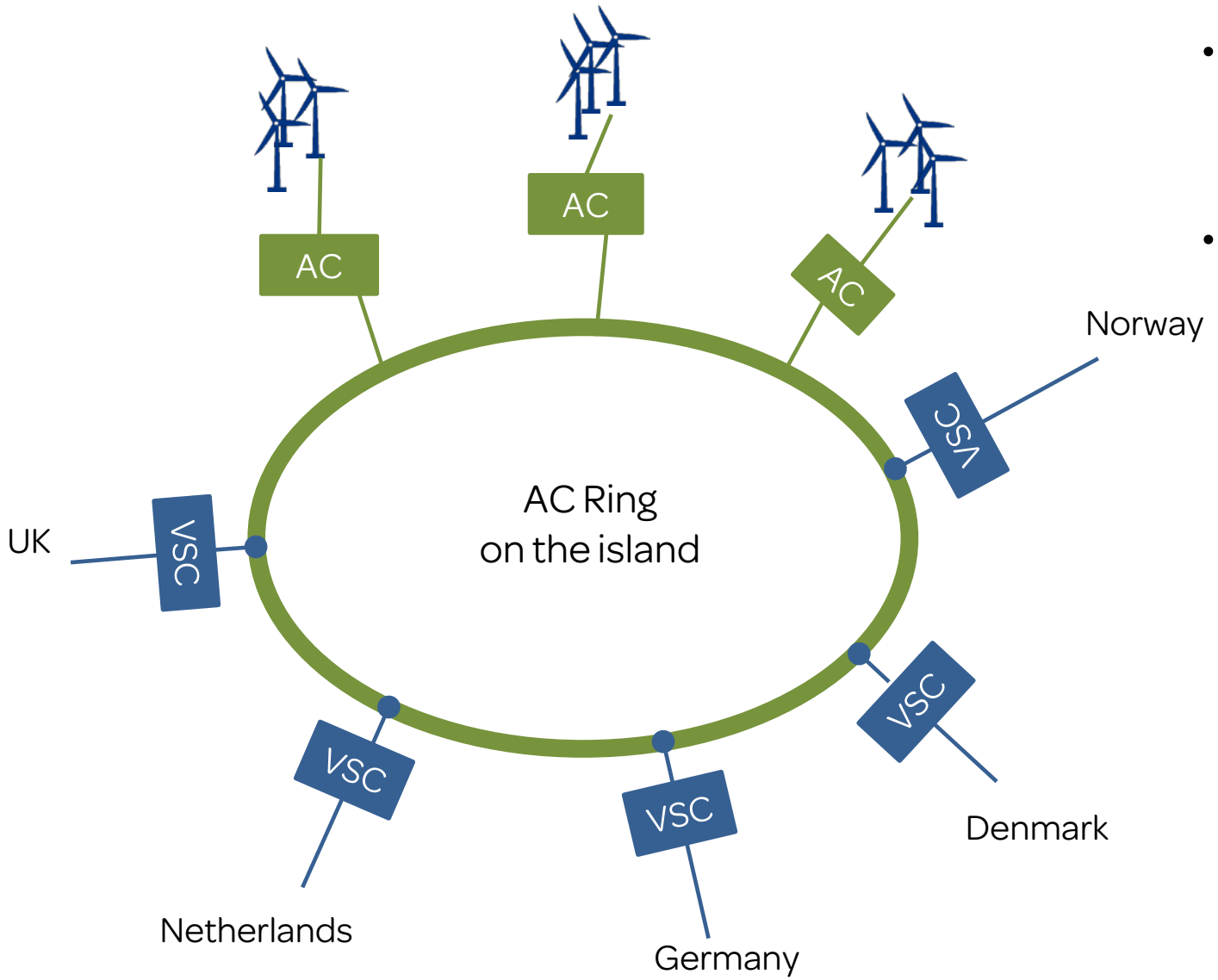
North Sea Wind Power Hub

Market integration of HVDC

Data-driven control and markets in distribution grids

Machine learning for power systems

North Sea Wind Power Hub



- Will probably be the **first true zero-inertia AC system in the world**

- A range of challenges:
  - How do you **ensure N-1** between grid-forming converters?
  - What kind of **controls** are necessary to maintain stability against much faster transients?
  - Need for **new simulation tools** (RMS-based tools, e.g. Powerfactory, insufficient to capture stability)

More on www.multi-dc.eu !

Our (my) research topics – 11 researchers – 8 nationalities

Zero-inertia systems

North Sea Wind Power Hub

Market integration of HVDC

Data-driven control and markets in distribution grids

Machine learning for power systems

Machine learning: Why shall we apply it in power systems?

Machine learning: Why shall we apply it in power systems? (1/3)

1. ML methods can handle well extremely complex systems
   - Sometimes impossible or very difficult to develop models based on first principles, e.g. weather forecasting, image processing, etc.

2. ML methods can infer from incomplete data
   - Given a limited set of training data, ML methods are known to generalize well, e.g. written text recognition

3. ML methods can be extremely fast
   - Milliseconds instead of seconds, minutes, or hours: Require a limited set of linear and simple non-linear computations instead of e.g. computationally intensive numerical integration methods

Machine learning: Why shall we apply it in power systems? (2/3)

Machine Learning	Power Systems
1. ML methods can handle well extremely complex systems	1. Real-life power systems are described by thousands of variables, parameters, and differential-algebraic equations
2. ML methods can infer from incomplete data	2. It is computationally impossible (intractable) to check for all possible operating conditions
3. ML methods can be extremely fast	3. Modern power systems have high variability and uncertainty in generation and demand: Need to assess and decide fast

Machine learning: Why shall we apply it in power systems? (3/3)

BUT:

1. Power Systems are safety-critical systems
   - An operator would never trust a neural network if it cannot interpret or anticipate its behavior

2. Researchers and Engineers have spent the last 100 years developing high-quality models to describe power systems
   - Why neglect all this and use black-box techniques only based on data?

Takeaway #1

Solid motivation is key: If you wish to apply machine learning (including deep learning) methods on any problem, develop **solid arguments** why this is the only or the best way to do it

Machine learning: Why shall we apply it in power systems? (3/3)

BUT:

1. Power Systems are safety-critical systems
   - An operator would never trust a neural network if it cannot interpret or anticipate its behavior

→ **Neural network verification:** neural networks are no longer a black-box

2. Researchers and Engineers have spent the last 100 years developing high-quality models to describe power systems
   - Why neglect all this and use black-box techniques only based on data?

→ **Physics-informed neural networks:** exploit the underlying physical models

Goals of this lecture

1. Basic principles of neural network training
2. Unifying framework for power system security assessment and optimization
   - Capture constraints impossible to capture before through ML and exact transformations
3. The importance of high quality data: sampling beyond statistics
4. Neural network verification
5. Physics-informed neural networks

Let’s talk about power systems!

Why are they safety-critical?

Why are they complex?

Spot the difference!

North East Blackout 2003

- North East Blackout 2003
- Affected 55 million people in the US and Canada
- Estimated cost of 7-10 billion USD
- Took 2 days (!) for full restoration!

Blackouts are rare but costly!

- Frequency of power interruptions
  - 1 hour per year
- Economic damage from power interruptions
  - about 80 billion USD/year (US only, 2005)
- Total electric energy cost in the US:
  - 370 billion USD/year

India Blackout 2012 → affected 700 million people (!)

Affected regions in red

Operators run every day a security assessment

Energinet Control Room, Denmark

- Security = ability to withstand disturbances
- Security Assessment:
  - Screen contingency list every 15 mins
  - Prepare contingency plans for critical scenarios
- Run both:
  - Steady-state, i.e. power flows to check N-1 and violation of limits
  - Dynamic simulations

Challenges

- Dynamic simulations are hard
  - System of differential-algebraic equations with 10k degrees of freedom

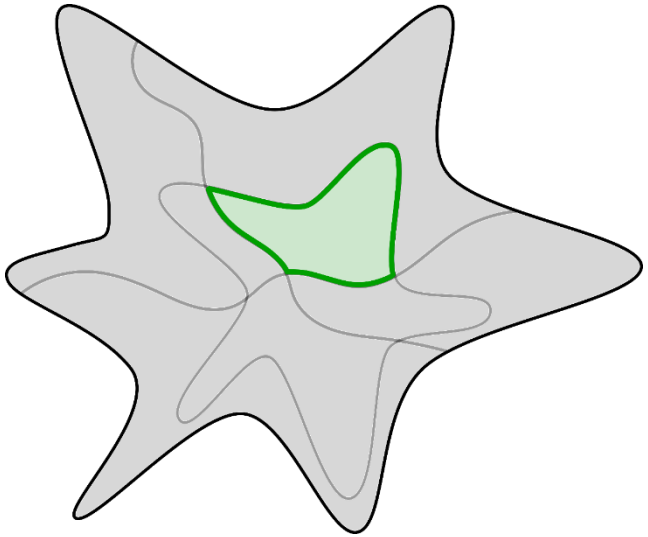
- Checking for N-k contingencies is a hard combinatorial problem
  - Usually computationally impossible to check even for all N-2 in a realistic systems with thousands of buses

- The safe operating region is a non-linear non-convex region
  - Impossible to use analytical tricks to determine it

So.... what do we do?

So… what do we do?

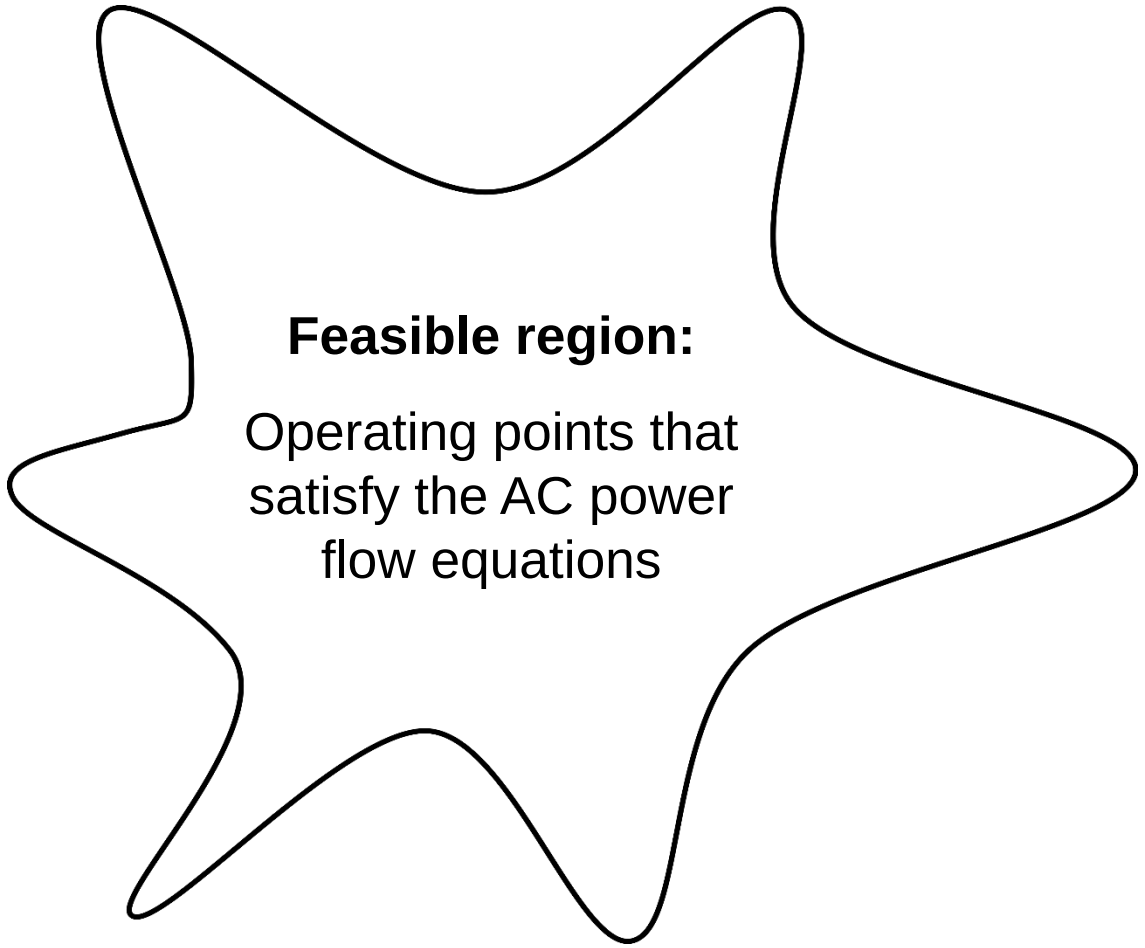
Identifying the power system security region



- Run a lot of simulations assessing each operating point
  - Several approaches for efficient approximations to boost computation speed
- Stability certificates
  - Extract **sufficient conditions** for sub-areas of the security region
- Machine learning approaches
  - Train for a given dataset and **infer** for all new points

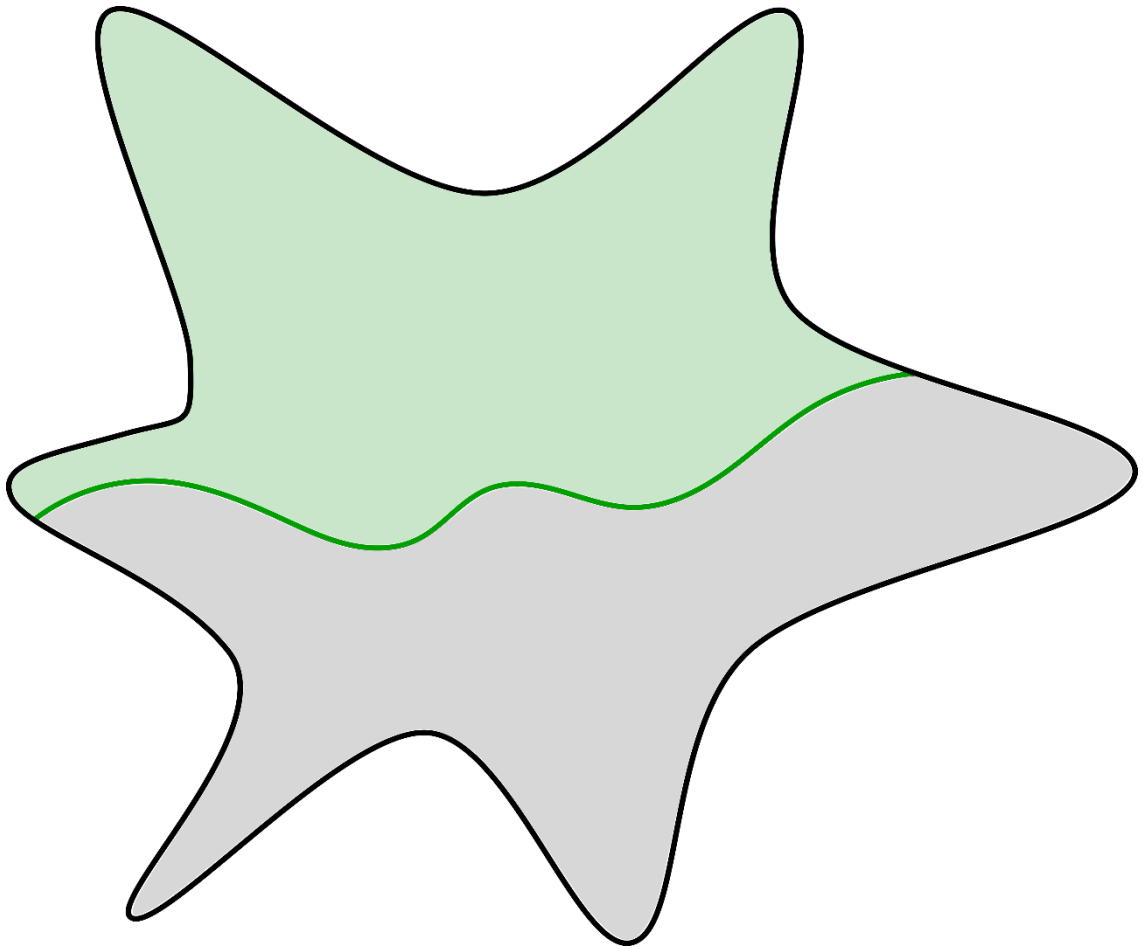
Power System Security Assessment

The safe operating region of power system operations



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

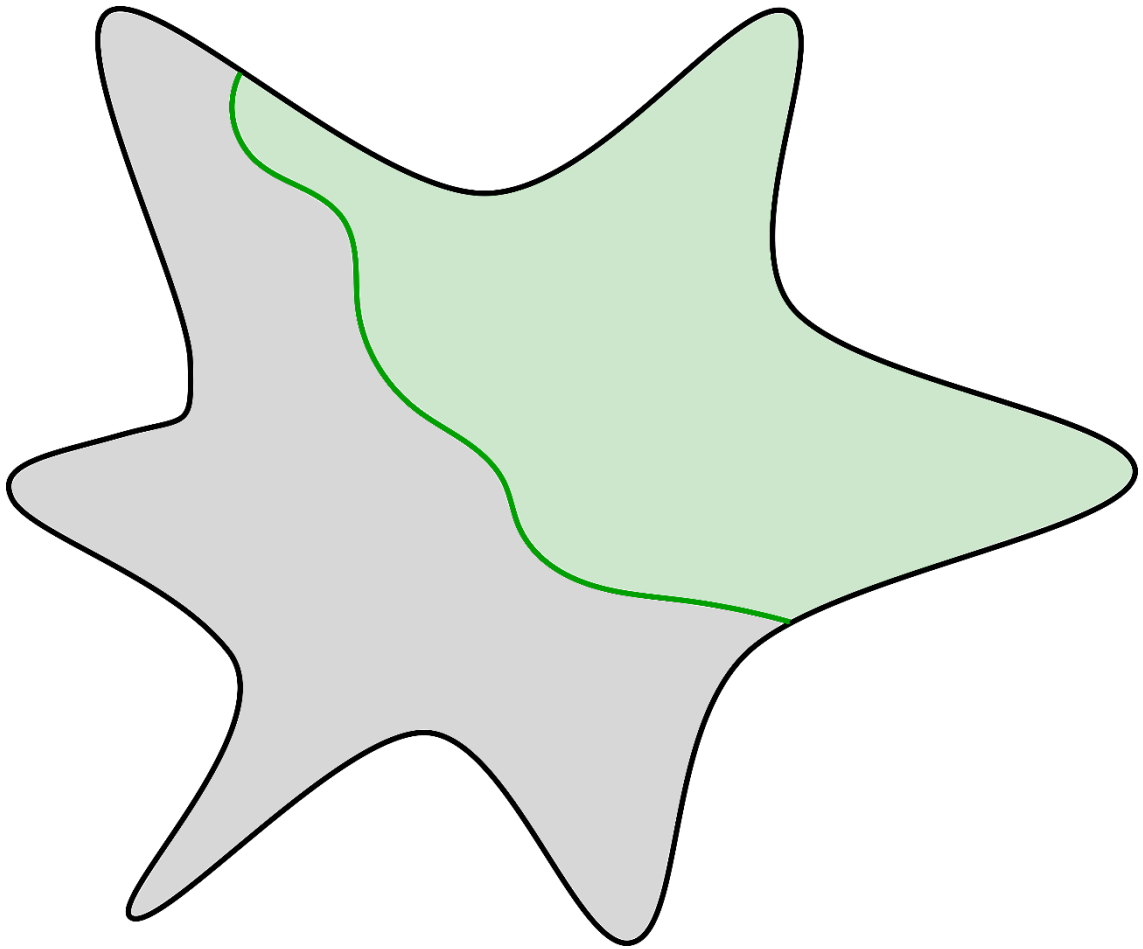
The feasible space of power system operations



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

+ N-1 security criterion

The feasible space of power system operations

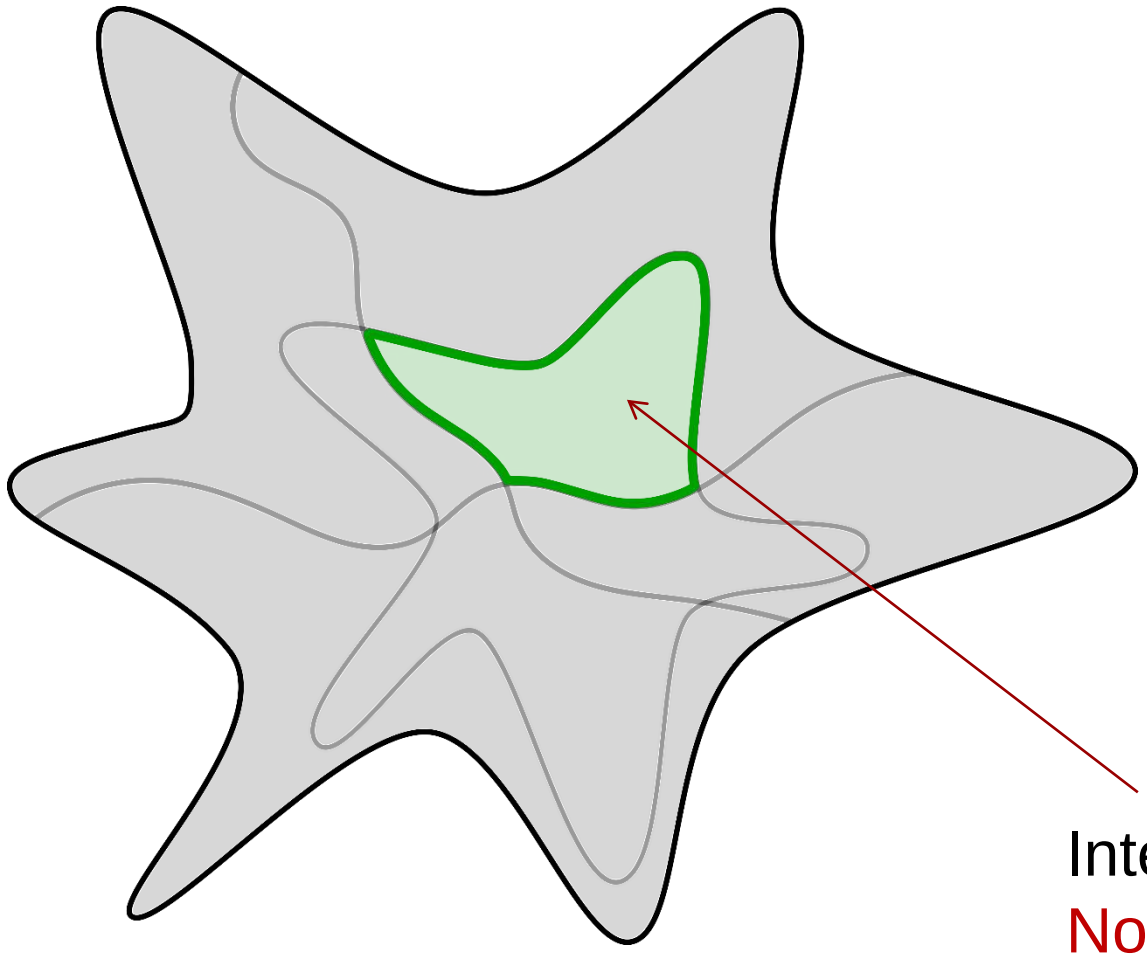


- Non-linear and non-convex AC power flow equations
- Component limits

+ N-1 security criterion

+ Stability Limits

The feasible space of power system operations



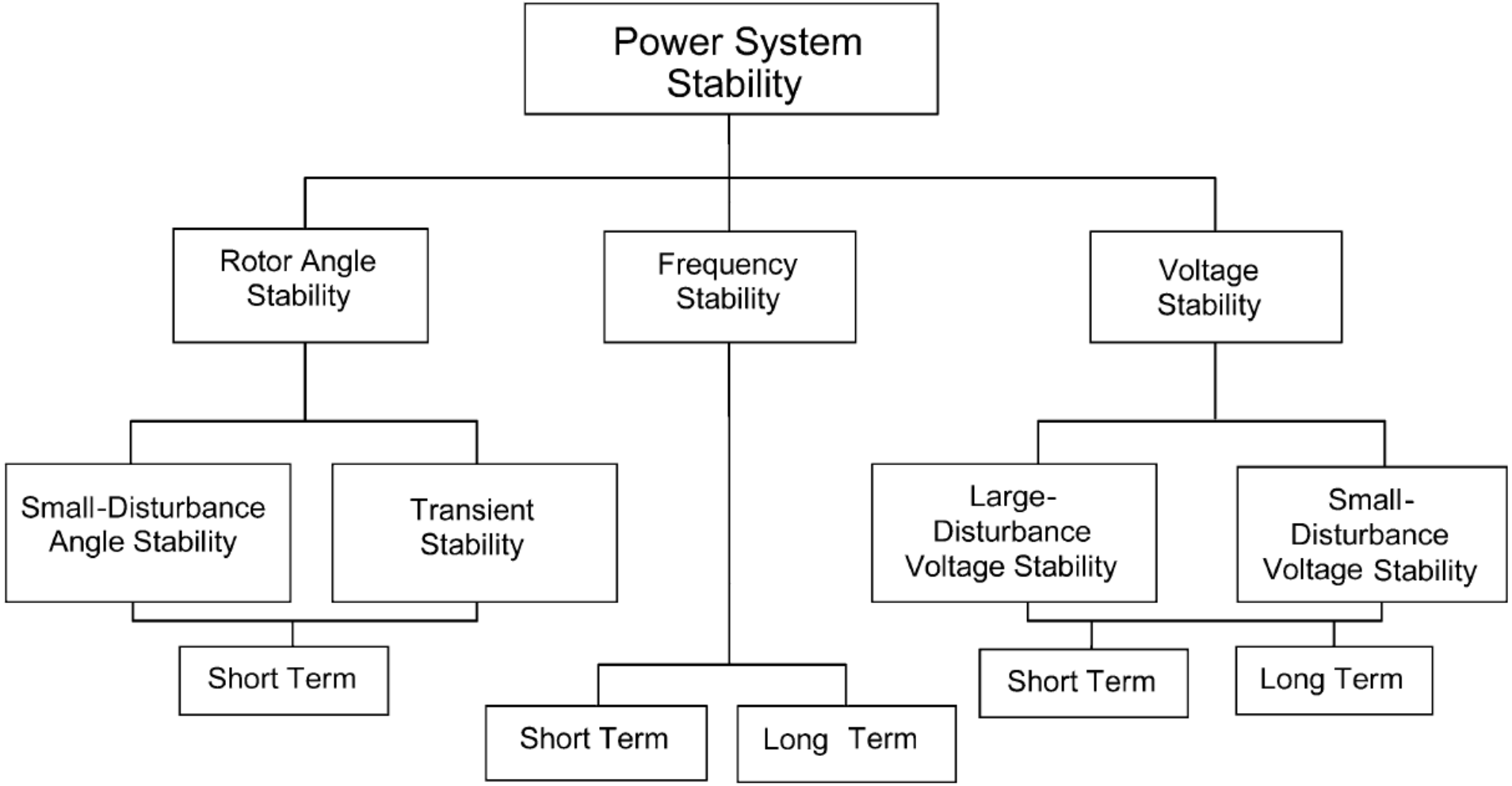
- Non-linear and non-convex AC power flow equations
- Component limits

+ N-1 security criterion

+ Stability Limits

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

Operators have to check for all instability types



AND steady-state security!

N-1 contingency analysis through power flows

Kundur et al., IEEE TPWRS, 2004

Guiding example for this lecture: Machine Learning for Power System Security Assessment

Possible Machine Learning Tools

- Decision Trees
  - First proposed by Louis Wehenkel (Univ of Liege) in the '90s
  - Very successful
  - Applications in the industry
  - Research is still ongoing; latest focus is on interpretability

- Neural networks (several papers)

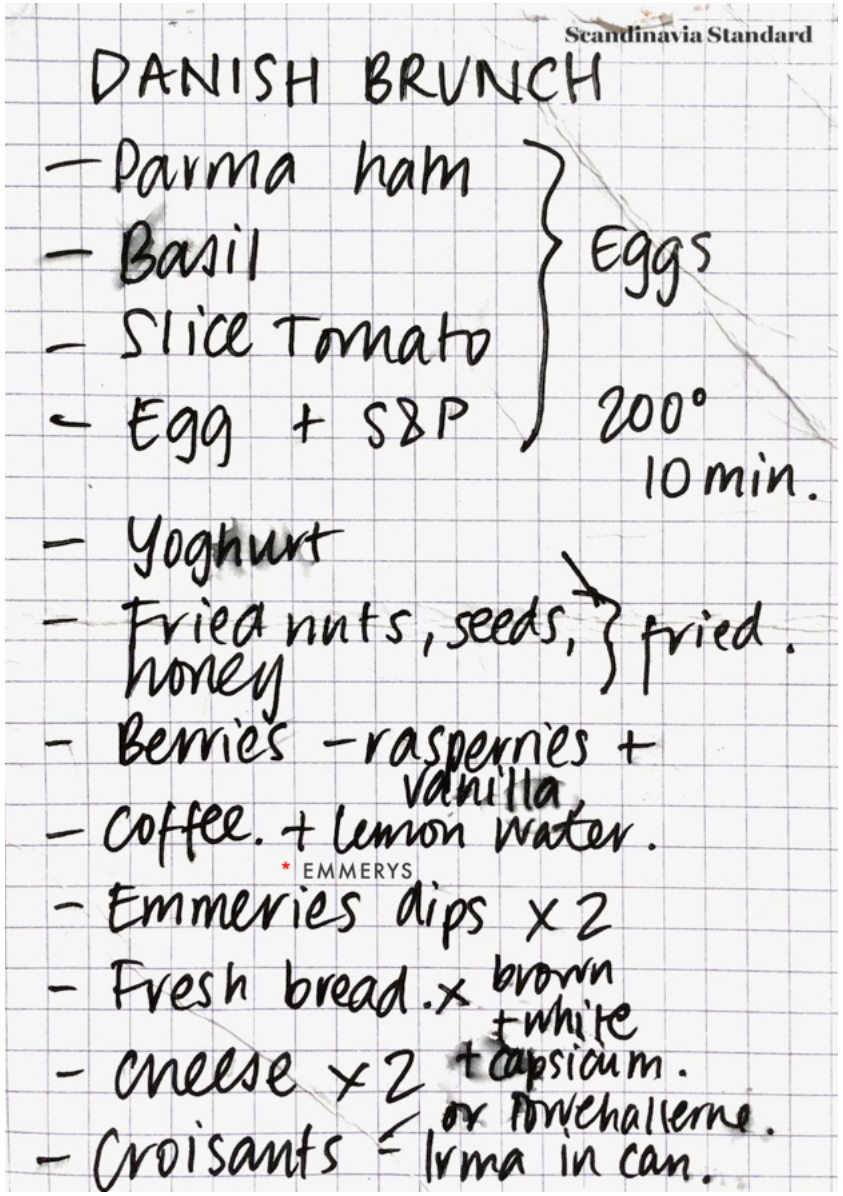
- Deep Neural Networks (same as neural networks but deep ☺)
  - One paper on feature extraction (Sun, Konstantelos, Strbac, 2018)
  - One paper inspired by image processing (Hidalgo, Hancharou, Thams, Chatzivasileiadis, 2019)
  - Few additional papers over the past 6 months

Machine learning applications
(for power system security assessment)
A very short overview

The ingredients

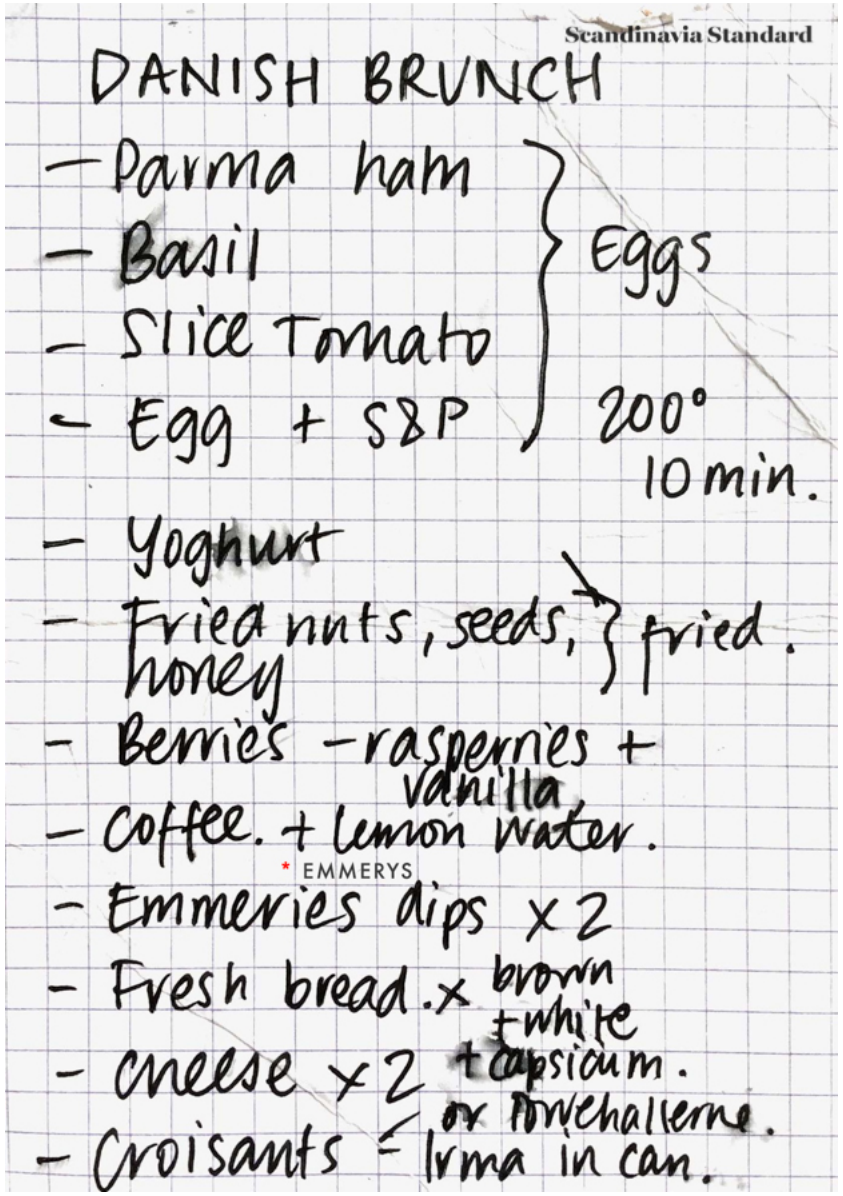
What do we need?

If we want to apply machine learning approaches for power system security



The ingredients

- A training database
- A training algorithm
- A test database
  - To test accuracy of the approach



Test Database

Test Database

Traditionally:

- Split training database to e.g. 80% training samples and 20% test samples
- Train with the 80%
- Test with the 20%

Modern toolboxes have this integrated and automatized → only need to provide a training database

Point to remember:

The test database determines the performance of your method. If the test data come from the same simulations as your training data, the accuracy can be deceivingly high. Would it be equally high in reality?

Ideally → use a different real-life dataset

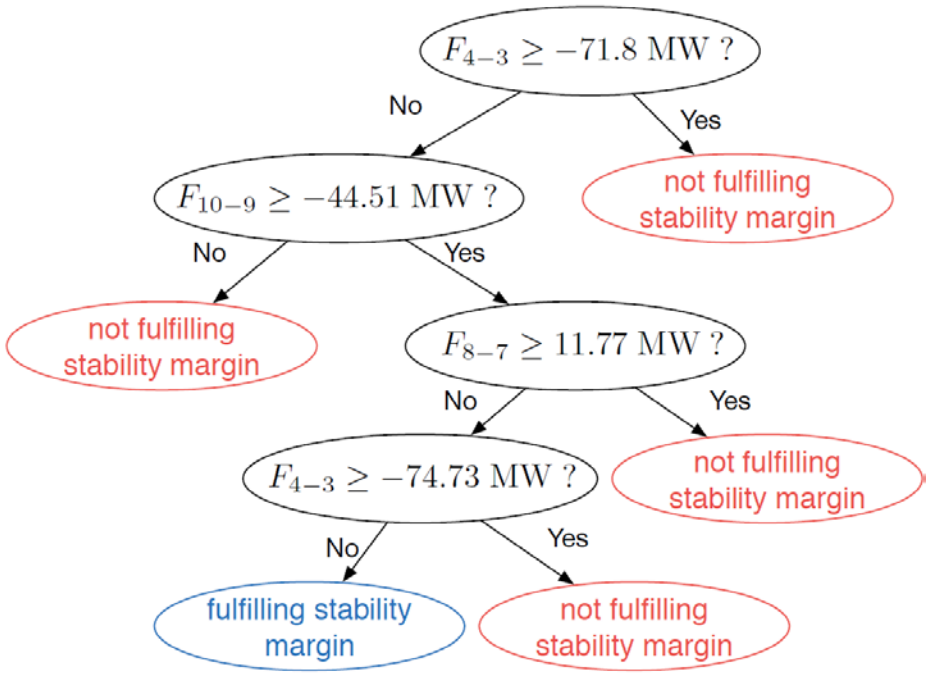
(Unfortunately, not always possible)

Takeaway #2

The quality of your test database is crucial: the test database determines the performance of your method; for a valid assessment, it needs to include a wide range of operating conditions with the same frequency of occurrence as in real-life

Training a neural network

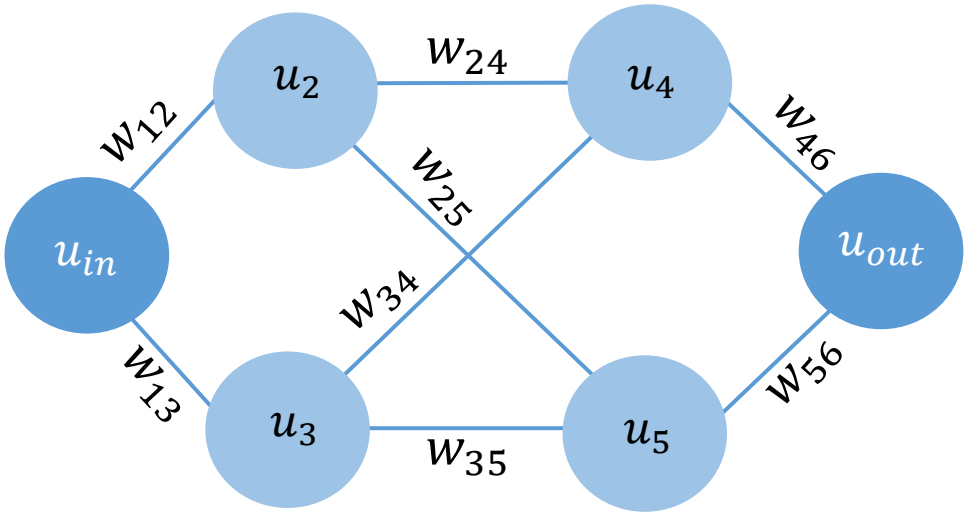
Decision Trees



- Decision rules
  - If line_flow>FF then …. else…

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.

Neural Networks



- Linear weights
- On every node: a non-linear activation function
  - e.g RELU: $\boldsymbol{u_j = \max(0, w_{ij}x_i + b_i)}$
  - Many other possibilities, e.g. sigmoid function and others

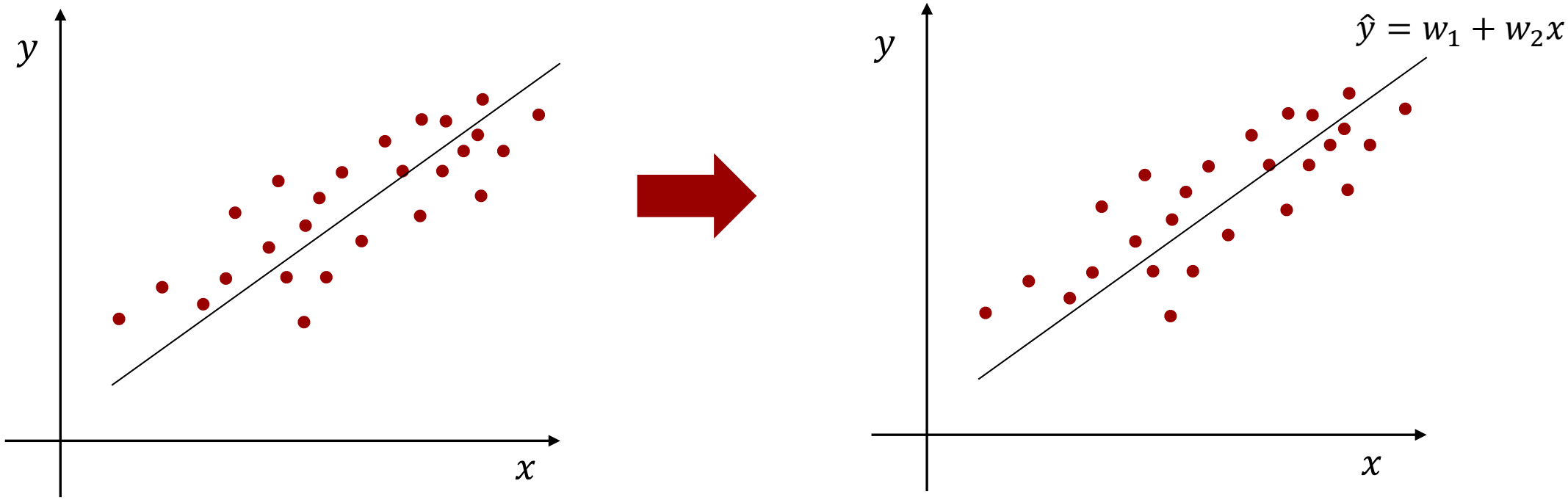
Training a neural network

- Batch size vs Iterations vs Epochs
- Loss function
- Evaluation of performance
  - Confusion matrix
  - Performance metrics

Batch size vs Iterations vs Epochs

Example from linear regression; neural network works similarly

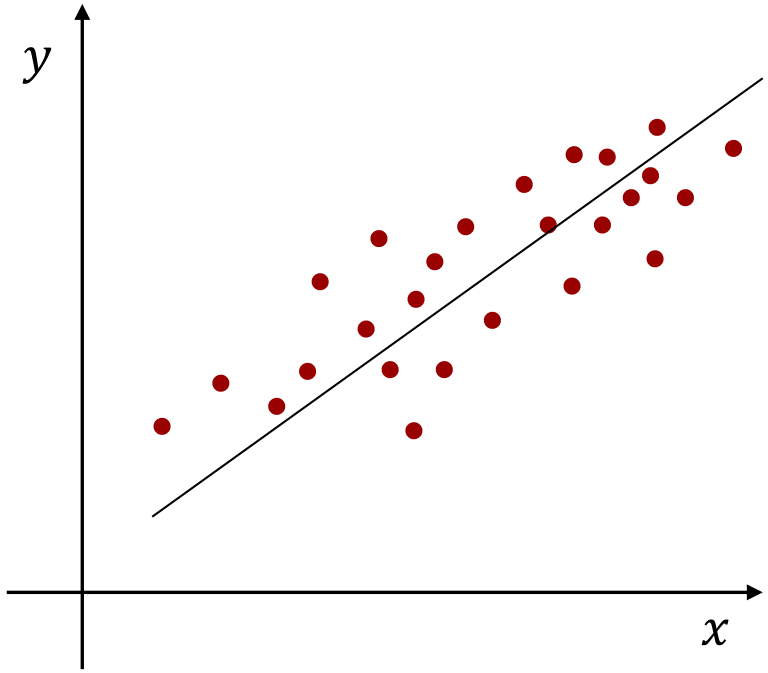
Goal: *estimate* w_1, w_2 *to fit* $\hat{y} = w_1 + w_2 x$



Batch size vs Iterations vs Epochs

Example from linear regression; neural network works similarly

Goal: *estimate* w_1, w_2 *to fit* $\hat{y} = w_1 + w_2 x$

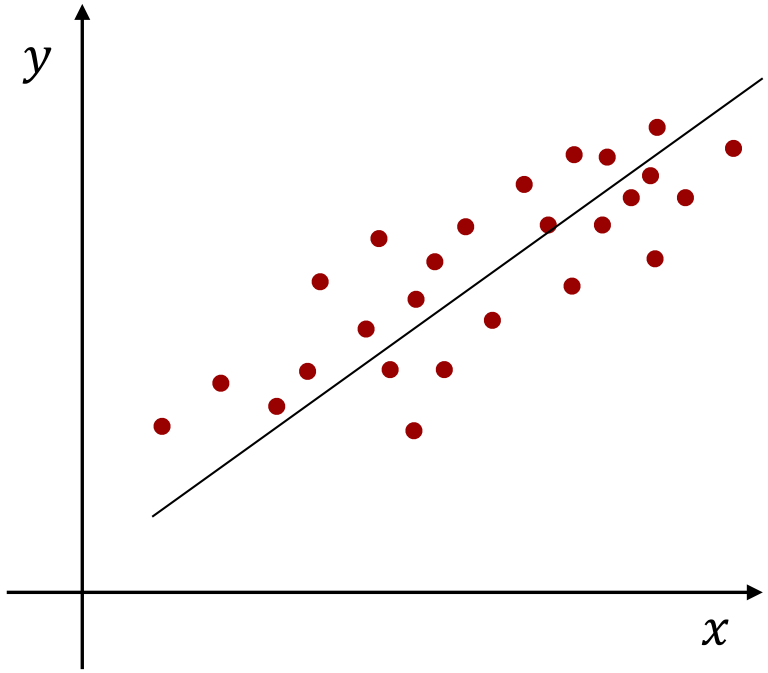


- Ideally we want to enter all our data in our linear regression algorithm to estimate w_1, w_2
- But, the size of data can be immense!
- That's why we need to split our dataset in batches

Batch size vs Iterations vs Epochs

Example from linear regression; neural network works similarly

Goal: *estimate* w_1, w_2 *to fit* $\hat{y} = w_1 + w_2 x$



- Ideally we want to enter all our data in our linear regression algorithm to estimate w_1, w_2
- But, the size of data can be immense!
- That's why we need to split our dataset in batches
- Total #data = batch_size * iterations

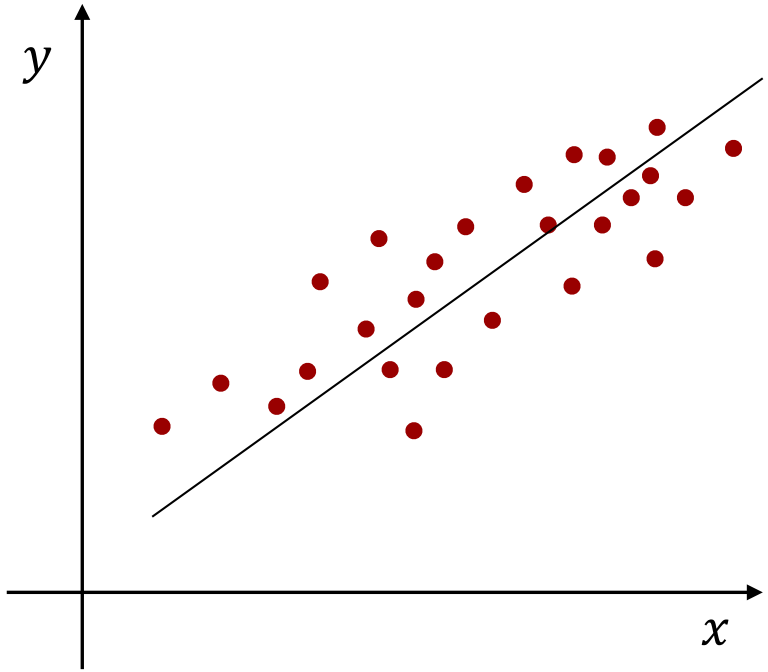
e.g. if 1000 datapoints and batch_size=200,

then 1000=200*5 iterations

Batch size vs Iterations vs Epochs

Example from linear regression; neural network works similarly

Goal: *estimate* w_1, w_2 *to fit* $\hat{y} = w_1 + w_2 x$



More info:
https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9

- Ideally we want to enter all our data in our linear regression algorithm to estimate w_1, w_2
- But, the size of data can be immense!
- That's why we need to split our dataset in batches
- Total #data = batch_size * iterations

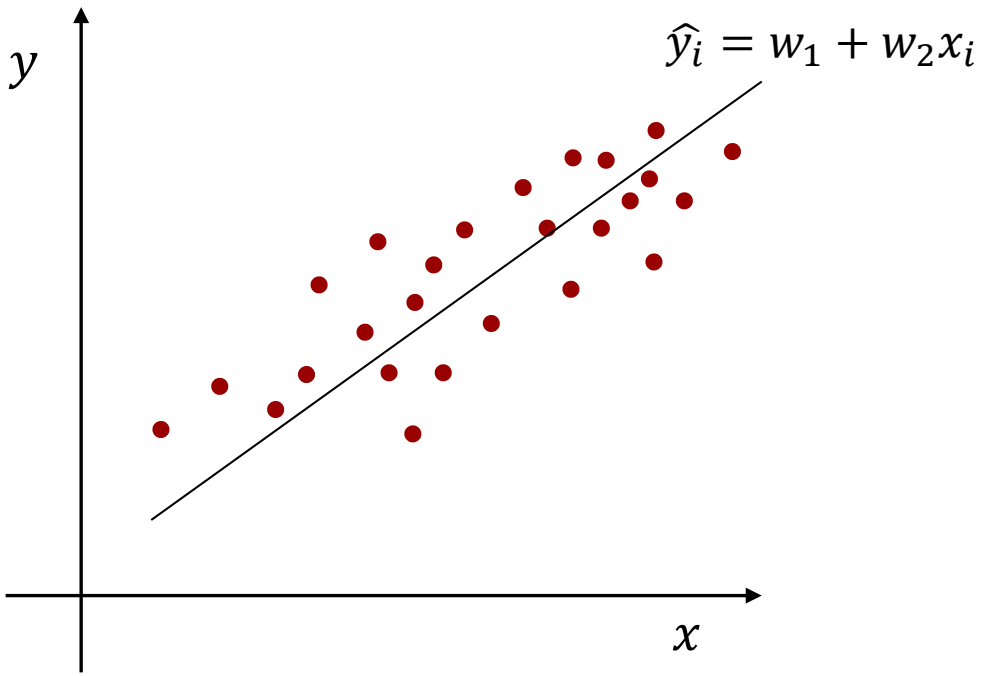
  e.g. if 1000 datapoints and batch_size=200,

  then 1000=200*5 iterations
- Epochs: passing the whole dataset only once is not enough!
  - 1 Epoch = passing the dataset through the neural network ONCE

Loss function:
To find the best estimates for w_1, w_2 we need to run an optimization!

y_i: actual/correct value

$\hat{y}_i$: estimated value



$$\min_{w_1, w_2} \quad \|y_i - \hat{y}_i\|$$

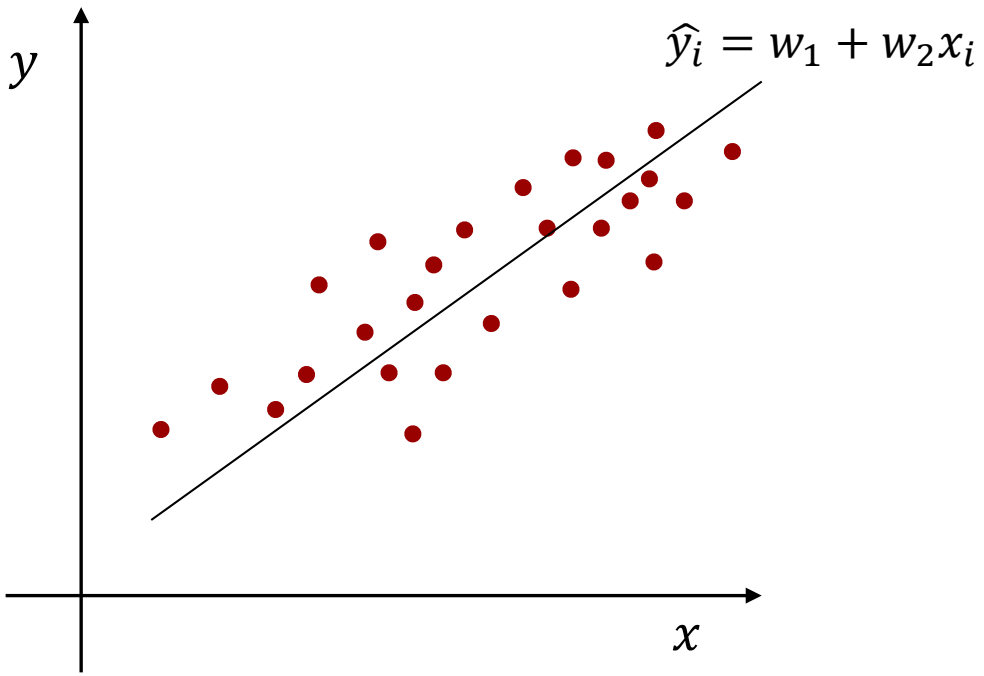
s.t.

$$\hat{y}_i = w_1 + w_2 x_i \quad \forall i$$

Loss function:
To find the best estimates for w_1, w_2 we need to run an optimization!

y_i: actual/correct value

$\hat{y}_i$: estimated value



$$\min_{w_1, w_2} \quad \|y_i - \hat{y}_i\|$$

s.t.

$$\hat{y}_i = w_1 + w_2 x_i \quad \forall i$$

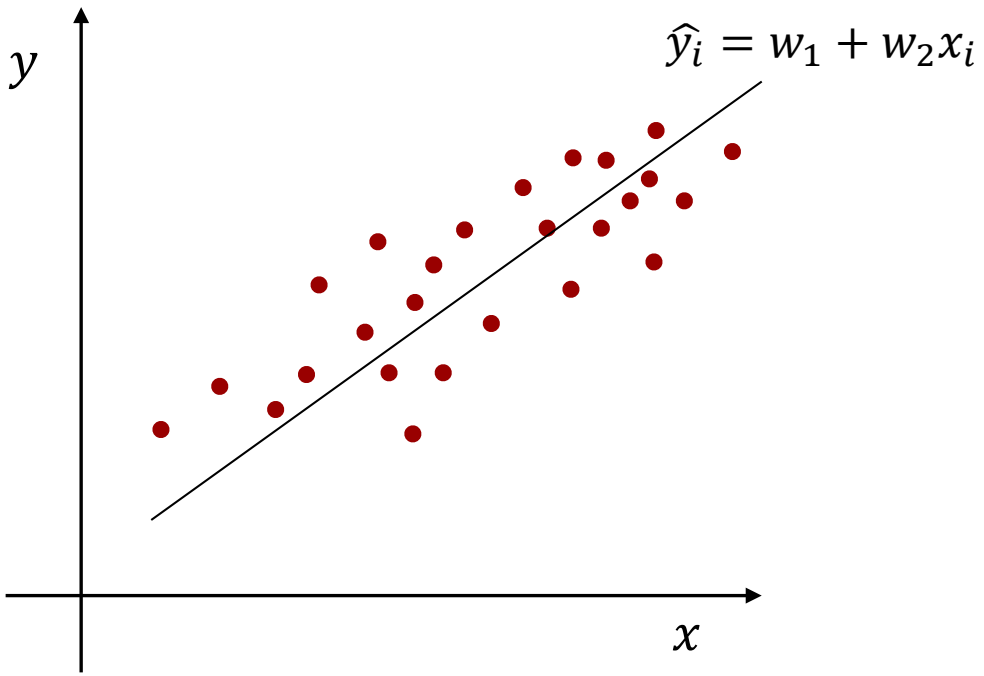
Rewrite:

$$\min_{w_1, w_2} \quad \|y_i - (w_1 + w_2 x_i)\| \quad \forall i$$

Loss function:
To find the best estimates for w_1, w_2 we need to run an optimization!

y_i: actual/correct value

$\hat{y}_i$: estimated value



$$\min_{w_1, w_2} \quad \|y_i - \hat{y}_i\|$$

s.t.

$$\hat{y}_i = w_1 + w_2 x_i \quad \forall i$$

Rewrite:

$$\min_{w_1, w_2} \quad \|y_i - (w_1 + w_2 x_i)\| \quad \forall i$$

What cost function shall we use?

Cost function (often called "Loss function")

- A series of different options:
  https://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html
  https://isaacchanghau.github.io/post/loss_functions/

- Often used for classification problems: cross-entropy

- For 2 classes, i.e. $y_i = \{0,1\}$ and $\widehat{y_i} = [0,1]$:

$$\mathcal{L} = -\frac{1}{n}\sum_{i=1}^{n} y_i \log \widehat{y_i} + (1 - y_i)\log(1 - \widehat{y_i})$$

Cross-entropy: the insights

- Insight #1: you minimize over the estimation errors of all points in the batch

$$\mathcal{L} = -\frac{1}{n}\sum_{i=1}^{n} y_i \log \hat{y}_i \ + (1 - y_i)\log(1 - \hat{y}_i)$$

- Insight #2:
  - Actual values: either 0 or 1
  - Estimated values: between 0 and 1

$$y_i = \{0,1\} \quad but \ \hat{y}_i = [0,1]$$

Cross-entropy: the insights

- Insight #1: you minimize over the estimation errors of all points in the batch

$$\mathcal{L} = -\frac{1}{n}\sum_{i=1}^{n} y_i \log \hat{y}_i \ + (1 - y_i)\log(1 - \hat{y}_i)$$

- Insight #2:
  - Actual values: either 0 or 1
  - Estimated values: between 0 and 1

$$y_i = \{0,1\} \quad but \ \hat{y}_i = [0,1]$$

- Insight #3: Why the minus?

Hint: explain it for one point

$$\mathcal{L} = -(y_i \log \hat{y}_i \ + (1 - y_i)\log(1 - \hat{y}_i))$$

Cross-entropy: the insights

- Insight #1: you minimize over the estimation errors of all points in the batch

$$\mathcal{L} = -\frac{1}{n}\sum_{i=1}^{n} y_i \log \hat{y}_i \ + (1 - y_i)\log(1 - \hat{y}_i)$$

- Insight #2:
  - Actual values: either 0 or 1
  - Estimated values: between 0 and 1

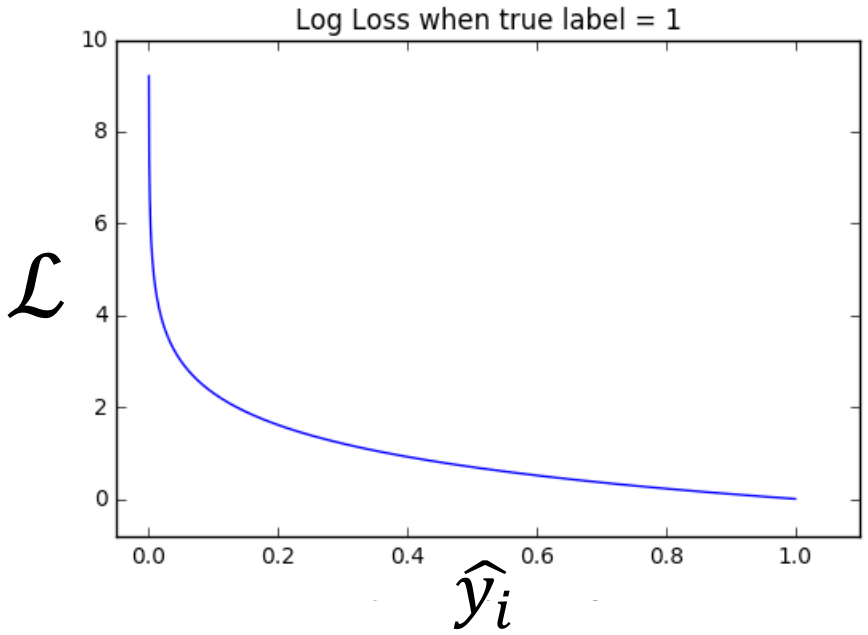
$$y_i = \{0,1\} \quad but \ \hat{y}_i = [0,1]$$

- Insight #3: Why the minus?

  Hint: explain it for one point

$$\mathcal{L} = -(y_i \log \hat{y}_i \ + (1 - y_i)\log(1 - \hat{y}_i))$$

- Insight #4: y_i=1, $\mathcal{L}$ *when* $\hat{y}_i = [0,1]$



Evaluating the performance: Confusion matrix

		Target class (actual values)	
		1	0
Output class (predicted values)	1	True positive (TP)	False positive (FP)
	0	False negative (FN)	True Negative (TN)

Evaluating the performance: Confusion matrix

		Target class (actual values)	
Output class (predicted values)		1	0
	1	True positive (TP)	False positive (FP)
	0	False negative (FN)	True Negative (TN)

Classification for power system security

	Actually Safe	Actually Unsafe
Predicted safe	True positive (TP)	False positive (FP)
Predicted Unsafe	False negative (FN)	True Negative (TN)

Performance Metrics: Accuracy

	Actually Safe	Actually Unsafe
Predicted safe	True positive (TP)	False positive (FP)
Predicted Unsafe	False negative (FN)	True Negative (TN)

- Accuracy: The proportion of correct classifications in the whole dataset

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN}$$

- **Example:** Assume 1000 datapoints
  - Actually safe: 500 TP=480 FP=30
  - Actually unsafe: 500 FN=20 TN=470

Accuracy = ?

Performance Metrics: Accuracy

	Actually Safe	Actually Unsafe
Predicted safe	True positive (TP)	False positive (FP)
Predicted Unsafe	False negative (FN)	True Negative (TN)

- Accuracy: The proportion of correct classifications in the whole dataset

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN}$$

- **Example:** Assume 1000 datapoints
  - Actually safe: 500 TP=480 FP=30
  - Actually unsafe: 500 FN=20 TN=470

$$\text{Accuracy} = \frac{480+470}{480+20+470+30} = 95\%$$

Evaluating performance by measuring only accuracy is often not enough

Why?

Performance Metrics: Accuracy

	Actually Safe	Actually Unsafe
Predicted safe	True positive (TP)	False positive (FP)
Predicted Unsafe	False negative (FN)	True Negative (TN)

- Accuracy: The proportion of correct classifications in the whole dataset

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN}$$

- **Example:** Assume 1000 datapoints
  - Actually safe: 500 TP=480 FP=30
  - Actually unsafe: 500 FN=20 TN=470

$$\text{Accuracy} = \frac{480+470}{480+20+470+30} = 95\%$$

Evaluating performance by measuring **only accuracy** is often **not enough**

Why?

- **Example:** Assume 1000 datapoints
  - Actually safe: 20 TP= 1 FP=30
  - Actually unsafe: 980 FN= 19 TN=950

Accuracy = ?

Performance Metrics: Accuracy

	Actually Safe	Actually Unsafe
Predicted safe	True positive (TP)	False positive (FP)
Predicted Unsafe	False negative (FN)	True Negative (TN)

- Accuracy: The proportion of correct classifications in the whole dataset

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN}$$

- **Example:** Assume 1000 datapoints
  - Actually safe: 500 TP=480 FP=30
  - Actually unsafe: 500 FN=20 TN=470

$$\text{Accuracy} = \frac{480+470}{480+20+470+30} = 95\%$$

Evaluating performance by measuring only accuracy is often not enough

Why?

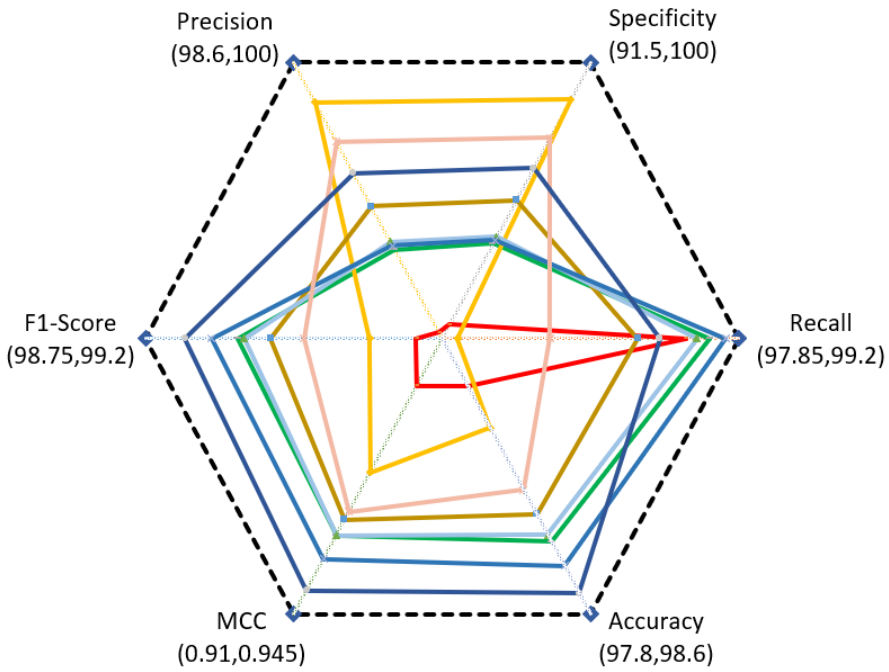
- **Example:** Assume 1000 datapoints
  - Actually safe: 20 TP= 1 FP=30
  - Actually unsafe: 980 FN= 19 TN=950

$$\text{Accuracy} = \frac{1+950}{1+19+950+30} = 95\%$$

- 95% accurate but we have misclassified almost all truly safe points!
- For heavily unbalanced data, accuracy is not sufficient!

Performance metrics

- Accuracy
- Recall: True Positive Rate $\frac{TP}{TP+FN}$
- Specificity: True Negative Rate $\frac{TN}{TN+FP}$
- Precision: Positive Predictive Value $\frac{TP}{TP+FP}$
- F1: harmonic mean of Precision and Recall $\mathrm{F1} = \frac{Precision \cdot Recall}{Precision+Recall}$
- MCC (Matthews correlation coefficient)
  (only for binary classification – 2 classes only)
  - MCC=1 → perfect prediction
  - MCC=0 → random (like flipping a coin)
  - MCC= –1→ Completely mistaken



Hidalgo-Arteaga, Hancharou, Thams, Chatzivasileiadis, Powertech 2019

$$\mathrm{MCC} = \frac{(TP \cdot TN) - (FP \cdot FN)}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

Key hints for your implementation

- Regularization: Training of neural networks work better if you normalize your inputs
  - Try to normalize your active power setpoints (e.g. if PG1 =30 MW and PG1max = 100 MW, then PG1=0.3)

- 1-hot encoding: Neural networks work better if you use *one vector for each class*

Instead of:

	Safe=1 Unsafe=0
x_1	0
x_2	1
x_3	0

	Safe	Unsafe
x_1	0	1
x_2	1	0
x_3	0	1

Takeaway #3:
Neural networks (or Decision Trees) for classification: you need a balanced training database → similar number of safe and unsafe points

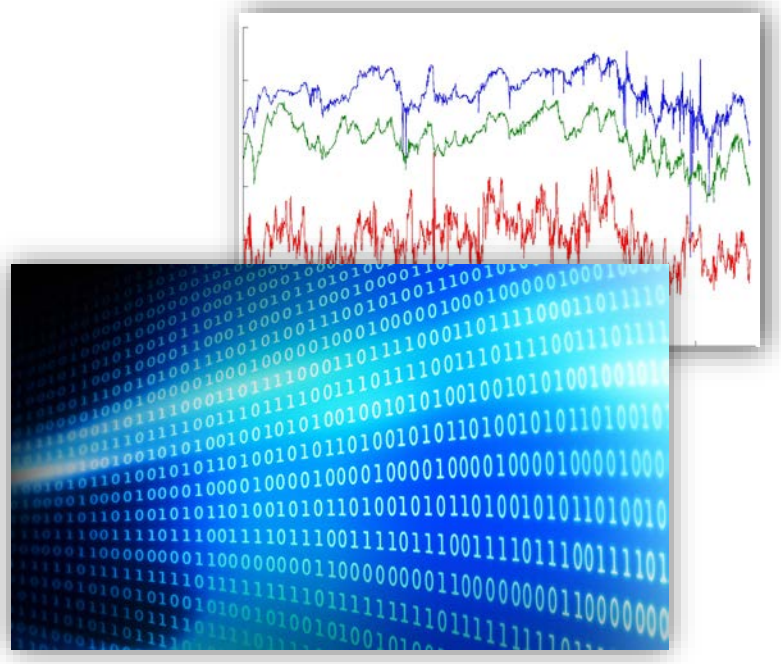
Takeaway #4:
Accuracy is not sufficient to assess the NN/DT performance. We need additional metrics

Takeaway #5:
Neural Network training requires additional “tricks” to boost its performance (e.g. 1-hot encoding/regularization)

Training database

We need data!

- Data that accurately capture the whole security region
  - so that we can successfully use machine learning approaches for classification
- Historical data are insufficient
  - Contain very limited number of abnormal situations
- Need to generate simulation data
- Assessing the stability of 100’000s of operating points is an extremely demanding task
  - Immense search space



Sampling beyond Statistics: Efficient Database Generation

- Modular and highly efficient algorithm
- Can accommodate numerous definitions of power system security (e.g. N-1, N-k, small-signal stability, voltage stability, transient stability, **or a combination** of them)
- **10-20 times faster** than existing state-of-the-art approaches
- Our use case: N-1 security + small-signal stability
- Generated Database for NESTA 162-bus system online available!

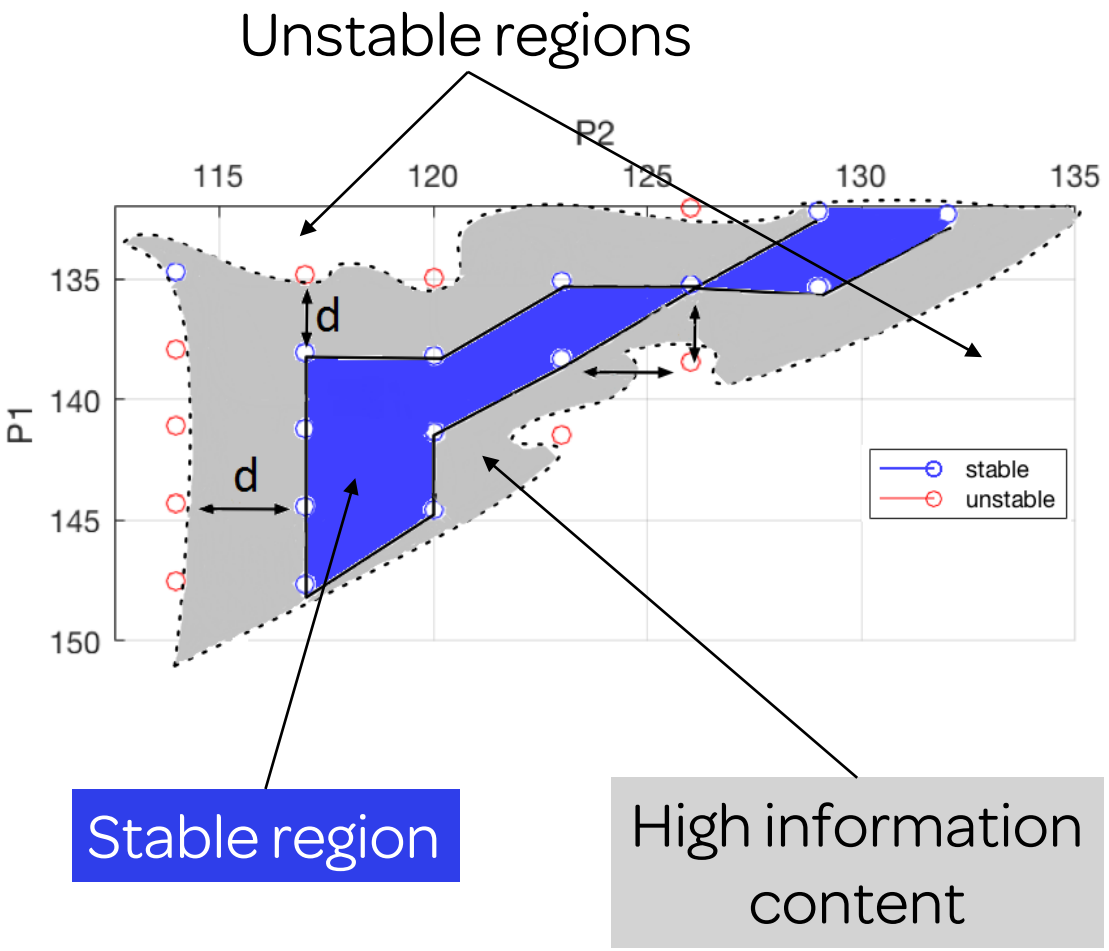
  https://github.com/johnnyDEDK/OPs_Nesta162Bus (>500,000 points)

F. Thams, A. Venzke, R. Eriksson, and S. Chatzivasileiadis, "Efficient database generation for data-driven security assessment of power systems". ". IEEE Trans. Power Systems, vol. 35, no. 1, pp. 30-41, Jan. 2020. https://www.arxiv.org/abs/1806.0107.pdf

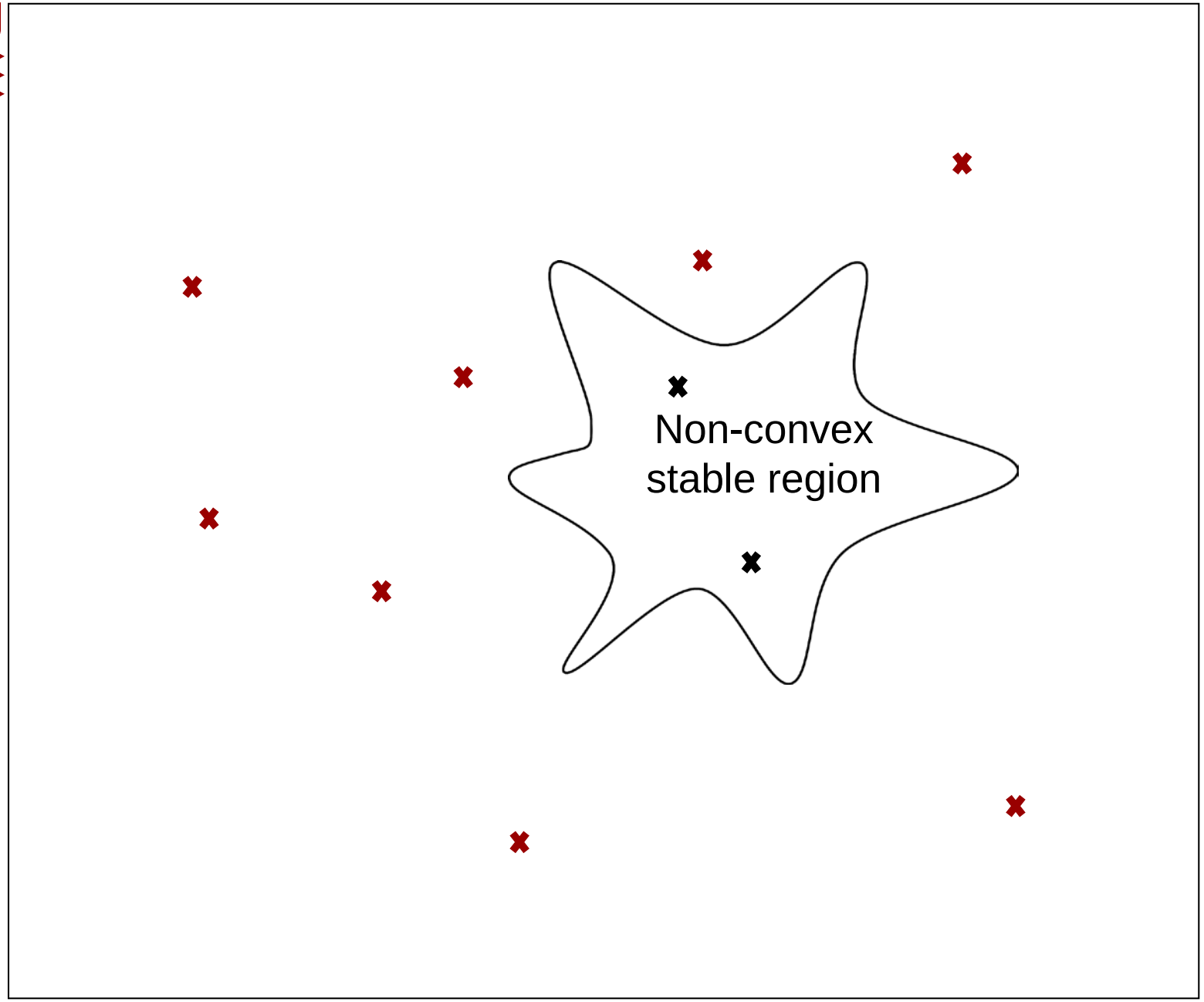
Sampling beyond Statistics: Efficient Database Generation

- The goal
  - Focus on the boundary between stability and instability
  - We call it: "high information content" region

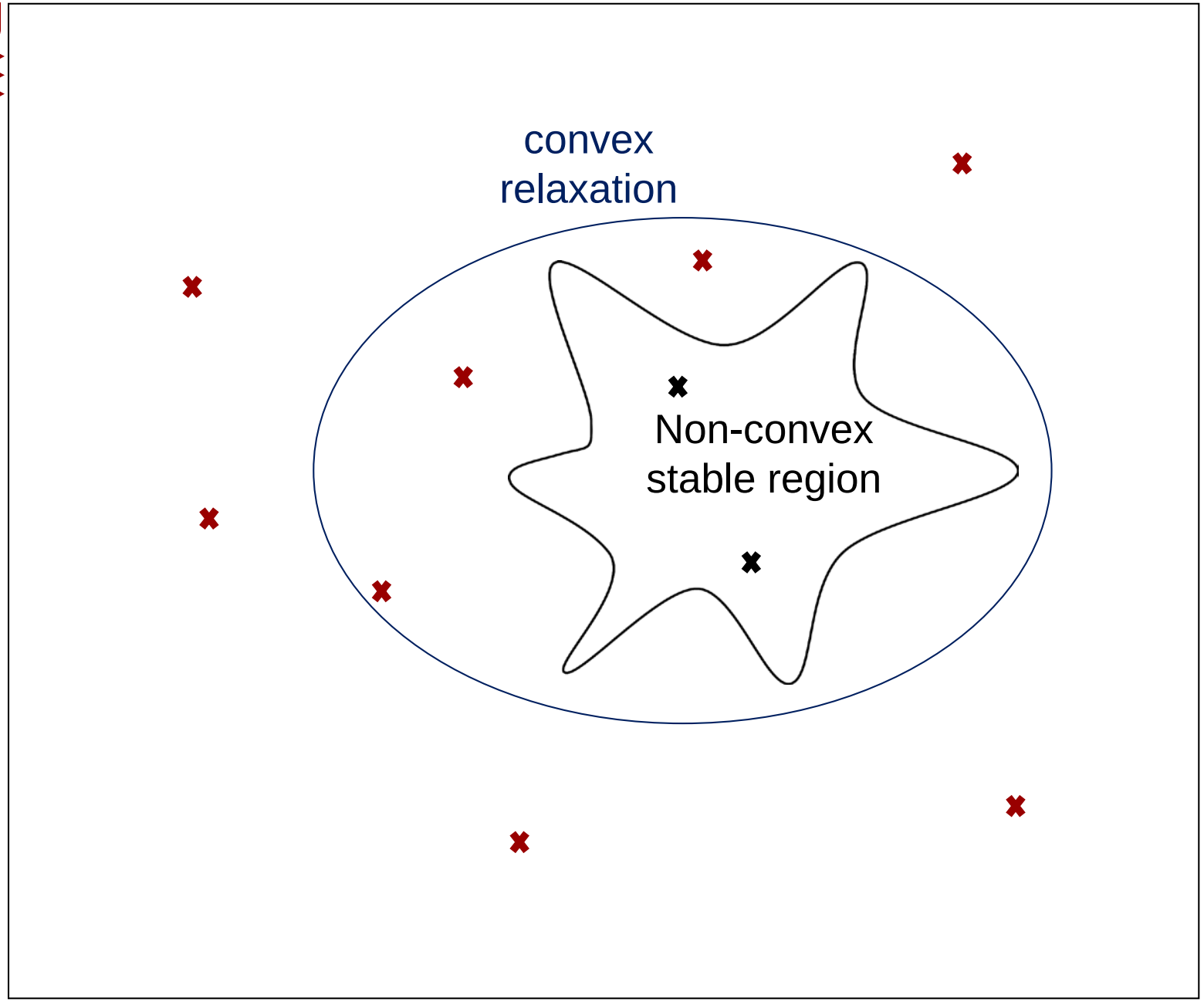
- How?
  1. Using convex relaxations
  2. And "Directed Walks"



Real data for the IEEE 14-bus system
N-1 security and small-signal stability

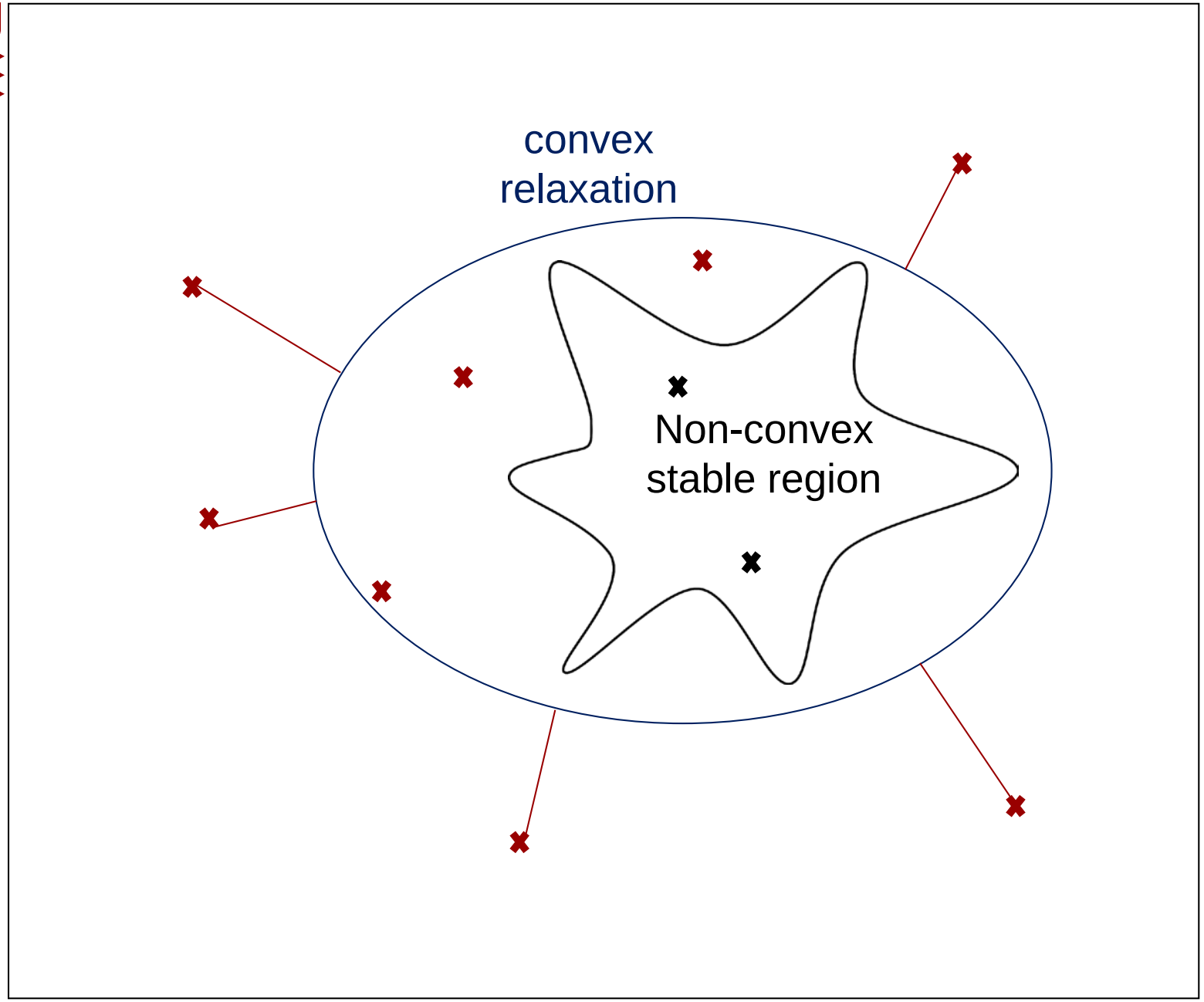


Convex relaxations to discard infeasible regions



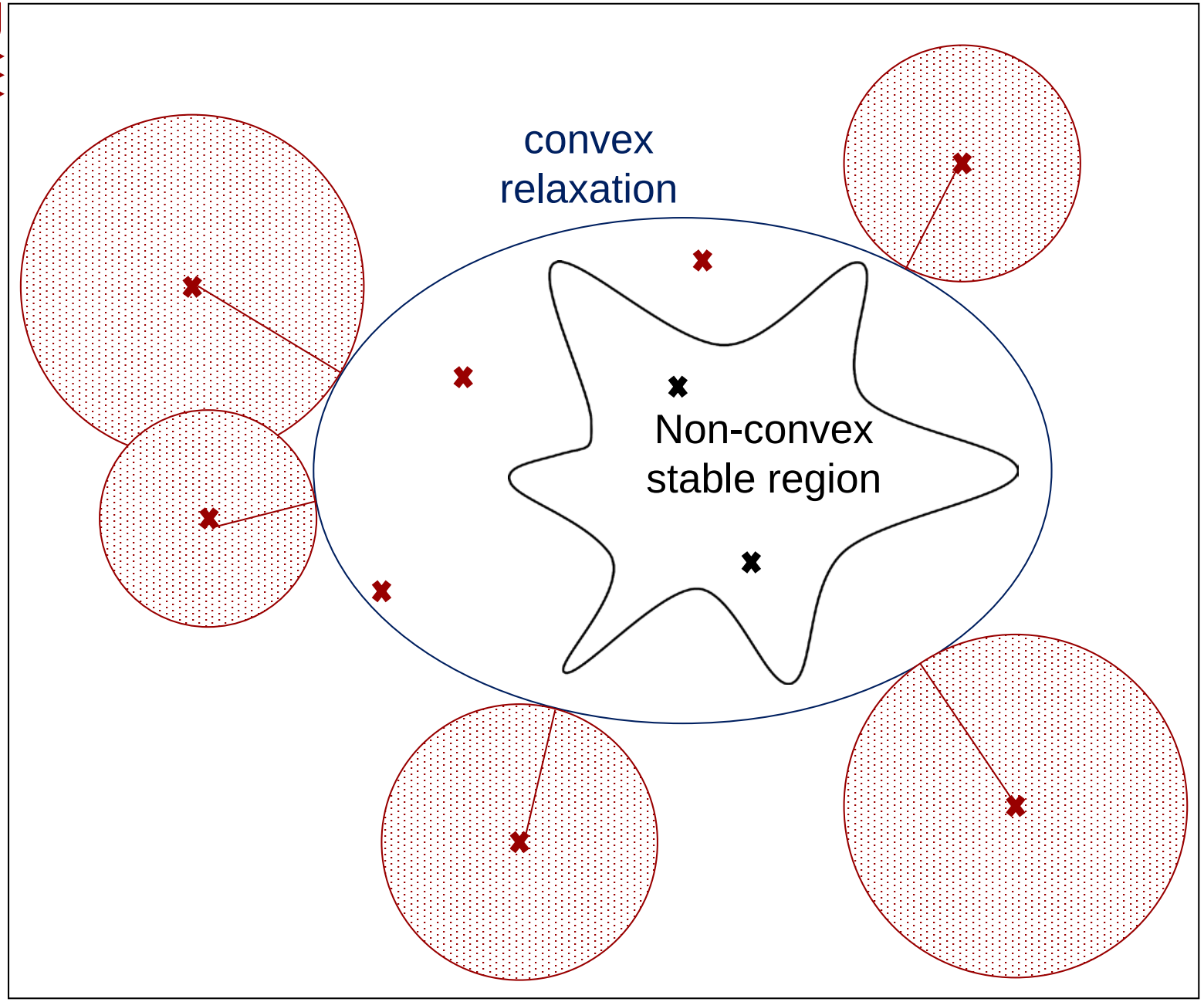
Convex relaxations to discard infeasible regions

- Certificate: if point infeasible for semidefinite relaxation → infeasible for the original problem



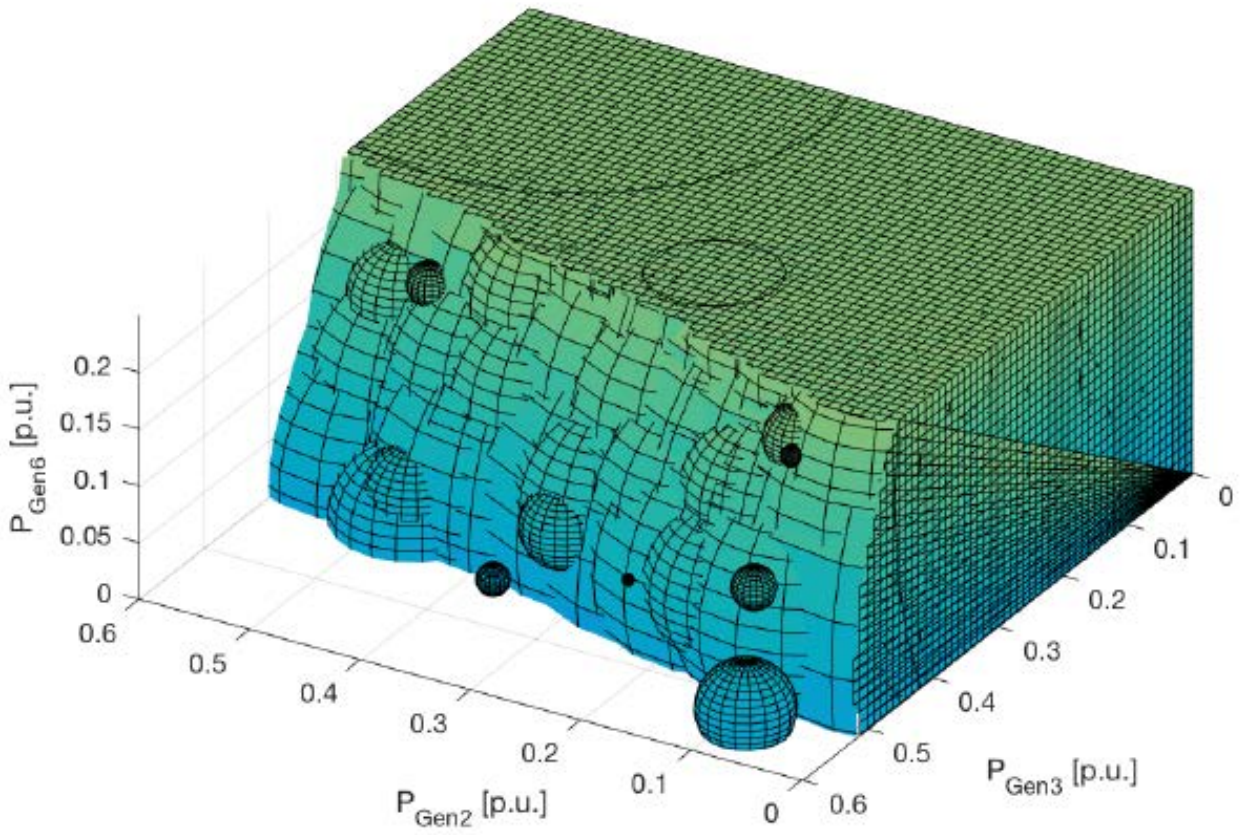
Convex relaxations to discard infeasible regions

- Certificate: if point infeasible for semidefinite relaxation → infeasible for the original problem
- If infeasible point: find minimum radius to feasibility



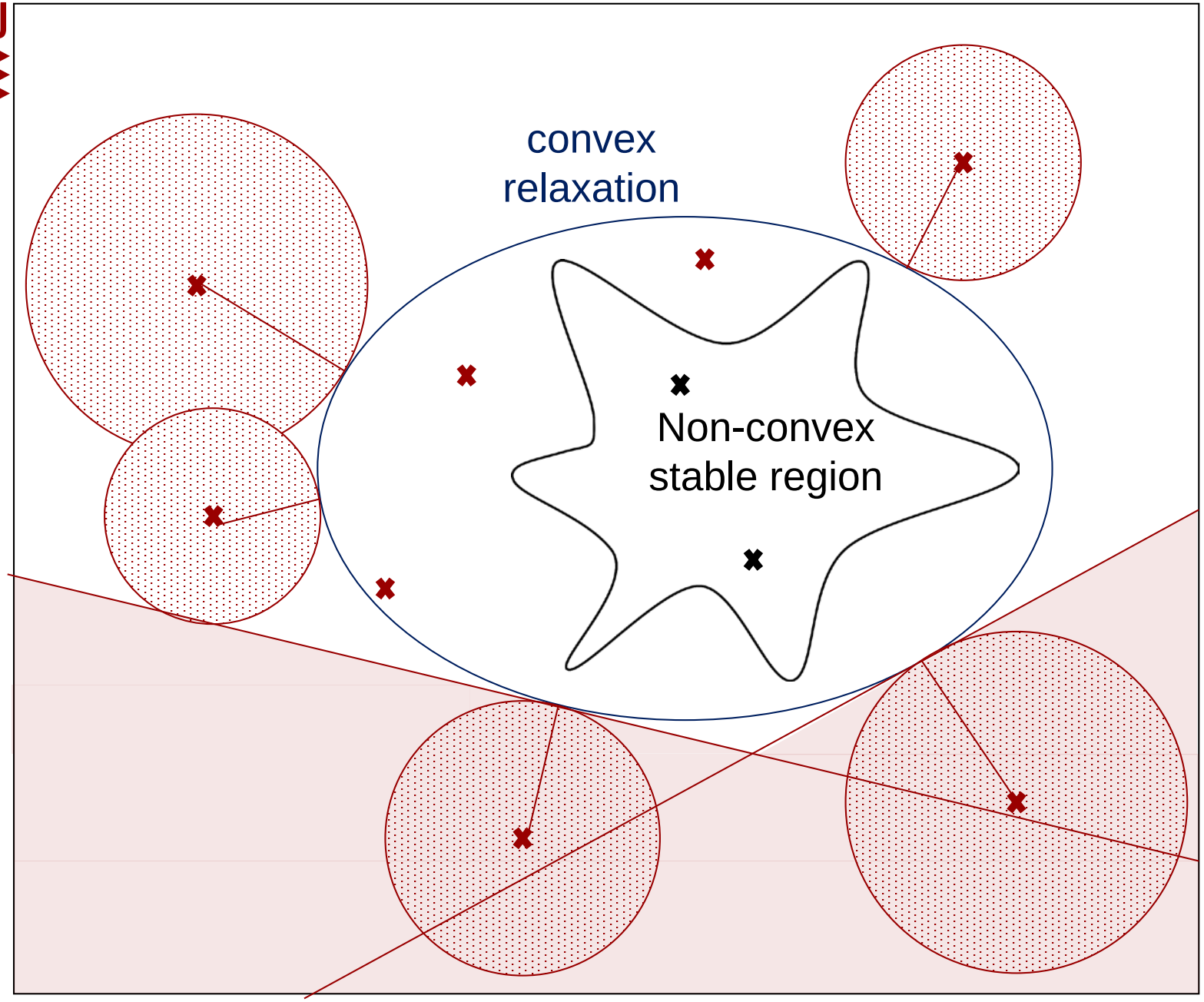
Convex relaxations to discard infeasible regions

- Certificate: if point infeasible for semidefinite relaxation → infeasible for the original problem
- If infeasible point: find minimum radius to feasibility
- Discard all points inside the (hyper)sphere



- 3D projection of hyperspheres
- IEEE 14-bus system
- Rapidly discarding (=classifying) large chunks of the search space as infeasible to focus on the boundary

F. Thams, A. Venzke, R. Eriksson, and S. Chatzivasileiadis, "Efficient database generation for data-driven security assessment of power systems". IEEE Trans. Power Systems, vol. 35, no. 1, pp. 30-41, Jan. 2020.. https://www.arxiv.org/abs/1806.0107.pdf



Convex relaxations to discard infeasible regions

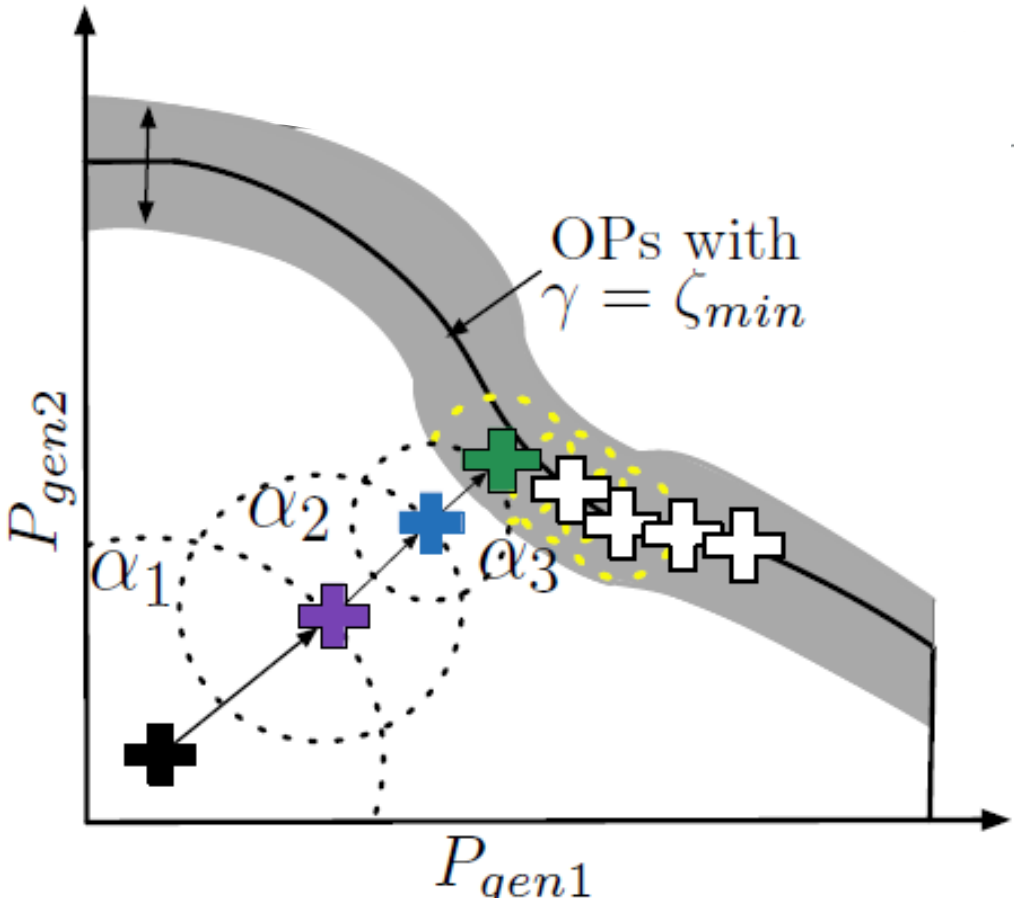
- Extension of this work to hyperplanes

- A. Venzke, D.K. Molzahn, S. Chatzivasileiadis, **Efficient Creation of Datasets for Data-Driven Power System Applications**. Accepted at PSCC 2020. https://arxiv.org/pdf/1910.01794.pdf

Directed Walks

- "Directed walks": steepest-descent based algorithm to explore the remaining search space, focusing on the area around the security boundary

  1. Variable step-size
  2. Parallel computation
  3. Full N-1 contingency check



Results

	Points close to the security boundary (within distance γ)	
	IEEE 14-bus	NESTA 162-bus
Brute Force	100% of points in **556.0 min**	*intractable*
Importance Sampling	100% of points in **37.0 min**	**901 points** in 35.7 hours
Proposed Method	100% of points in **3.8 min**	**183'295 points** in 37.1 hours

- We tested these databases with decision trees. Further benefits for the decision trees:
  - Higher accuracy
  - Better classification quality (Matthews correlation coefficient)

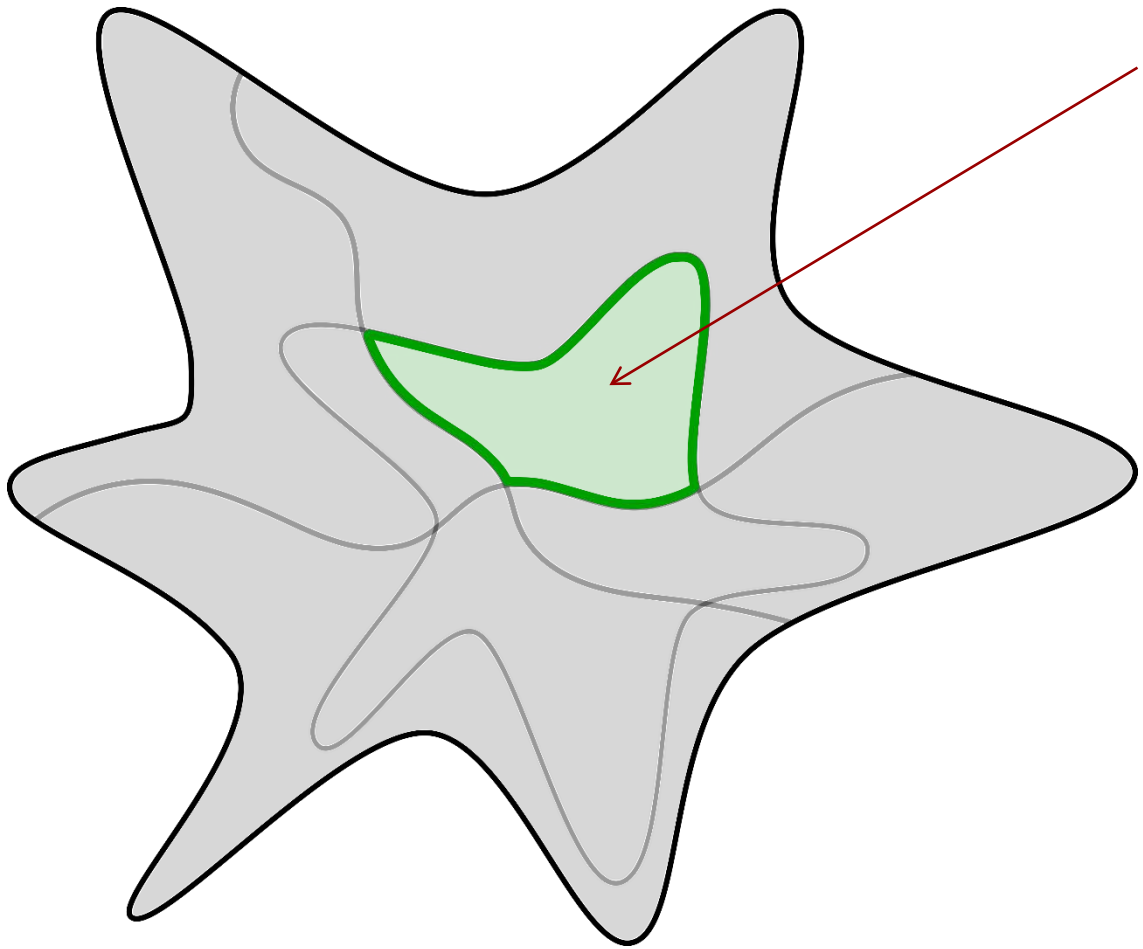
Generated Database for NESTA 162-bus system online available!
https://github.com/johnnyDEDK/OPs_Nesta162Bus

Takeaway #6

Creating high-quality training databases is extremely complex and an open research topic. We need to go beyond purely statistical methods and exploit the underlying physics during sampling.

From decision trees to mixed integer linear programming

The feasible space of power system operations



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

Electricity markets should determine setpoints that are only within this area

Optimization constraints should represent this area

Impossible → differential and non-linear algebraic equations

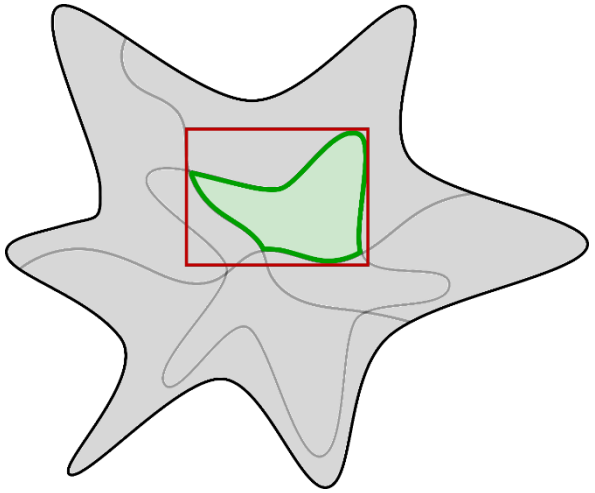
What do TSOs and market operators do?

Linear approximations

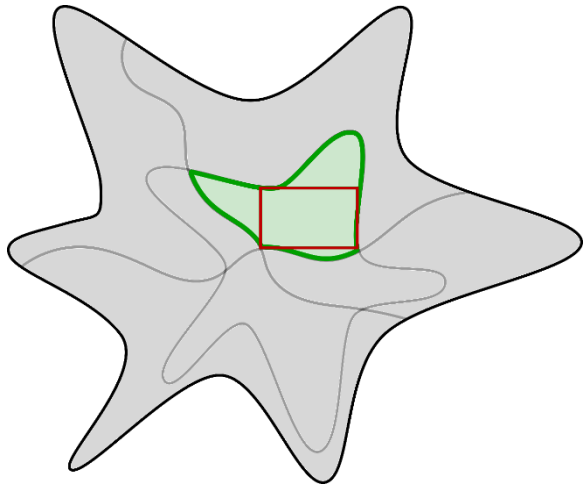
Net Transfer Capacity

Flow-based market coupling

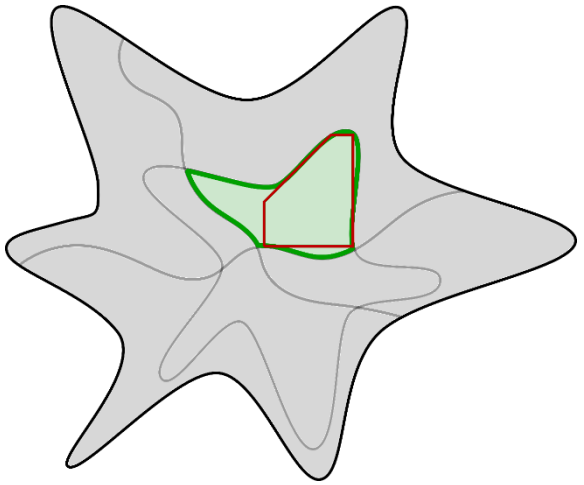
Inaccurate



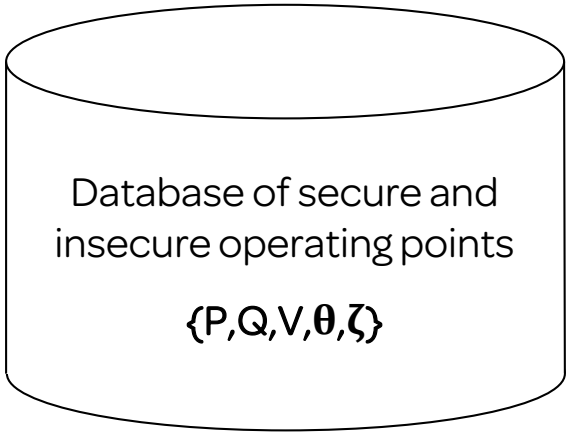
Too conservative

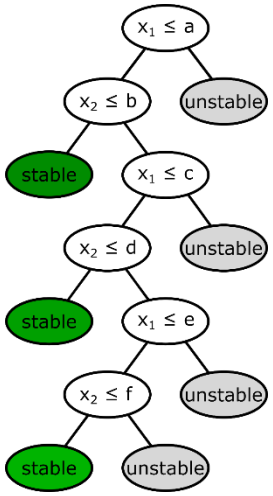


Single convex region



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





$$\mathbf{PTDF} \cdot \left(\mathbf{P_G} - \mathbf{P_D}\right) \leq \mathbf{F}^{\max}_{\mathbf{L,p}} y_p + \mathbf{F}^{\max}_{\mathbf{L}}(1 - y_p)$$
$$\mathbf{PTDF} \cdot \left(\mathbf{P_G} - \mathbf{P_D}\right) \geq \mathbf{F}^{\min}_{\mathbf{L,p}} y_p - \mathbf{F}^{\max}_{\mathbf{L}}(1 - y_p)$$

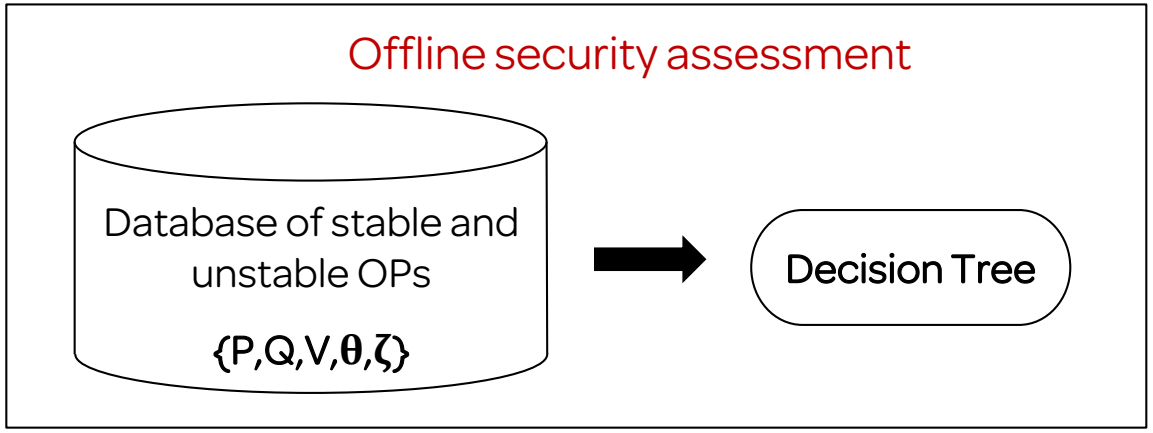
Operating points provided by the TSOs through simulated and real data

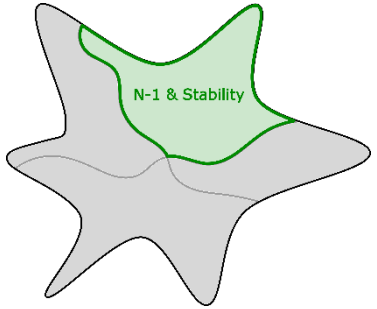
Train a decision tree to classify secure and insecure regions

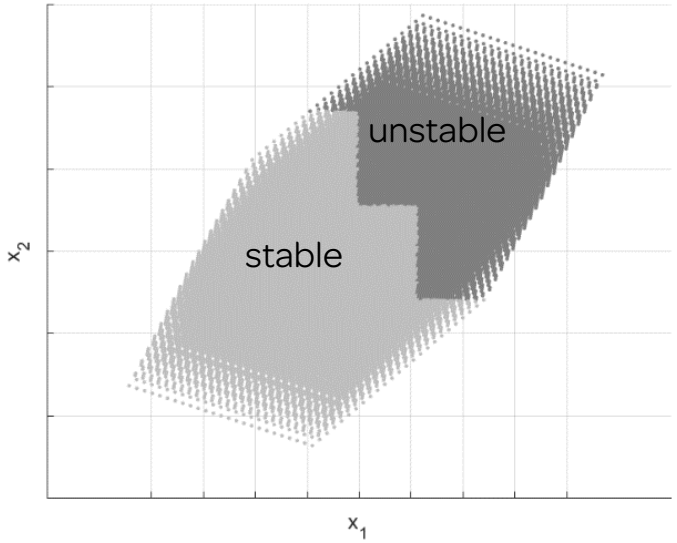
Exact reformulation to MILP

Thams et al IREP 2017, Halilbasic et al PSCC 2018

Data-driven security-constrained OPF

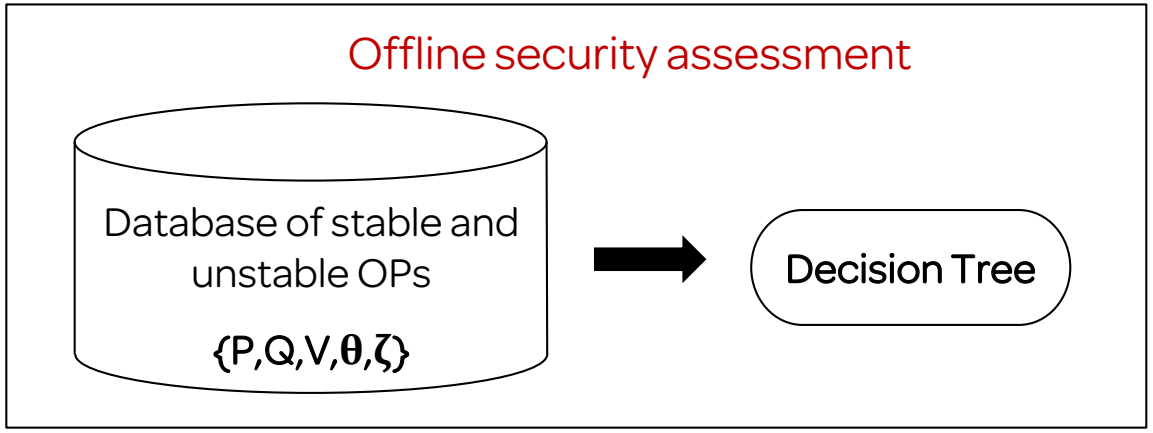


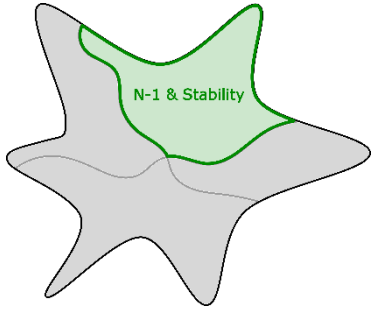




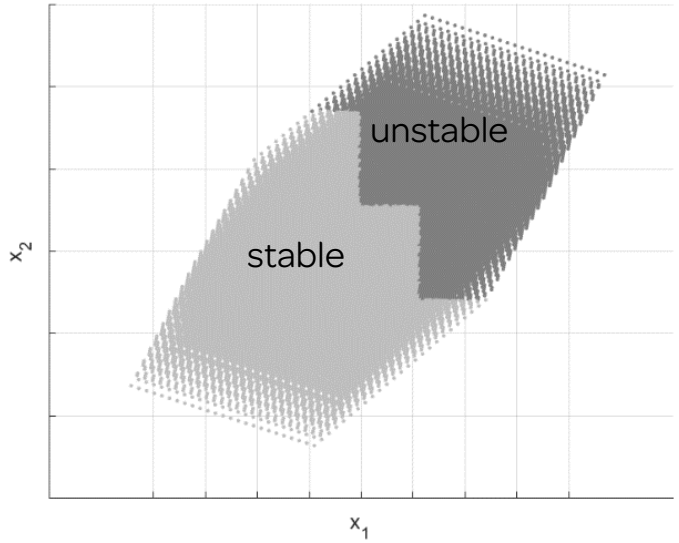
Thams et al IREP 2017,
Halilbasic et al PSCC 2018

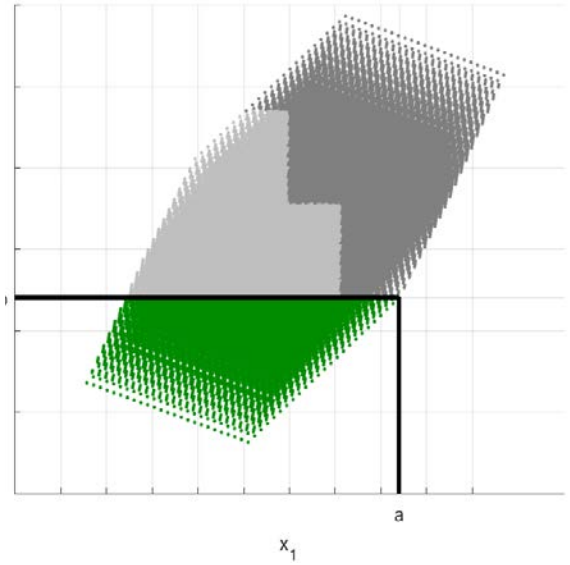
Data-driven security-constrained OPF

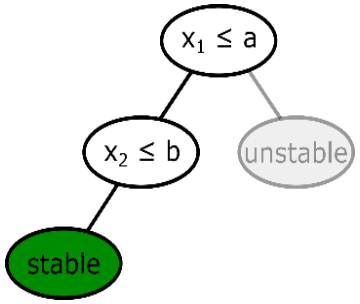




Partitioning the secure operating region

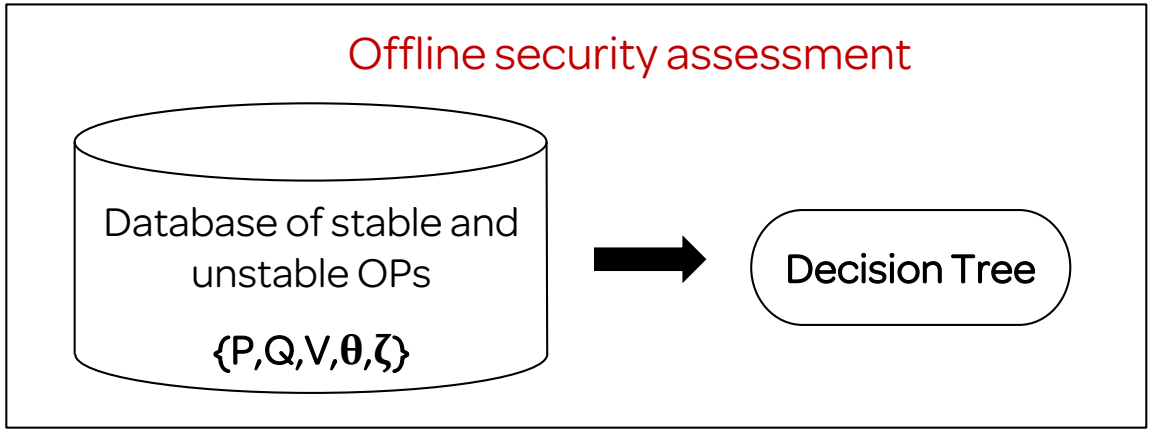


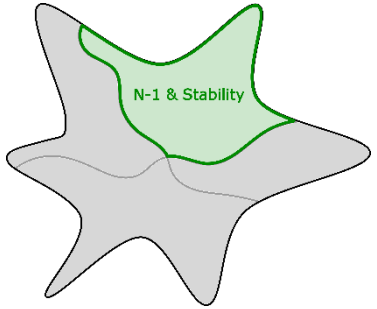




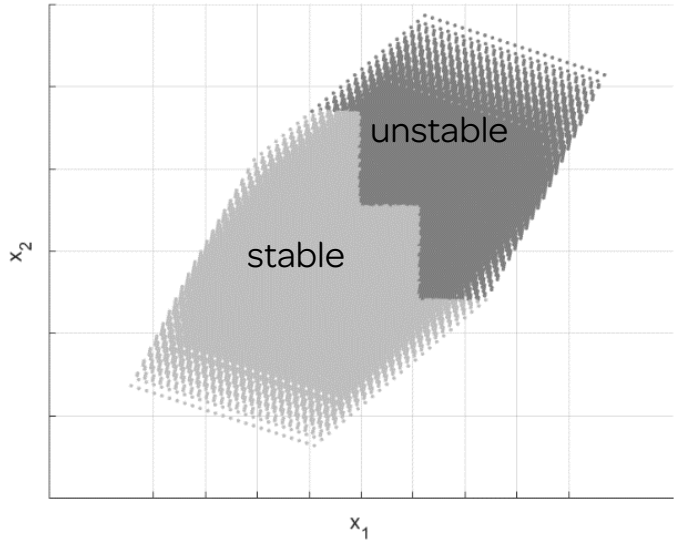
Thams et al IREP 2017,
Halilbasic et al PSCC 2018

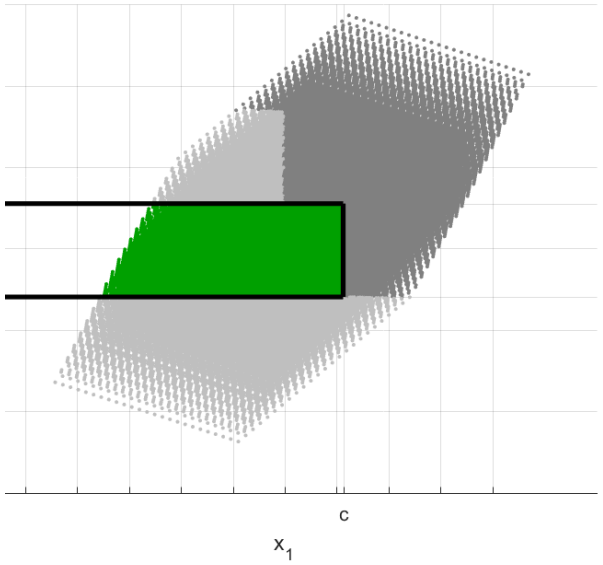
Data-driven security-constrained OPF

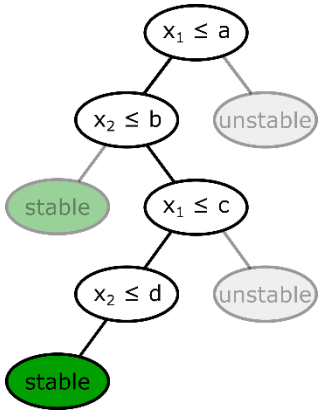




Partitioning the secure operating region

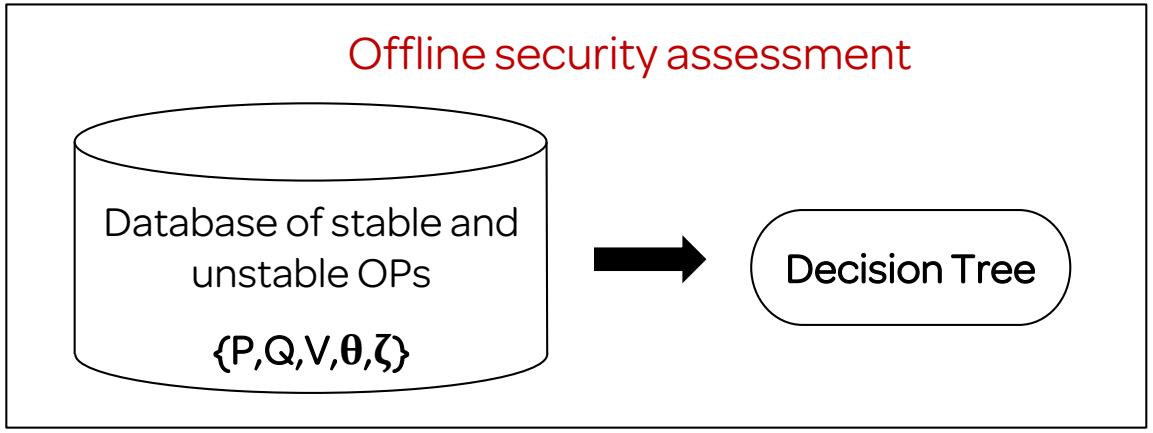


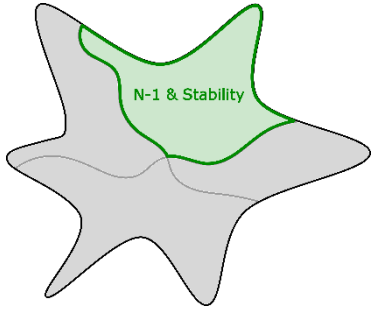




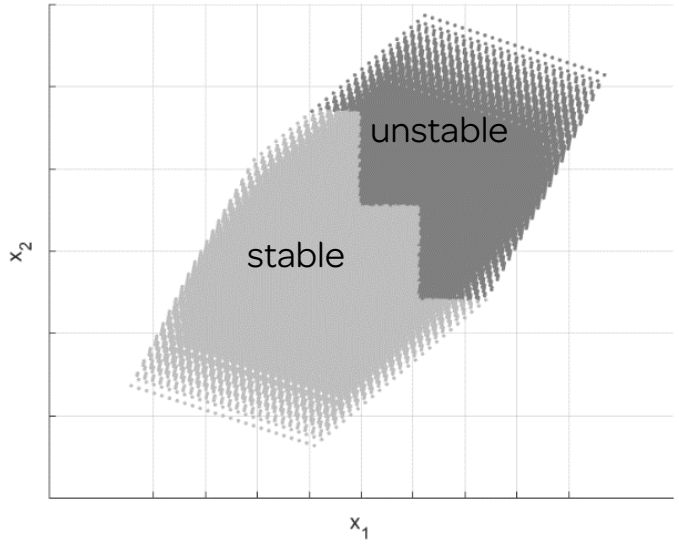
Thams et al IREP 2017,
Halilbasic et al PSCC 2018

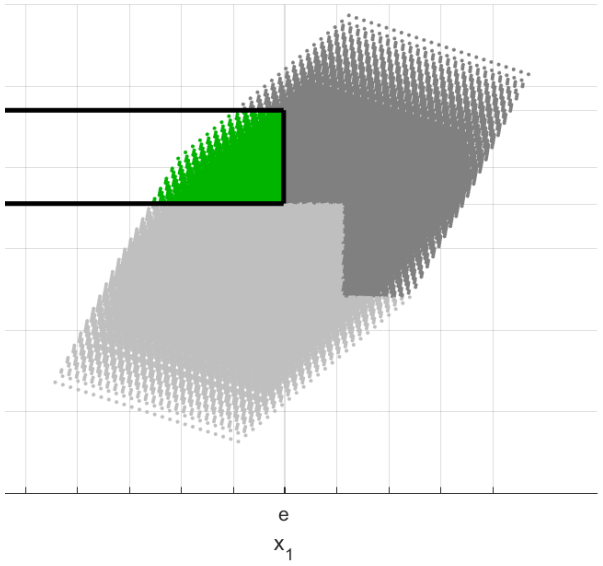
Data-driven security-constrained OPF

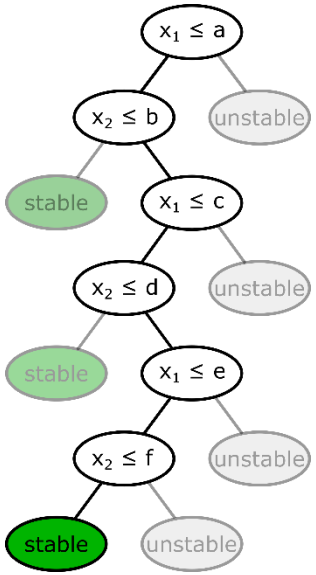




Partitioning the secure operating region







Thams et al IREP 2017,
Halilbasic et al PSCC 2018

Data-driven security-constrained OPF

Offline security assessment

Database of stable and unstable OPs

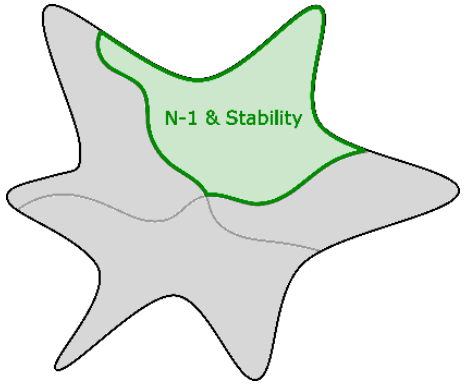
{P,Q,V,θ,ζ}

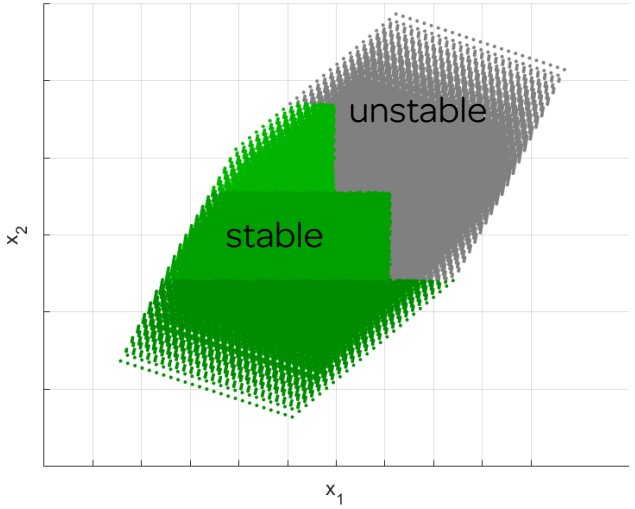
→ Decision Tree

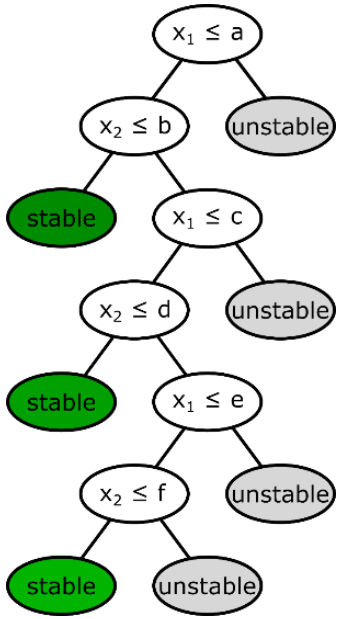
Optimization

Integer Programming to incorporate partitions (DT)

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



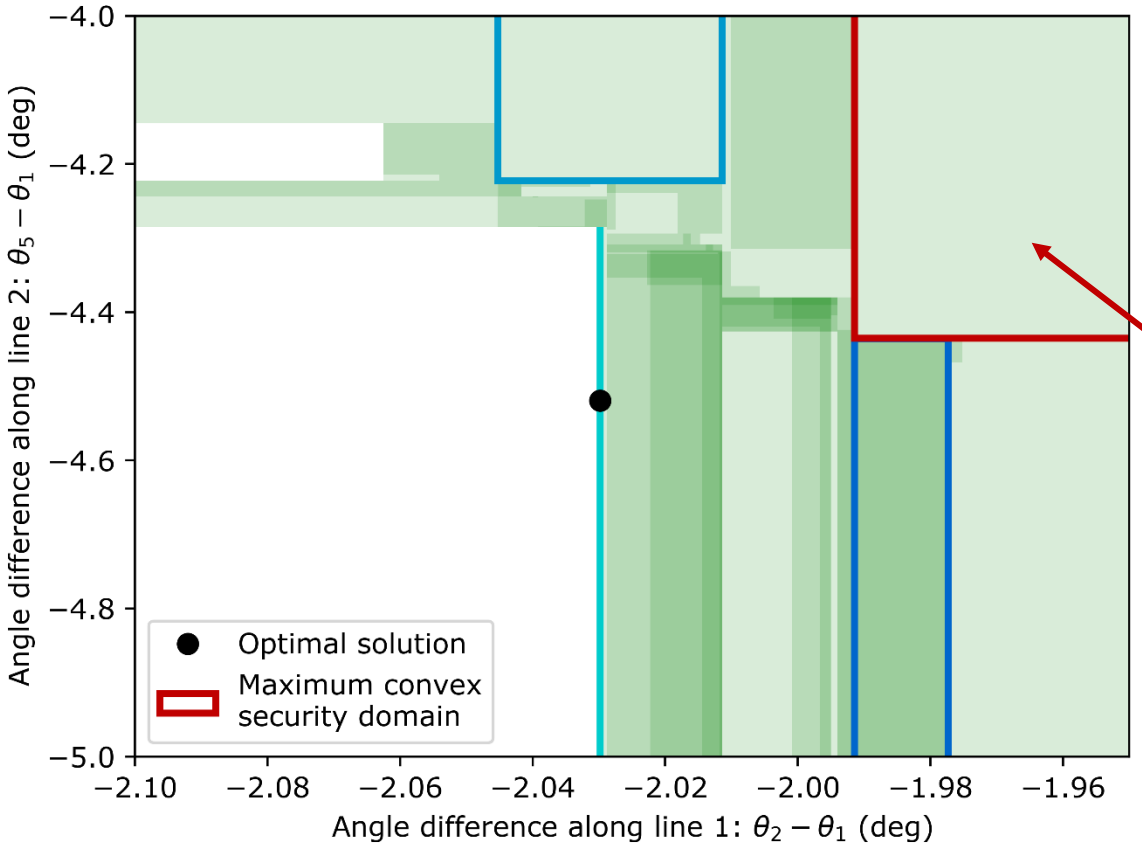


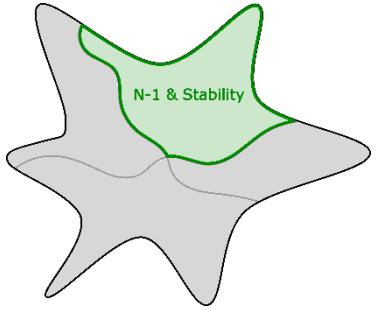


- Each leaf is a convex region
- FBMC corresponds to the leaf that maps the largest convex region

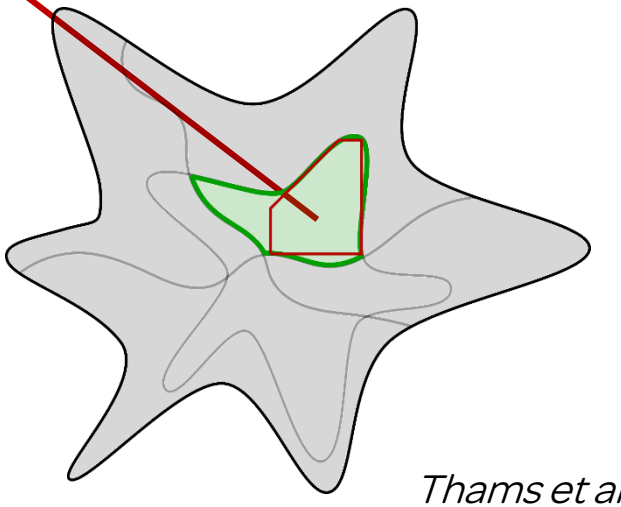
Thams et al IREP 2017, Halilbasic et al PSCC 2018

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



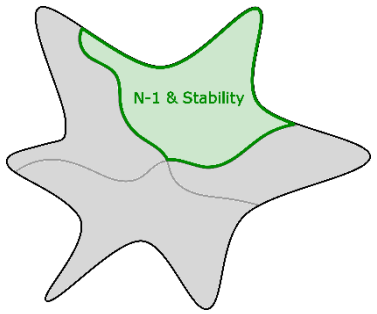


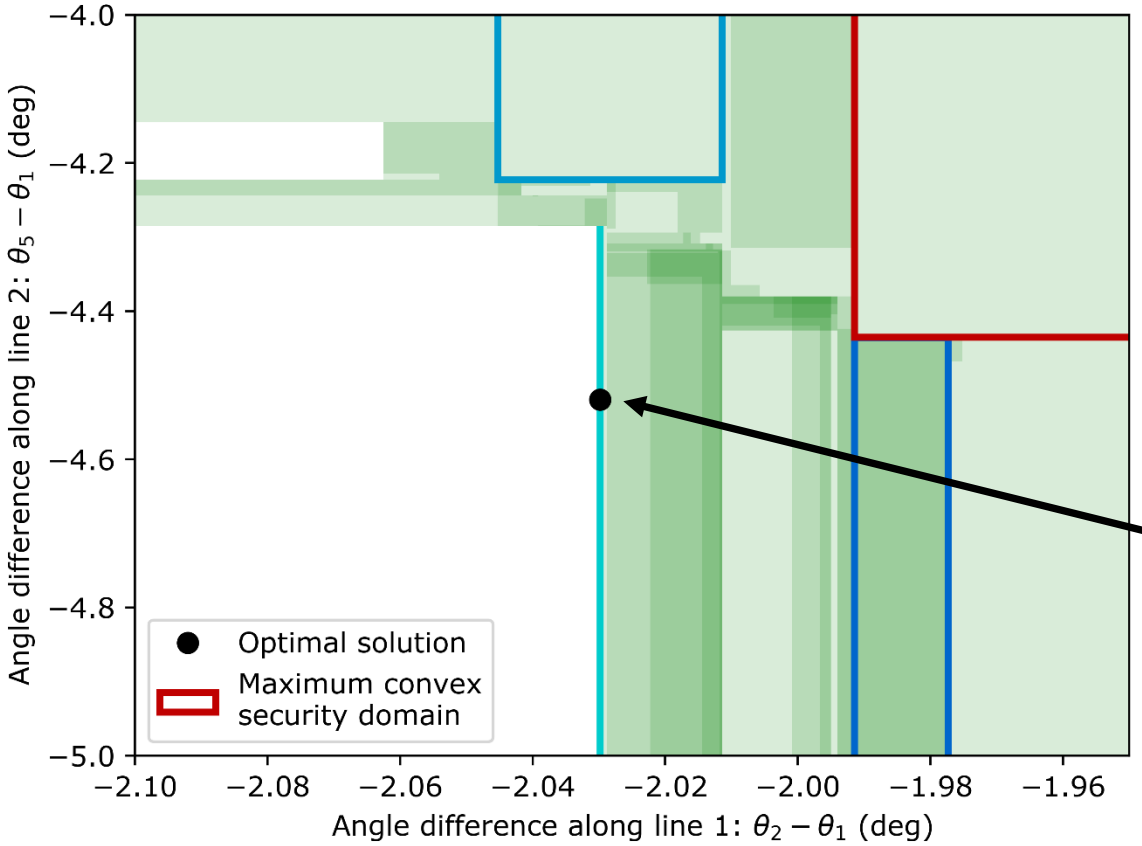
Largest convex region covers ~78%



Thams et al IREP 2017, Halilbasic et al PSCC 2018

MIP + convex AC-OPF approximation finds better solutions than nonconvex problem!





Optimum located at boundary of considered security region

Thams et al IREP 2017, Halilb et al PSCC 2018

Takeaway #7

Decision Trees can convert to a MILP and help capture optimization constraints that were impossible to capture before. E.g. small-signal stability constraints can now be accurately captured in an OPF

Neural Network Verification

Why is neural network verification important?

- Neural networks have been shown to be extremely fast
  - Assess if an operating point is safe or unsafe 100x-300x faster (combination of different security criteria)
  - Application in optimization: find the optimal point >100x faster

- However, neural networks will never be applied in critical (power system) operations, if there are no guarantees about how they behave

- Until recently, the only way to assess the output of the neural networks was to **individually test** each input of interest and pass it through the neural network
  - Accuracy was assessed on discrete samples
  - **Challenge #1:** No way to guarantee what the output is for a **continuous** range of inputs

Evaluating Accuracy: Test Database

Challenge #2:

The test database determines the performance of the neural network.

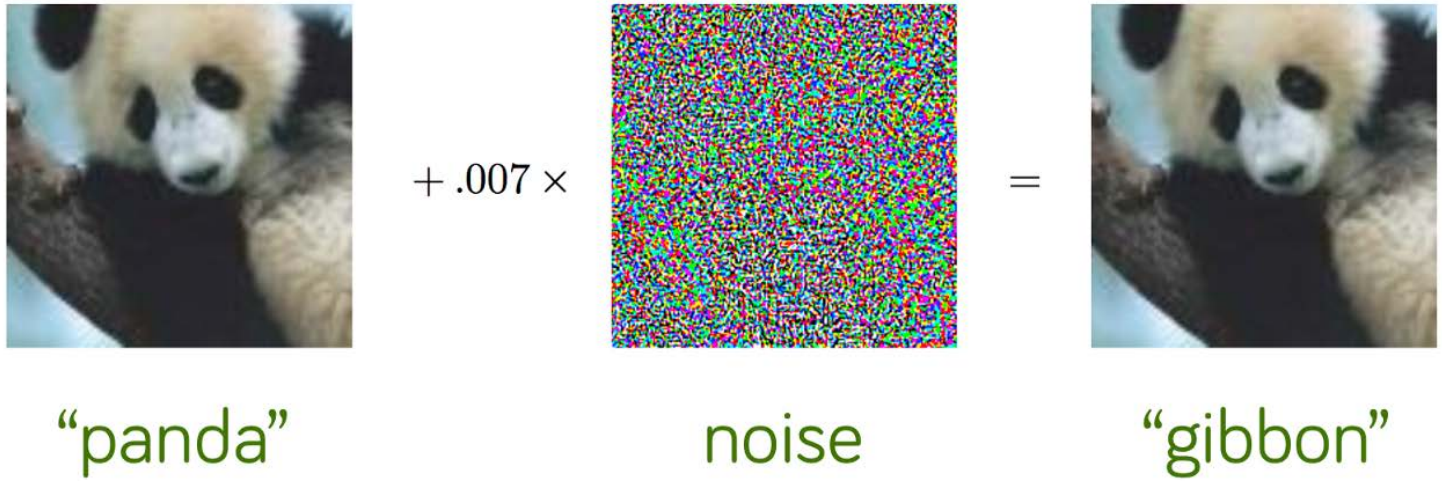
Neural network verification does not depend on a test database:

a. Performance metrics are not based on statistics!

b. Rigorous performance guarantees = certificates

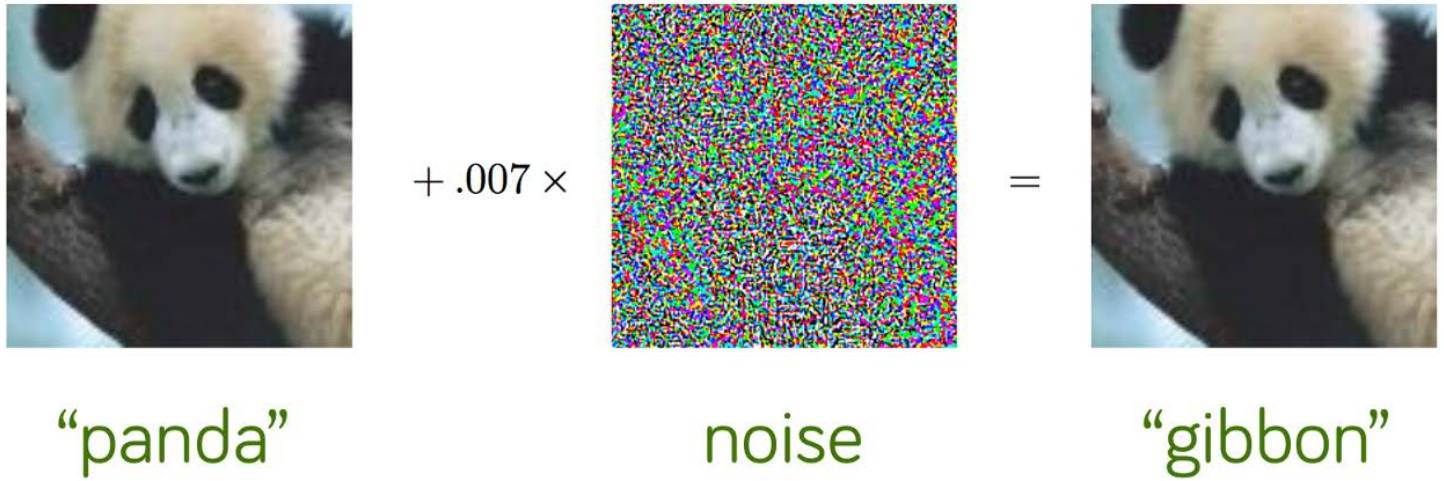
Adversarial examples

- Adversarial example: small perturbations lead to a false prediction



Adversarial examples

- Adversarial example: small perturbations lead to a false prediction



- **Challenge #3: No way to systematically identify adversarial examples**

Neural Network Verification: HOW?

1. Convert the neural network to a **set of linear equations with binaries**
   - The Neural Network can be included in mixed-integer linear program

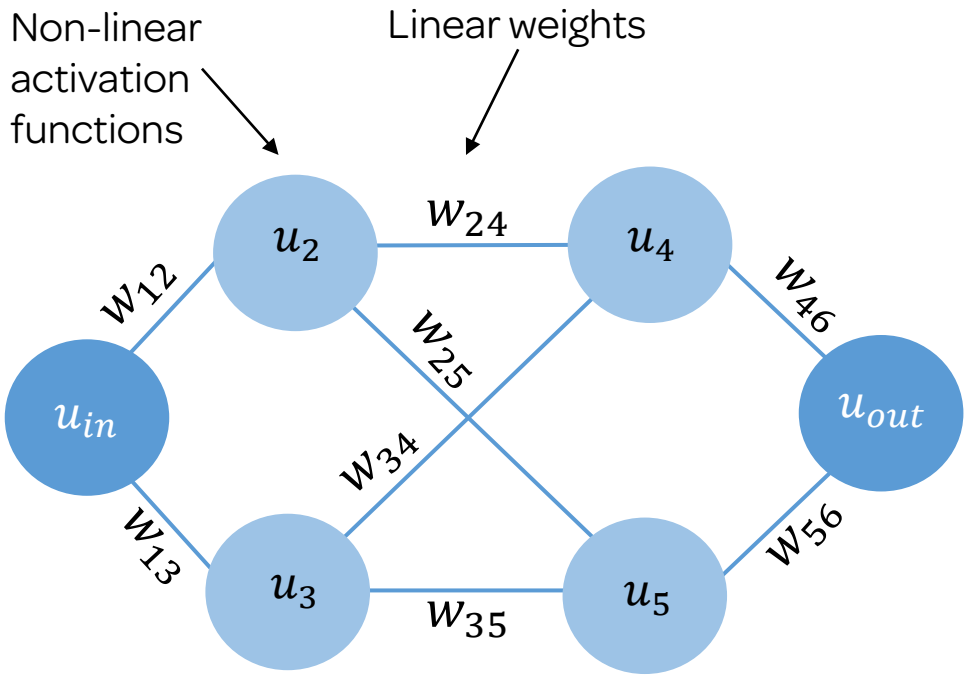
2. Formulate an optimization problem (MILP)

3. Solve the MILP to zero duality gap (find the global optimal) → certificate for the behavior for neural network

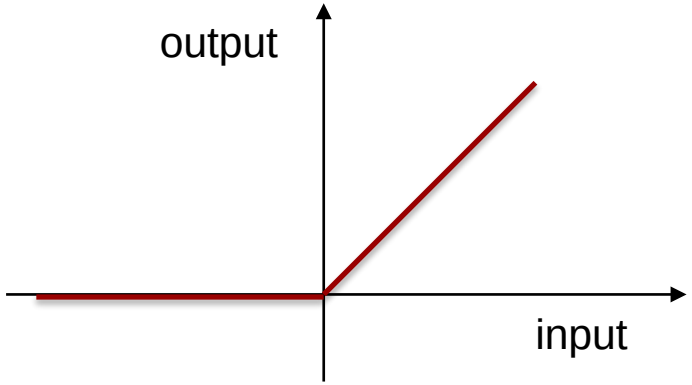
4. Assess if the neural network output complies with the ground truth

- Two types of optimization problem:
  a. Certify that in a region around a given input x_{ref} the neural network maintains the same classification → guarantee that all input points (continuous range) in the neighborhood will be classified the same
  b. Find the minimum distance from x_{ref} that the classification changes (possibility of adversarial examples)

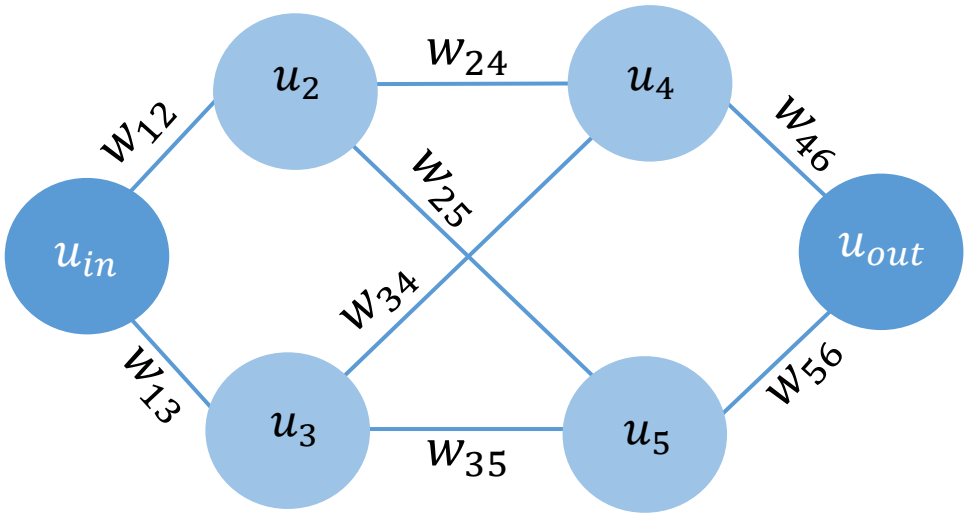
From Neural Networks to Mixed-Integer Linear Programming



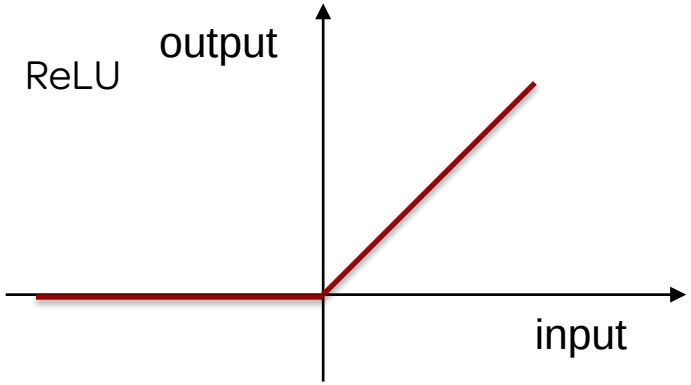
- Most usual activation function: ReLU
- ReLU: Rectifier Linear Unit



From Neural Networks to Mixed-Integer Linear Programming

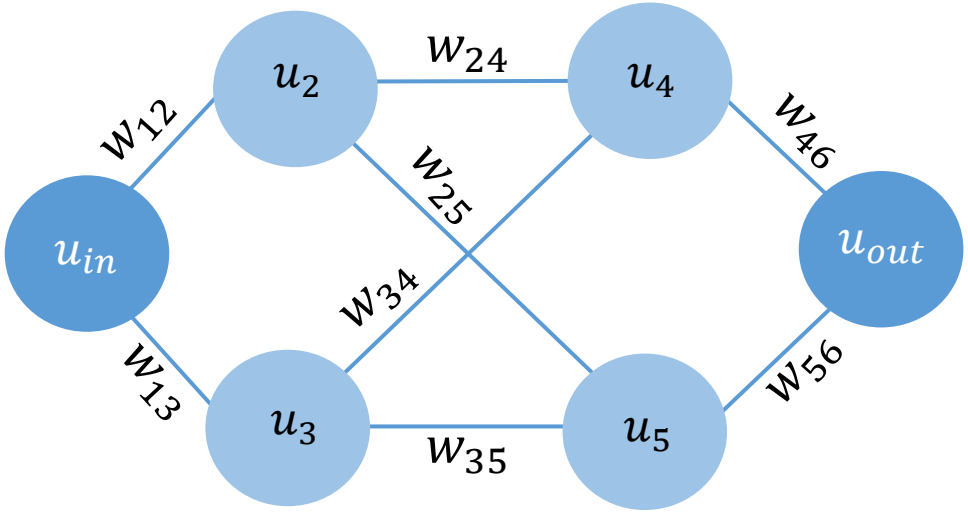


- Linear weights
- On every node: a non-linear activation function
  - ReLU: $\boldsymbol{u_j = \max(0, w_{ij}x_i + b_i)}$
- But ReLU can be transformed to a piecewise linear function with binaries



MILP

From Neural Networks to Mixed-Integer Linear Programming



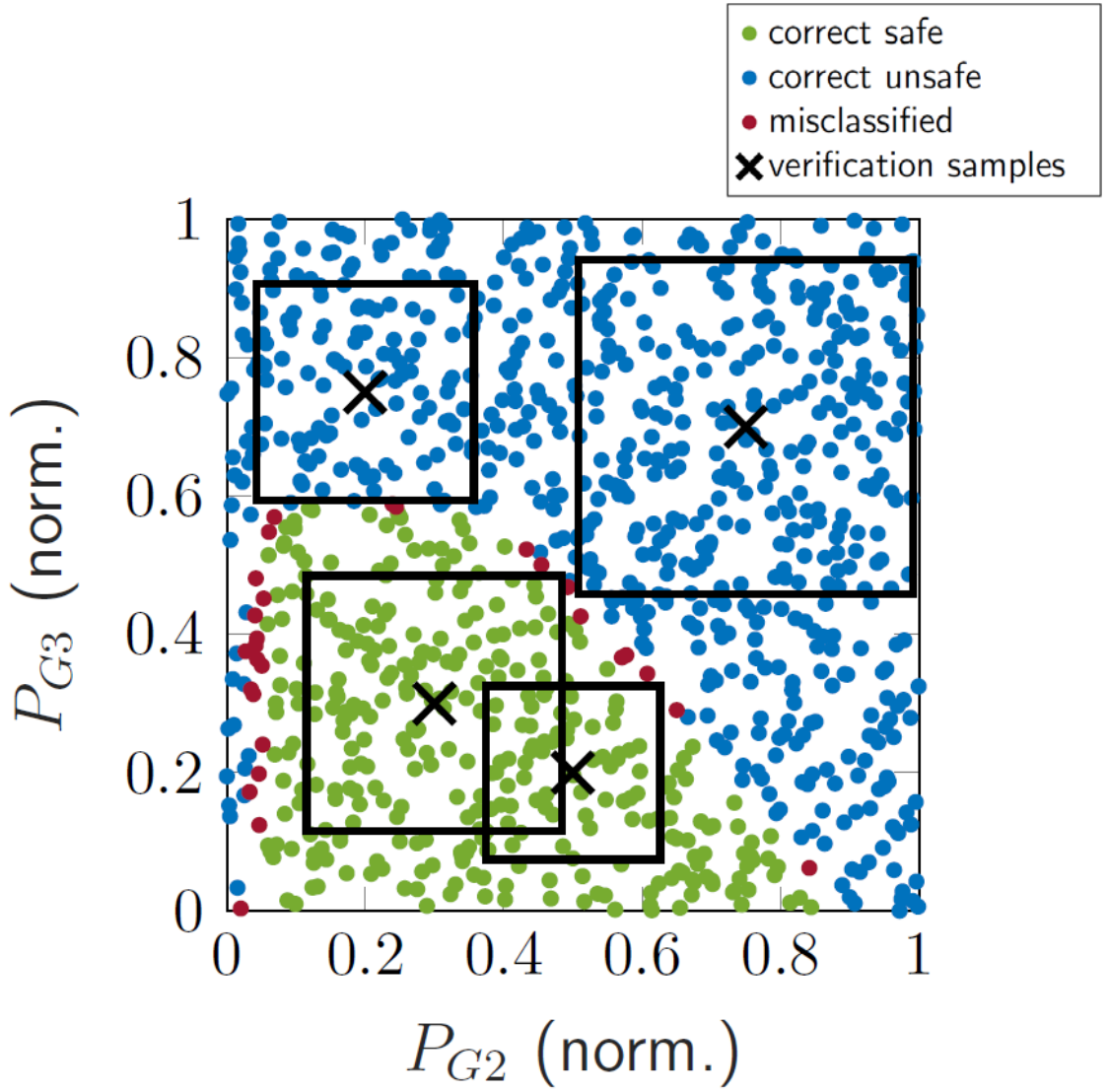
- Output
  - For now: binary classification
  - Security assessment for power systems
  - Output vector y with two elements:
    - $y_1 \geq y_2$: safe
    - $y_2 \geq y_1$: unsafe

Certify the output for a continuous range of inputs

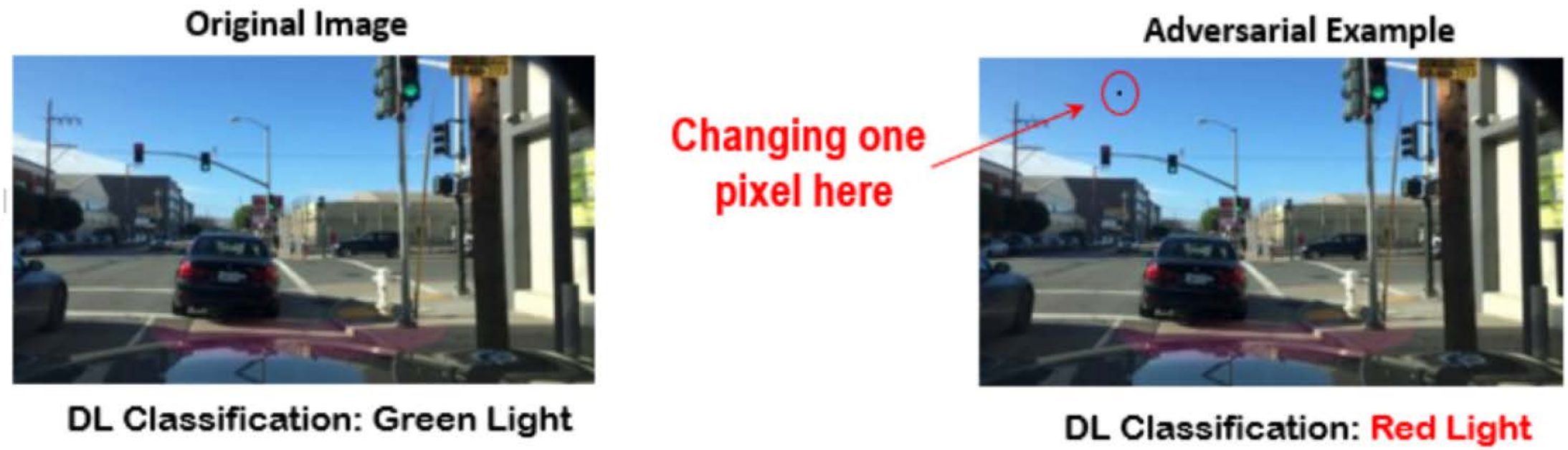
- We assume a given input x_{ref} with classification y: $y_1 > y_2$

1. For distance ϵ evaluate if input x exists with different classification y_2

$$\max_{x,y} \quad \mathbf{y_2} - \mathbf{y_1}$$
$$\text{s.t.} \quad y = NN(x)$$
$$|\mathbf{x} - \mathbf{x}_{\text{ref}}|_\infty \leq \epsilon$$



Adversarial examples in safety-critical systems



source: Wu et al. A game-based approximate verification of deep neural networks with provable guarantees. arXiv:1807.03571.

- Adversarial examples exist in many (deep) learning applications
- Major barrier for adoption of machine learning techniques in safety-critical systems!

Systematically identify adversarial examples

- We assume a given input x_{ref} with classification y: $y_1 > y_2$

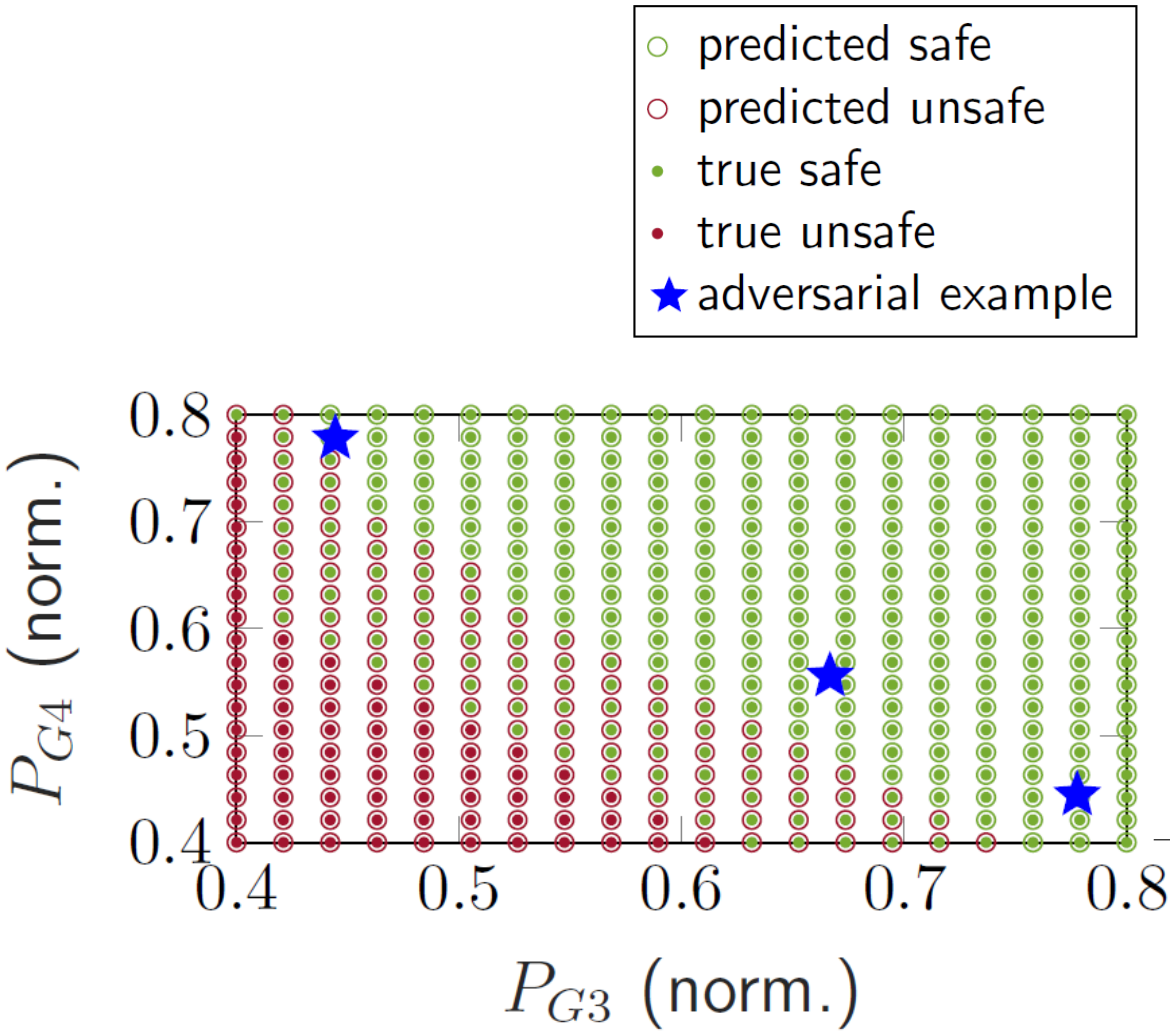
2. Minimize distance ϵ from $\mathbf{x}_{\text{ref}}$ to input $\mathbf{x}$ with classification y_2

$$\min_{\mathbf{x},\mathbf{y},\epsilon} \quad \epsilon$$

$$\text{s.t.} \quad \mathbf{y} = NN(\mathbf{x})$$

$$|\mathbf{x} - \mathbf{x}_{\text{ref}}|_\infty \leq \epsilon$$

$$y_2 \geq y_1$$



Challenges

- **Tractability** for large neural networks
  - Up to now, we have verified NNs with 5 layers and 50 nodes at each layer (NN used for the 162-bus system)
  - We require weight sparsification, bound tightening, and ReLU pruning (remove binary variables) to maintain tractability

- **Connect verification with ground truth assessment**
  - Currently, we can first certify the neural network output, and we should then assess if this output is correct (i.e. that the NN can be trusted in real operation)
  - Currently working on a verification procedure that will be integrated in the training of the neural network → training results in a certificate of performance (no more statistics!)

- **Retraining** is necessary to avoid adversarial examples
  - The **quality of the training database is crucial** for good performance!

Takeaway #8

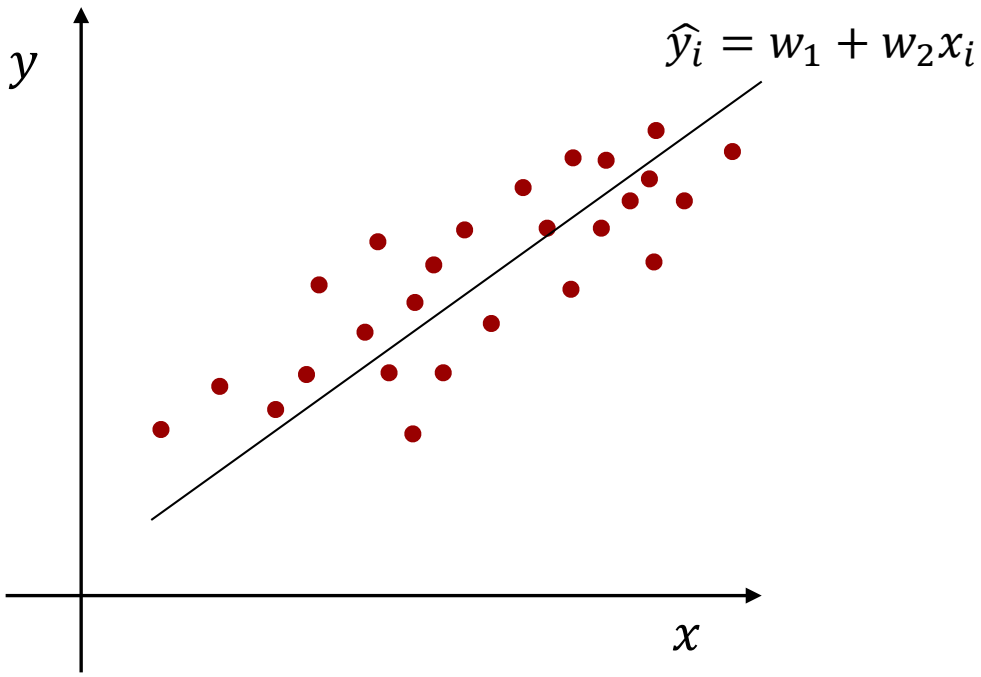
Neural network verification removes the barrier for neural network applications in safety-critical operations. High potential for power system applications.

Physics-Informed Neural Networks for Power Systems

Loss function: Estimate best w_1, w_2 to fit the training data

y_i: actual/correct value

$\hat{y}_i$: estimated value



$$\min_{w_1, w_2} \quad \|y_i - \hat{y}_i\|$$

s.t.

$$\hat{y}_i = w_1 + w_2 x_i \quad \forall i$$

Rewrite:

$$\min_{w_1, w_2} \quad \|y_i - (w_1 + w_2 x_i)\| \quad \forall i$$

Traditional training of neural networks required no information about the underlying physical model. Just data!

Physics Informed Neural Networks

- Automatic differentiation: derivatives of the neural network output can be computed during the training procedure
- A differential-algebraic model of a physical system can be included in the neural network training*
- Neural networks can now exploit knowledge of the actual physical system
- Machine learning platforms such as Tensorflow enable these capabilities

*M. Raissi, P. Perdikaris, and G. Karniadakis, Physics-Informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations", Journal of Computational Physics, vol.378, pp. 686-707, 2019

Physics-Informed Neural Networks for Power Systems

Loss function

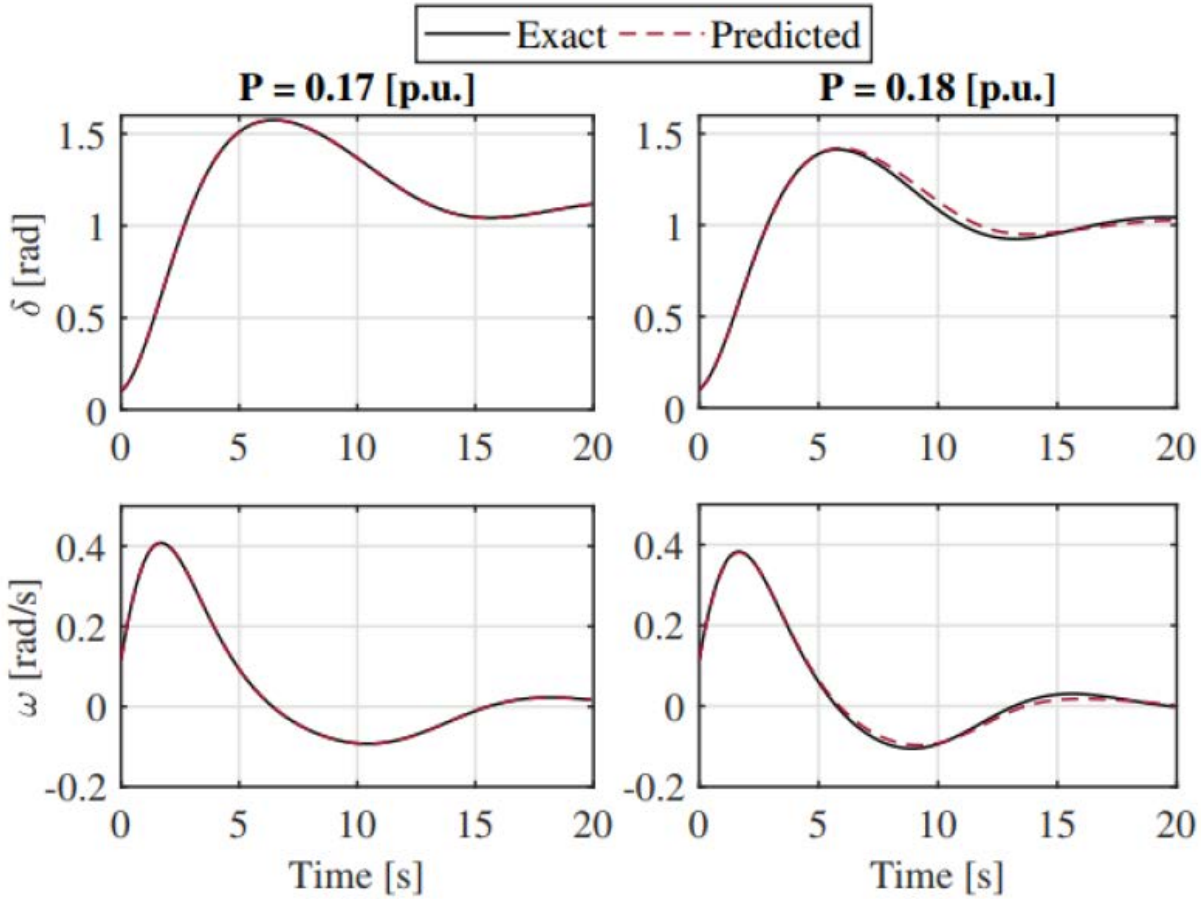
$$\min_{\mathbf{W},\mathbf{b}} \quad \frac{1}{|N_\delta|}\sum_{i\in N_\delta}|\hat{\delta}-\delta^i|^2+\frac{1}{|N_f|}\sum_{i\in N_f}|f(\hat{\delta})|^2 \tag{6a}$$

$$s.t. \quad \hat{\delta}=NN(t,P_m,\mathbf{W},\mathbf{b}) \tag{6b}$$

$$\dot{\hat{\delta}}=\frac{\partial\hat{\delta}}{\partial t}, \qquad \ddot{\hat{\delta}}=\frac{\partial\dot{\hat{\delta}}}{\partial t} \tag{6c}$$

$$f(\hat{\delta})=M\ddot{\hat{\delta}}+D\dot{\hat{\delta}}+A\sin\hat{\delta}-P_m \tag{6d}$$

Swing equation

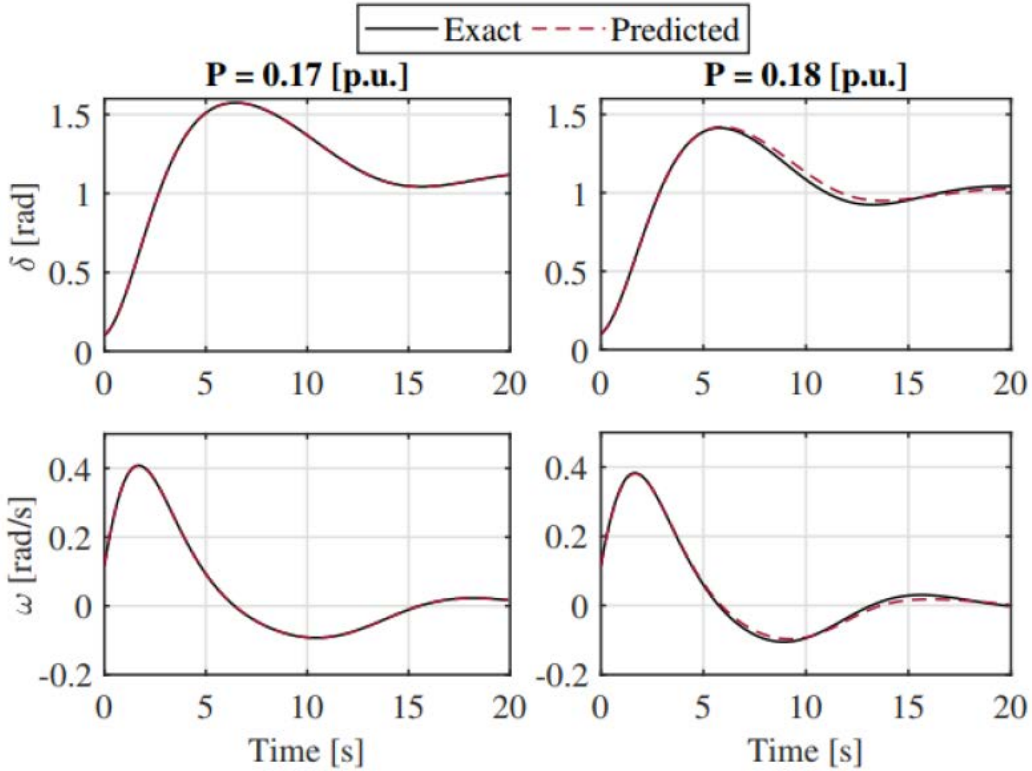


Code is available on GitHub: https://github.com/gmisy/Physics-Informed-Neural-Networks-for-Power-Systems/

G. S. Misyris, A. Venzke, S. Chatzivasileiadis, Physics-Informed Neural Networks for Power Systems. Accepted at IEEE PES GM 2020. https://arxiv.org/pdf/1911.03737.pdf

Physics-Informed Neural Networks for Power Systems

- Physics-Informed Neural Networks **can potentially replace** solvers for systems of differential-algebraic equations
- In our example: PINN 87 times faster than ODE solver
- Can directly estimate the rotor angle at **any** time instant



Code is available on GitHub: https://github.com/gmisy/Physics-Informed-Neural-Networks-for-Power-Systems/

G. S. Misyris, A. Venzke, S. Chatzivasileiadis, Physics-Informed Neural Networks for Power Systems. Accepted at IEEE PES GM 2020. https://arxiv.org/pdf/1911.03737.pdf

Takeaway #9

Physics-informed neural networks exploit the underlying physics in the training procedure. They are still at a very initial development stage; they have the potential to accelerate by 100-1000 times functions of conventional solvers

Takeaway wrap-up

1. Before applying machine learning, develop a solid motivation about *why* ML would work better.
2. The quality of your test database is crucial
3. Neural network and Decision Trees for classification: you need a *balanced* training database
4. Accuracy is not a sufficient performance metric
5. Get familiar with the tips and tricks to improve neural network training (e.g. 1-hot encoding)
6. Sampling beyond statistics: creating high-quality training databases is extremely complex; exploit the physical laws. **Open research topic**
7. Decision trees can convert to a MILP and help capture optimization constraints that were impossible to capture before. **Open research topic**
8. Neural network verification removes the barrier for neural network applications in safety-critical operations. **Open research topic**
9. Physics-informed neural networks exploit the underlying physical laws during training. **Open research topic**

Highlights

- Decision Trees can be converter to a MILP through an exact transformation
  - Capture constraints impossible to capture before (e.g. based on differential equations)

- Sampling beyond statistics
  - Need to exploit physics for creating the training databases -- an open research topic

- Neural network verification
  - A world of new opportunities for practical applications in power systems
  - Certify the behavior of neural networks
  - Systematically identify adversarial examples

- Physics-Informed Neural Networks
  - Exploit the underlying physical model in the neural network training
  - Extremely fast computing times: no need to integrate $t_0 \rightarrow t_1$, can directly estimate $x(t_1)$
  - Potential to replace differential-algebraic solvers for real-time applications?

Thank you!

Spyros Chatzivasileiadis
Associate Professor, PhD
www.chatziva.com
spchatz@elektro.dtu.dk

Deep learning, NIPS Tutorial 2017: https://www.youtube.com/watch?v=YJnddoa8sHk

Sampling beyond Statistics: Efficient Database Creation

F. Thams, A. Venzke, R. Eriksson, and S. Chatzivasileiadis, "Efficient database generation for data-driven security assessment of power systems". IEEE Trans. Power Systems, vol. 35, no. 1, pp. 30-41, Jan. 2020. https://www.arxiv.org/abs/1806.0107.pdf

A. Venzke, D.K. Molzahn, S. Chatzivasileiadis, Efficient Creation of Datasets for Data-Driven Power System Applications. PSCC 2020. https://arxiv.org/pdf/1910.01794.pdf

Neural Network Verification

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

V. Tjeng, K. Y. Xiao, and R. Tedrake, "Evaluating robustness of neural networks with mixed integer programming," in International Conference on Learning Representations (ICLR 2019), 2019. [Online]. Available: https://openreview.net/forum?id=HyGIdiRqtm

A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in International Conference on Learning Representations (ICLR 2018), 2018. [Online]. Available: https://openreview.net/forum?id=rJzIBfZAb

C. Liu, T. Arnon, C. Lazarus, C. Barrett, M. J. Kochenderfer, Algorithms for Verifying Deep Neural Networks, 2019. Available: https://arxiv.org/ftp/arxiv/papers/1903/1903.06758.pdf

Physics-Informed Neural Networks

M. Raissi, P. Perdikaris, and G. Karniadakis, Physics-Informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations", Journal of Computational Physics, vol.378, pp. 686-707, 2019 https://arxiv.org/abs/1711.10561

G. S. Misyris, A. Venzke, S. Chatzivasileiadis, Physics-Informed Neural Networks for Power Systems. IEEE PES GM 2020. https://arxiv.org/pdf/1911.03737.pdf

F. Fioretto and T. W. K. Mak and P. Van Hentenryck , Predicting AC Optimal Power Flows: Combining Deep Learning and Lagrangian Dual Methods, 2019. https://arxiv.org/pdf/1909.10461.pdf

Machine Learning for Power Systems

L. Halilbašić, F. Thams, A. Venzke, S. Chatzivasileiadis, and P. Pinson, "Data-driven security-constrained AC-OPF for operations and markets," in *2018 Power Systems Computation Conference (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]

L. Wehenkel, M. Pavella, E. Euxibie, and B. Heilbronn, "Decision treebased transient stability method a case study,"IEEE Transactions onPower Systems, vol. 9, no. 1, pp. 459–469, 1994.

N. Hatziargyriou, G. Contaxis, and N. Sideris, "A decision tree method for on-line steady state security assessment," IEEE Transactions onPower Systems, vol. 9, no. 2, pp. 1052–1061, 1994.

I. Genc, R. Diao, V. Vittal, S. Kolluri, and S. Mandal, "Decision tree-based preventive and corrective control applications for dynamic securityenhancement in power systems,"IEEE Transactions on Power Systems,vol. 25, no. 3, pp. 1611–1619, 2010.

C. Hidalgo-Arteaga, F. Hancharou, F. Thams, S. Chatzivasileiadis, Deep Learning for Power System Security Assessment. In *IEEE Powertech 2019*, Milan, Italy, pages 1-6, June 2019. [.pdf]