2023 Los Alamos Grid Science
Winter School and Conference

Machine Learning for Power Systems: Is it time to trust it?

Spyros Chatzivasileiadis
Associate Professor
Head of Section Power Systems

DTU Department of Wind and Energy Systems

Working for a sustainable future

Management

Head of Department
Peter Hauge Madsen

Deputy Head of Department
Peter Hjuler Jensen

Administration

Head of Administration
Søren Knudsen

Wind Energy Materials & Components (WMC)

Head of Division
Ignacio Martí

Wind Turbine Design (WTD)

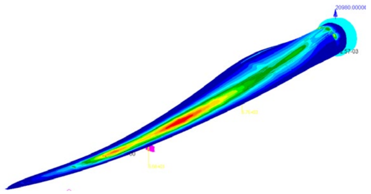
Head of Division
Kenneth Thomsen

Wind Energy Systems (WES)

Head of Division
Cathy Suo

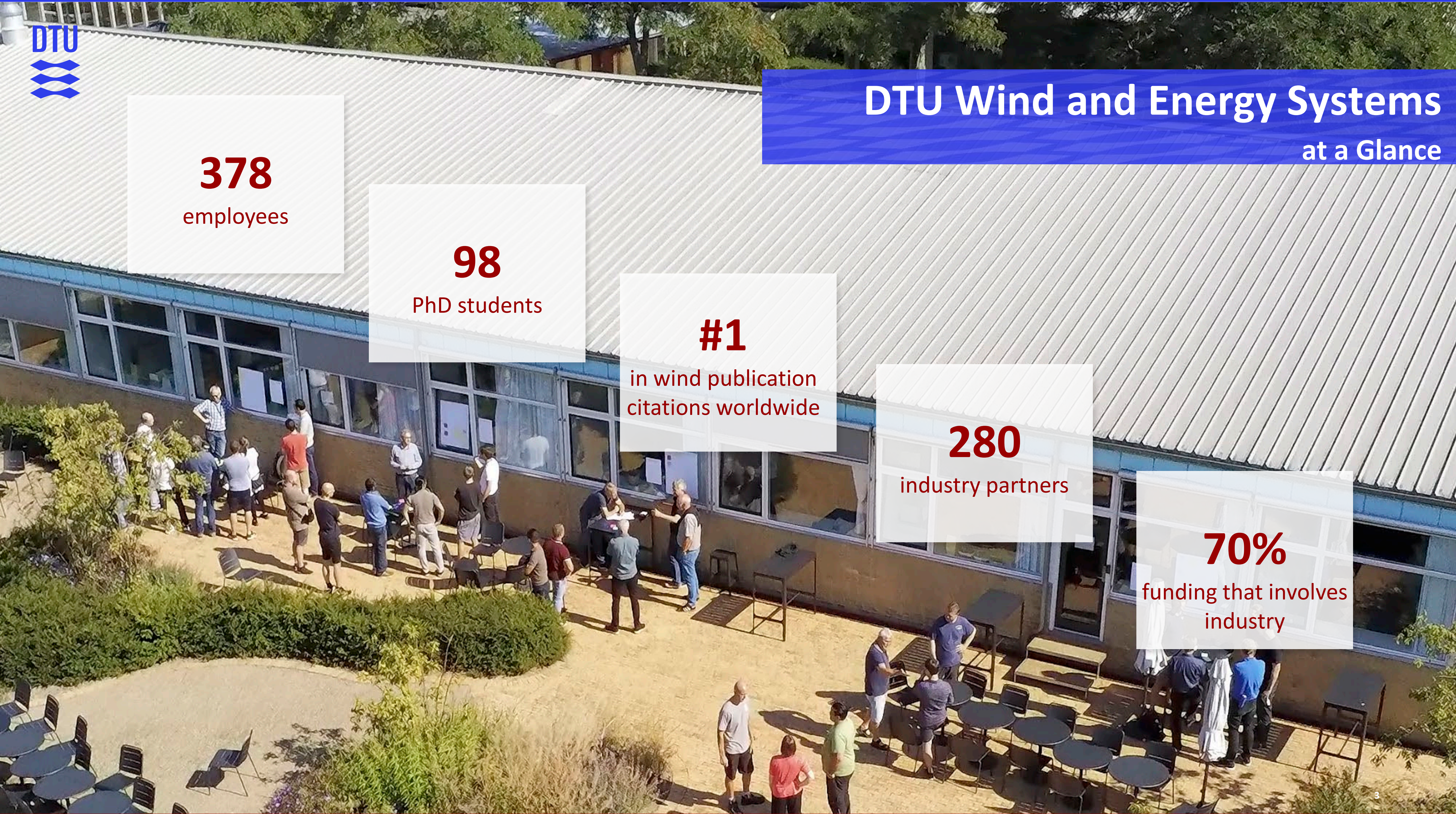
Power and Energy Systems (PES)

Head of Division
Jacob Østergaard



DTU Wind and Energy Systems

at a Glance



378
employees

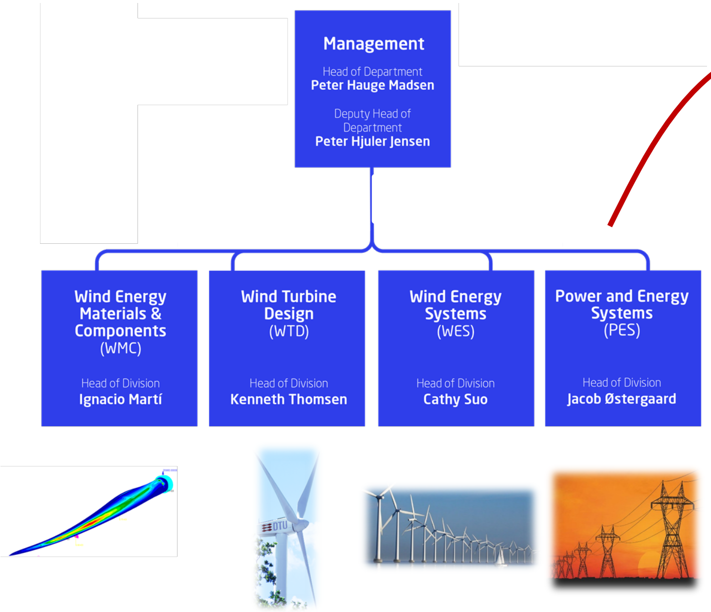
98
PhD students

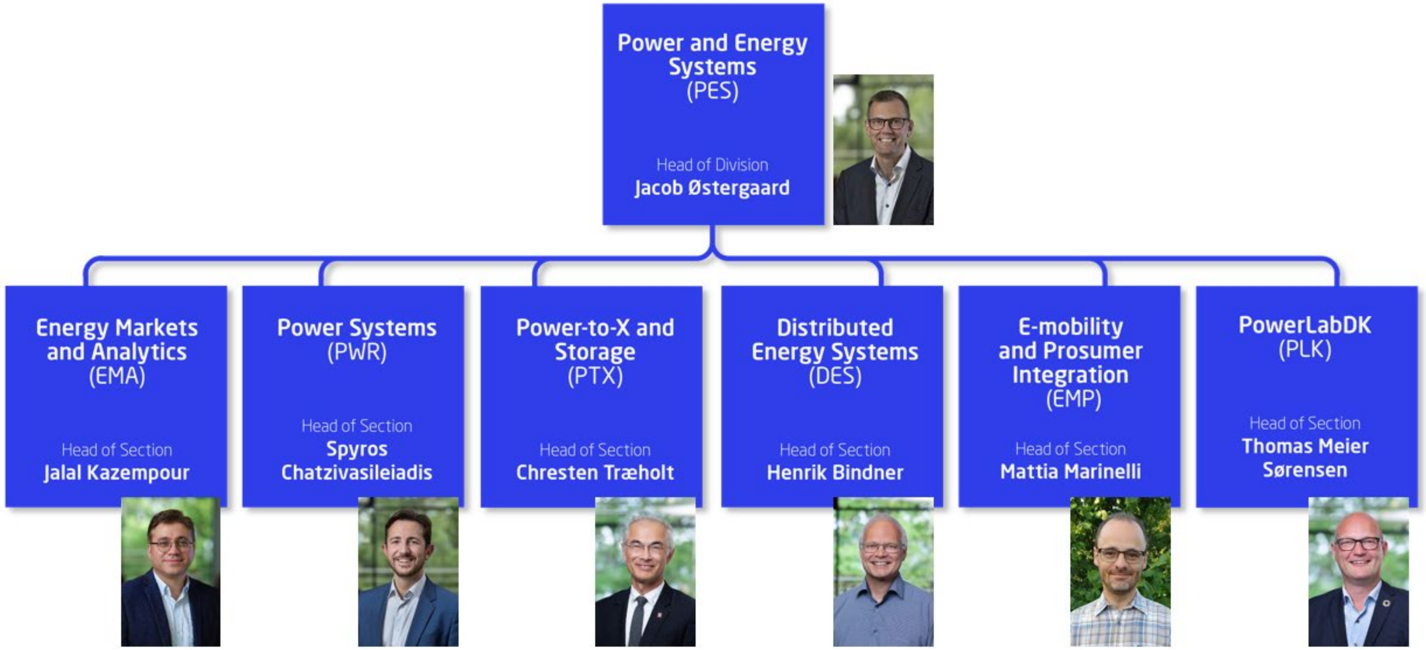
#1
in wind publication citations worldwide

280
industry partners

70%
funding that involves industry

DTU Department of Wind and Energy Systems
Working for a sustainable future





~100 people working on power systems

Electric Power Systems

PWR Section: 30+3 members; 20 nationalities


Spyros Chatzivasileiadis


Guangya Yang


Hjörtur Jóhannsson


Nikos Cutululis


Tonny W. Rasmussen


Oscar Saborio-Romano


Vassilis Kekatos
(Visiting Professor, Virginiatech, US)


Mirza Nuhic


Lars Herre


Sam Chevalier


Ilaria Sorrenti


Jochen Stiasny


Konrad Sundsgaard


Jose A. L. Vilaplana


Ana Turk


Daniel Müller


Agnes Nakiganda


Gabriel M.G. Guerreiro


Kaio Vinicius Vilerá


Rahul Nellikkath


Ayşegül Kahraman


Brynjar Sævarsson

Nikita Taranin


Karoline Reich


Amir Arasteh


Sulav Ghimire


Sujay Ghosh

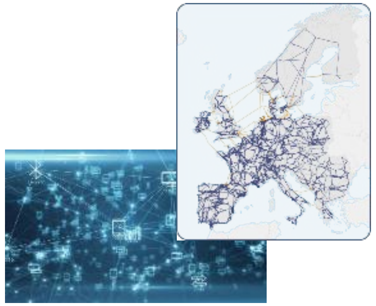

Ilgiz Murzakhanov


Alessandra Follo

Guests

Yan Xu


Simon Stock

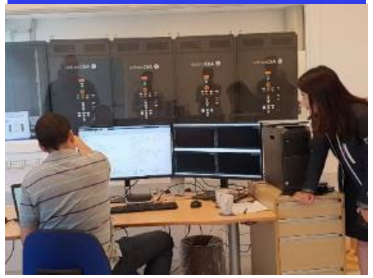


PowerLabDK

Energy System Simulation Lab

Digital Energy Lab

AC/DC Wind Power Lab



PWR: Advanced Tools to Avoid Blackouts

Next-Generation Scientific Computing

- **Physics-Informed** and **Trustworthy AI**
- **Quantum** Computing
- Energy **Data Spaces**

Bornholm "Living Lab"

- Danish Island. "Living Lab" of 40'000 people
- **Demonstrations** for Energy Islands, Energy Data Spaces, Smart Control of Converters

Cyberphysical Systems

- **Digital Twins**
- Hardware and Software in the Loop
- **Open-Source Models** of the Nordic and European Systems

Extreme Converter-Based Power Systems

- North Sea **Energy Islands**
- Baltic Energy Island
- **HVDC Grids**
- Interoperability and Standards

Funding

erc European Research Council
INNOVATIVE TRAINING NETWORKS
EUDP Energiteknologisk udvikling og demonstration
TotalEnergies
Horizon Europe
Innovation Fund Denmark

North Sea Energy Island

Artist's impression

This work would not have been possible without the hard work of several people! Many thanks to…

Andreas Venzke

Rahul Nellikkath

Sam Chevalier

Lejla Halilbasic

Elea Prat

Ilgiz Murzakhanov

Simon Stock (TU Hamburg)

Spyros Chatzivasileiadis

Agnes Nakiganda

Georgios Misyris

Florian Thams

Jochen Stiasny

Brynjar Sævarsson

And to our collaborators:

Dan Molzahn, GeorgiaTech

Steven Low, Caltech
Guannan Qu, Caltech (now at CMU)

Alyssa Kody, Argonne National Lab
Panagiotis Papadopoulos, Robert Hamilton, Tabia Ahmad, Univ. Strathclyde
Vladimir Dvorkin, MIT

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

1. **Extremely fast → can assess 100x-1'000x more of critical scenarios**
   - computation within only a **few milliseconds** (100x – 1000x faster than conventional methods)
     - Predict fast and act faster → drastically increase power system resilience

2. Can handle **very complex systems** and **infer** from incomplete data
   - Excellent potential to create accurate **surrogate models**
     - Accelerate simulations; and offer good approximations of previously intractable systems

ML Proxies

Extremely fast, and hopefully accurate

But: Would an Operator ever trust AI in the Control Room?

This talk: Two Challenges

- **Challenge #1:** Machine Learning is extremely dependent on high-quality data. What about all the physical models we have developed over the past 100 years?

- **Challenge #2:** Has the Neural Network been trained to generalize well? Can we trust it?

Abbreviations I will use:

- ML: Machine Learning
- NN: Neural Network

Challenge #1: ML extremely dependent on high-quality data

Facts

1. **All data are not the same**
   Example: Assume we train a NN to determine if a system is stable → Training data close to the stability boundary contain much more information than training data far away from it.

Consequence

Statistical sampling is not enough

Challenge #1: ML extremely dependent on high-quality data

Facts	Consequence
1. All data are not the same Example: Assume we train a NN to determine if a system is stable → Training data close to the stability boundary contain much more information than training data far away from it.	Statistical sampling is not enough
2. Training data must follow the same statistical properties as real data Do we have enough historical data about e.g. outages? Is this possible?	1. For power systems: We have so many physical models. Add them! 2. Unbalanced datasets. We cannot trust "Neural Network Accuracy" as a performance metric

	Facts	Consequence
	1. All data are not the same Example: Assume we train a NN to determine if a system is stable → Training data close to the stability boundary contain much more information than training data far away from it.	Statistical sampling is not enough
Challenge #1: ML extremely dependent on high-quality data	**2. Training data must follow the same statistical properties as real data** Do we have enough historical data about e.g. outages? Is this possible?	1. For power systems: We have so many physical models. Add them! 2. Unbalanced Datasets. We cannot trust “Neural Network Accuracy” as a performance metric
Challenge #2: Has NN been trained to generalize well?	**3. NN training is an extremely complex optimization procedure** Prone to overfitting/underfitting	Can we trust it?

Facts

Solution

1. **All data are not the same**
   Example: Assume we train a NN to determine if a system is stable → Training data close to the stability boundary contain much more information than training data far away from it.

Statistic… …ough

Sampling beyond Statistics

Challenge #1:
ML extremely dependent on high-quality data

2. **Training data must follow the same statistical properties as real data**
   Do we have enough historical data about e.g. outages? Is this possible?

1. For power systems: We have so many physical models. Add them!

2. Unbalanced Datasets. We cannot trust "Neural Network Accuracy" as a performance metric

Challenge #2:
Has NN been trained to generalize well?

3. **NN training is an extremely complex optimization procedure**
   Prone to overfitting/underfitting

Can we trust it?

DTU

Facts

Solution

Challenge #1:
ML extremely dependent on high-quality data

1. **All data are not the same**
   Example: Assume we train a NN to determine if a system is stable → Training data close to the stability boundary contain much more information than training data far away from it.

Statistic... ...ough

Sampling beyond Statistics

2. **Training data must follow the same statistical properties as real data**
   Do we have enough historical data about e.g. outages? Is this possible?

1. For ... ...ave so many ph... ...els. Add them!

Physics-Informed NNs

2. Unbalanced Datasets. We cannot trust "Neural Network Accuracy" as a performance metric

Challenge #2:
Has NN been trained to generalize well?

3. **NN training is an extremely complex optimization procedure**
   Prone to overfitting/underfitting

Can we trust it?

Facts

Solution

Challenge #1:
ML extremely dependent on high-quality data

1. **All data are not the same**
Example: Assume we train a NN to determine if a system is stable → Training data close to the stability boundary contain much more information than training data far away from it.

Statistic… …ough

Sampling beyond Statistics

2. **Training data must follow the same statistical properties as real data**
Do we have enough historical data about e.g. outages? Is this possible?

1. For … …ave so many ph… …els. Add them!

Physics-Informed NNs

2. Unbalanced Datasets. W… …trust "Neural Netw… P…

Neural Network Verification

Challenge #2:
Has NN been trained to generalize well?

3. **NN training is an extremely complex optimization procedure**
Prone to overfitting/underfitting

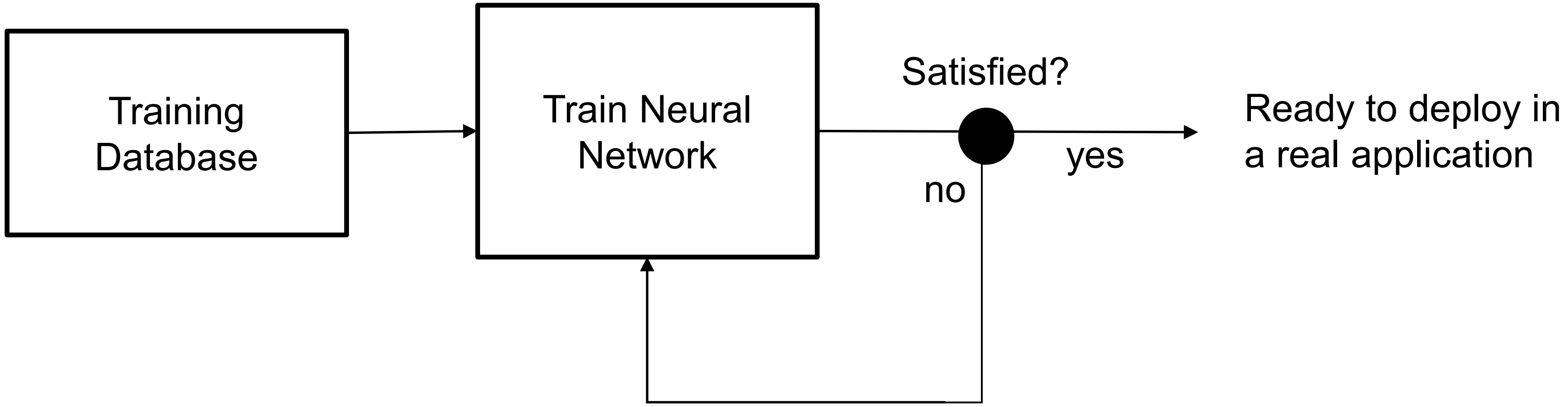
Can we trust it?

Closing the Loop: A Framework for Trustworthy Machine Learning in Power Systems

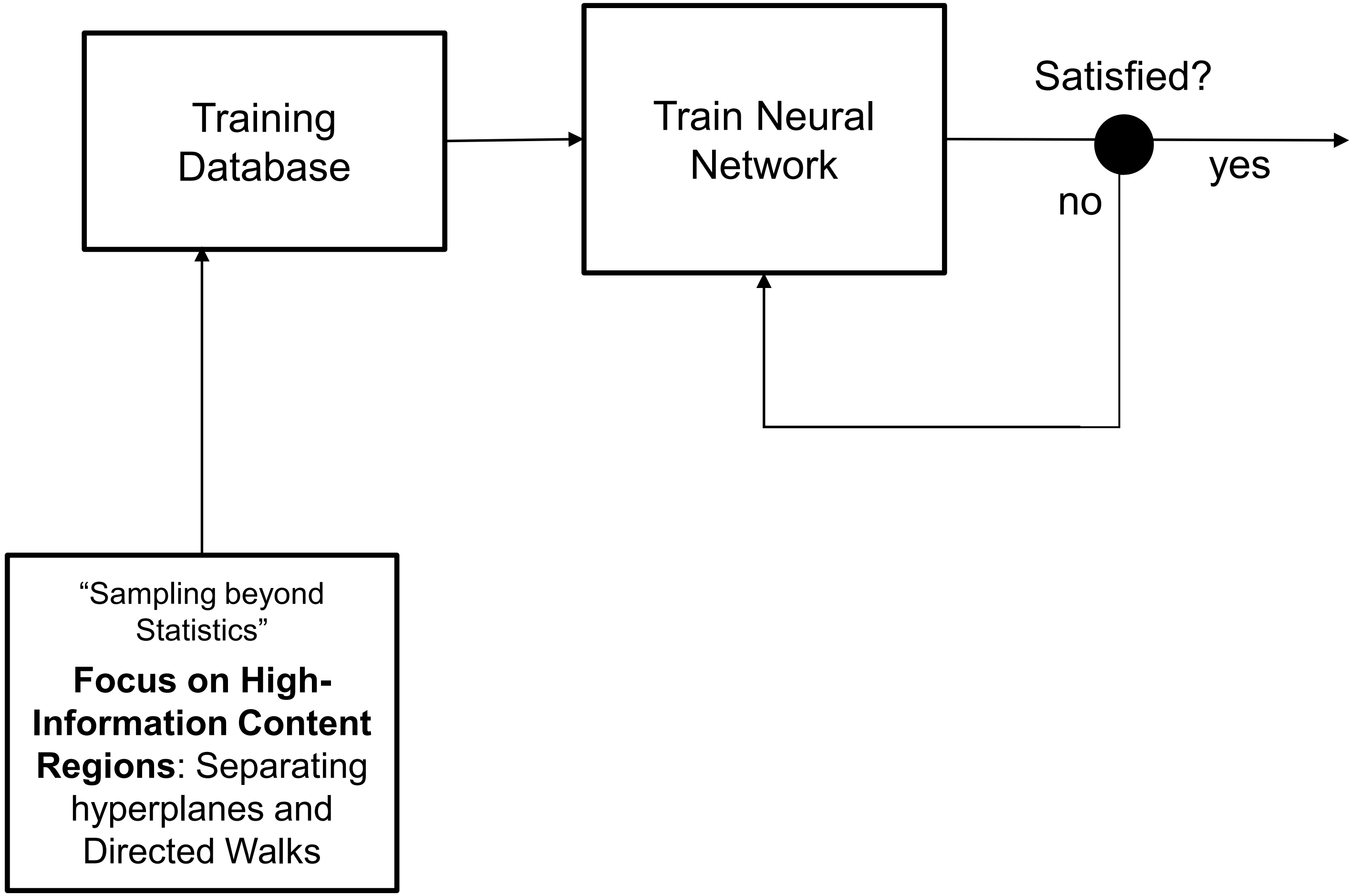
J. Stiasny, S. Chevalier, R. Nellikkath, B. Sævarsson, S. Chatzivasileiadis. Closing the Loop: A Framework for Trustworthy Machine Learning in Power Systems. Accepted to *2022 iREP Symposium - Bulk Power System Dynamics and Control - XI (iREP)*. Banff, Canada. July 2022. [paper | code]

Closing the Loop: Trustworthy ML for Power Systems

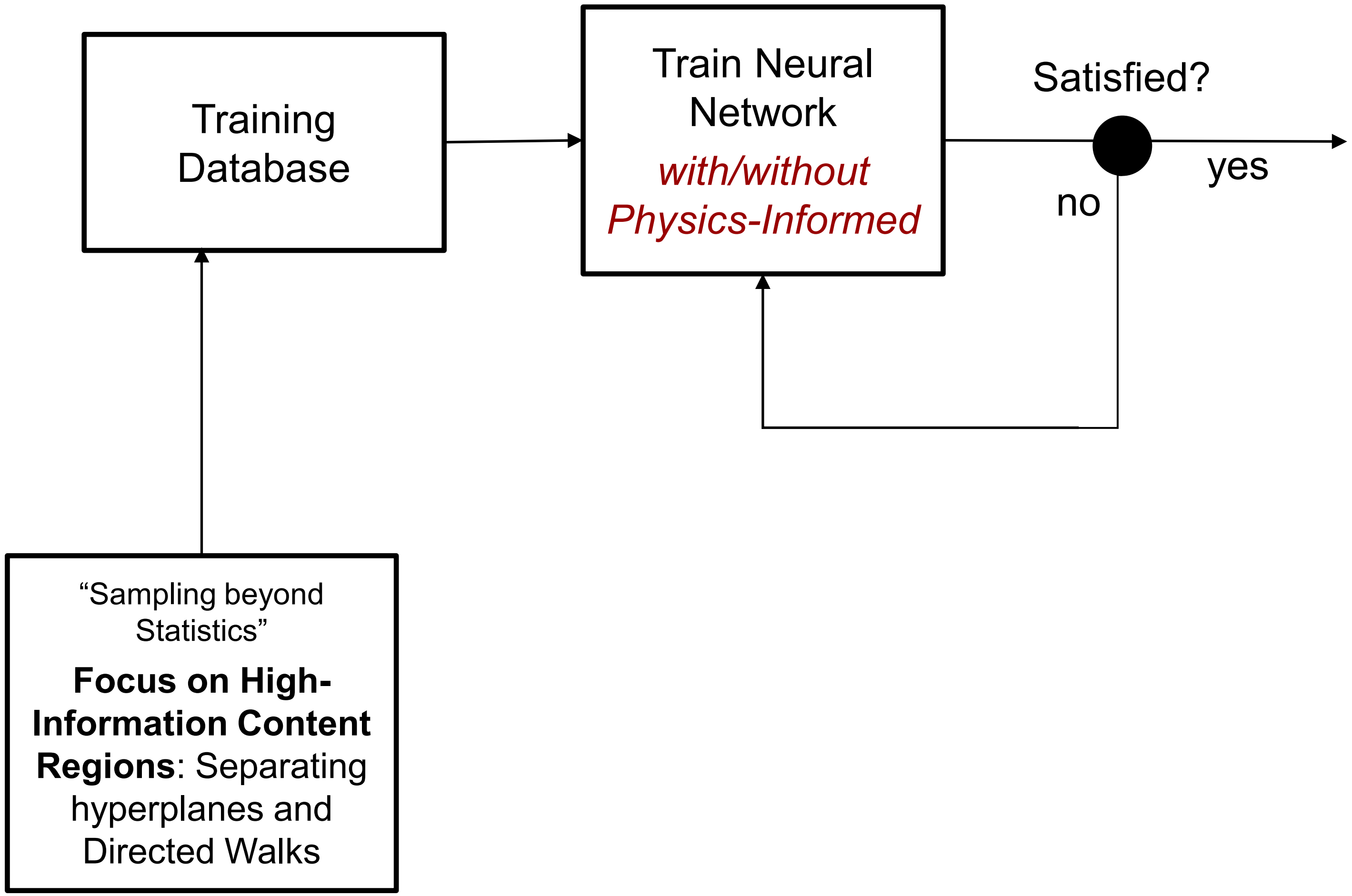
Conventional Neural Network Training for Power System Applications



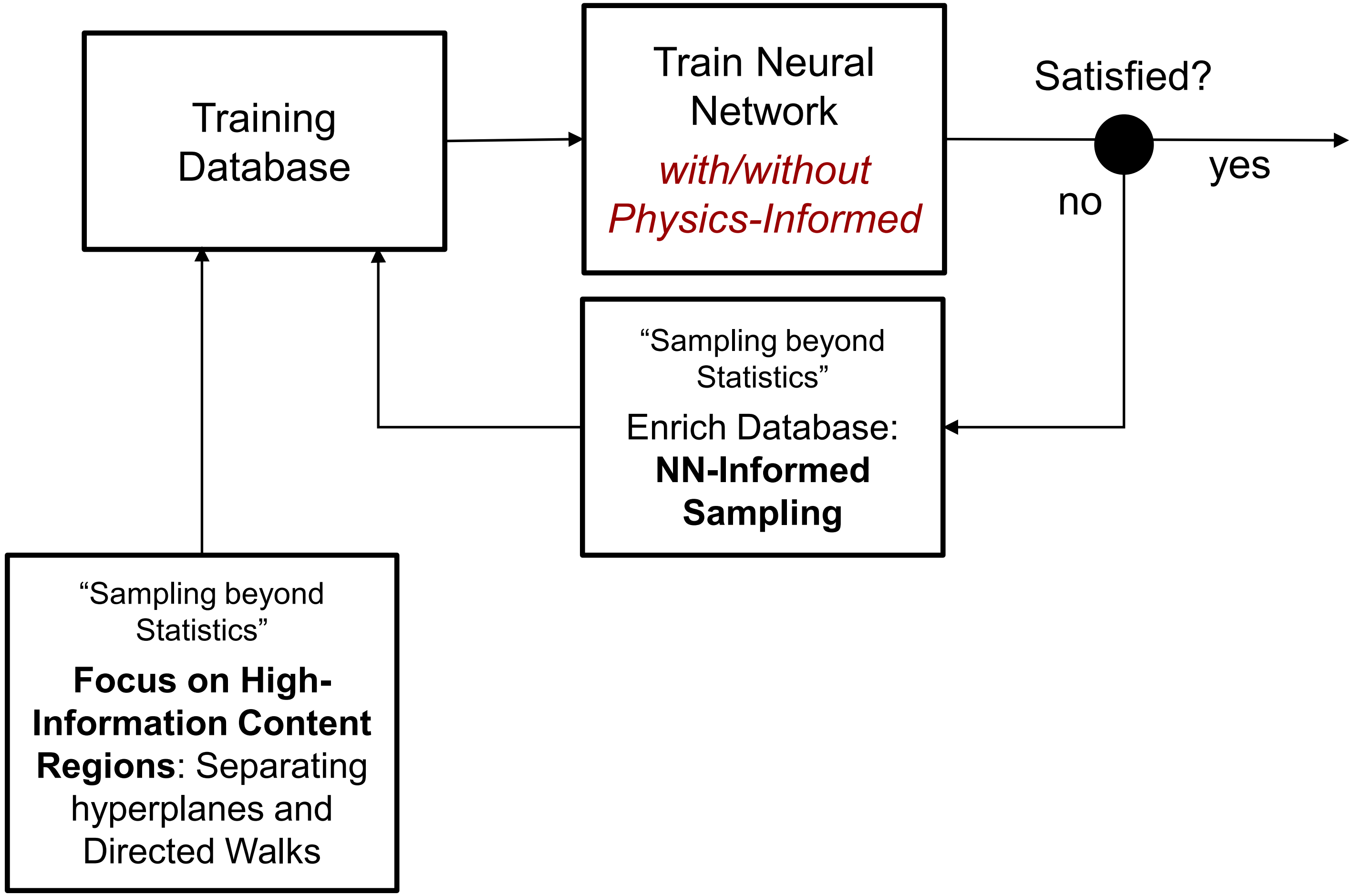
Closing the Loop: Trustworthy ML for Power Systems



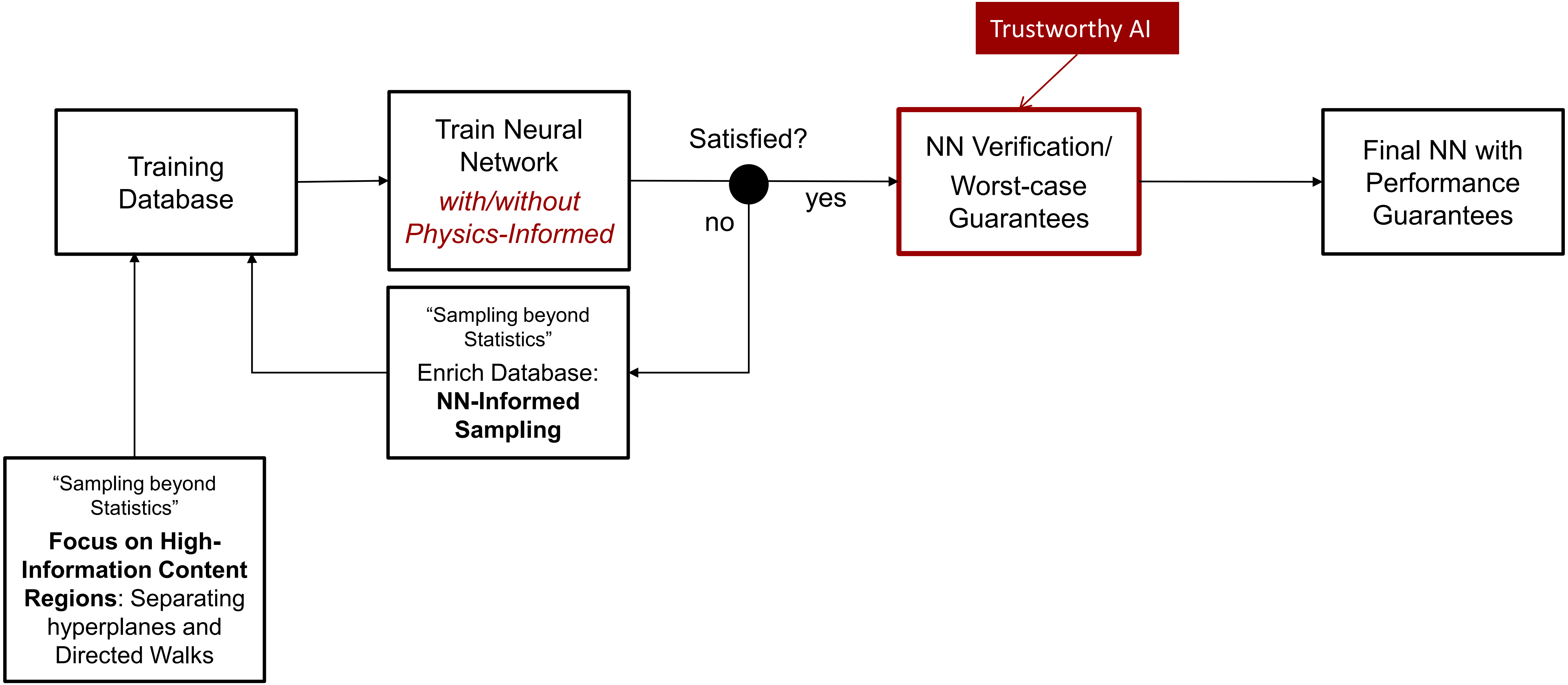
Closing the Loop: Trustworthy ML for Power Systems



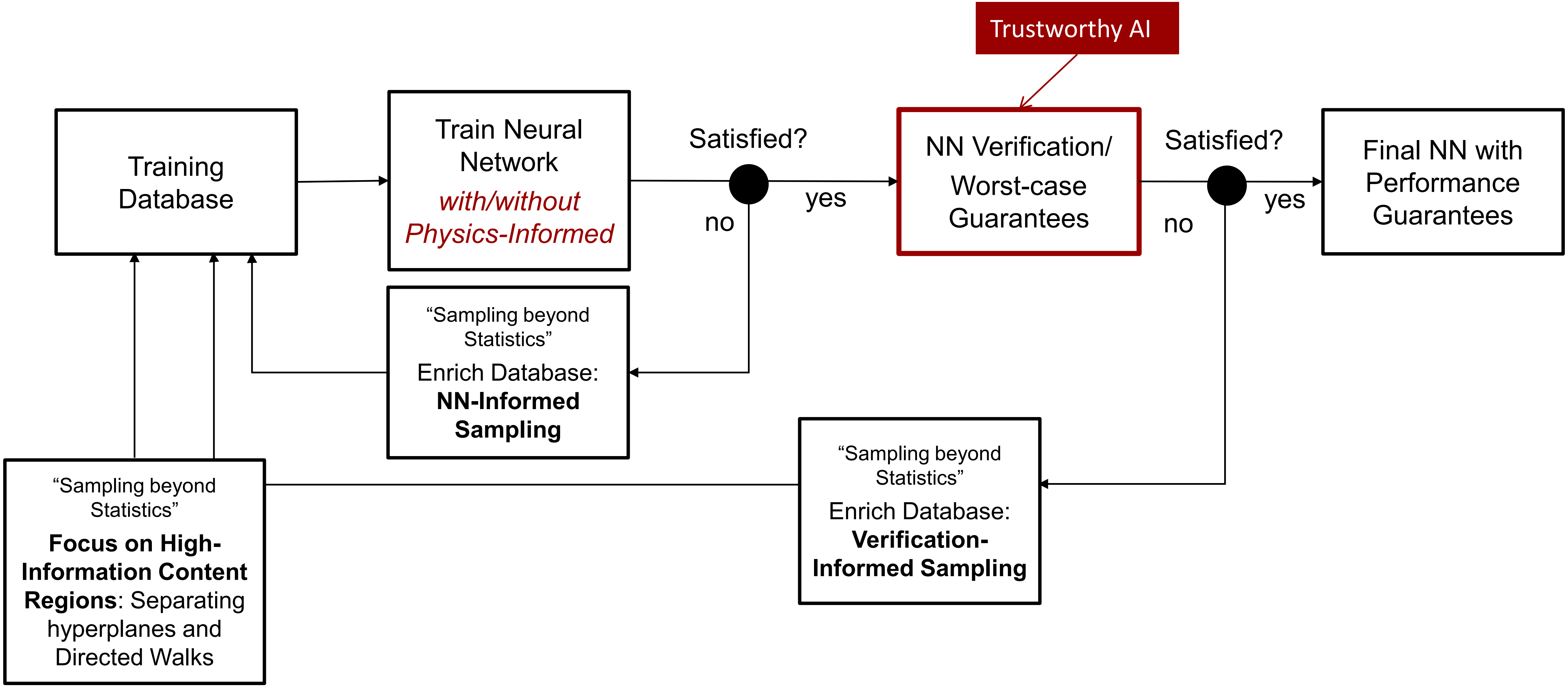
Closing the Loop: Trustworthy ML for Power Systems



Closing the Loop: Trustworthy ML for Power Systems



Closing the Loop: Trustworthy ML for Power Systems



Sampling beyond Statistics: Separating Hyperplanes and Directed Walks

- Historical data are often insufficient
- Need to generate our own data

- Here: generate data for N-1 security+small-signal stability
  - Assessing the stability of 100'000s of operating points is an extremely demanding task
  - Immense search space
  - How can I do it efficiently?

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: Better results with less data

- Historical data are often insufficient
- Need to generate our own data
- Here: generate data for N-1 security+small-signal stability
  - Assessing the stability of 100’000s of operating points is an extremely demanding task
  - Immense search space
  - How can I do it efficiently?

Proposed approach:

- 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
- Generated Databases for IEEE 14-bus and NESTA 162-bus system available!

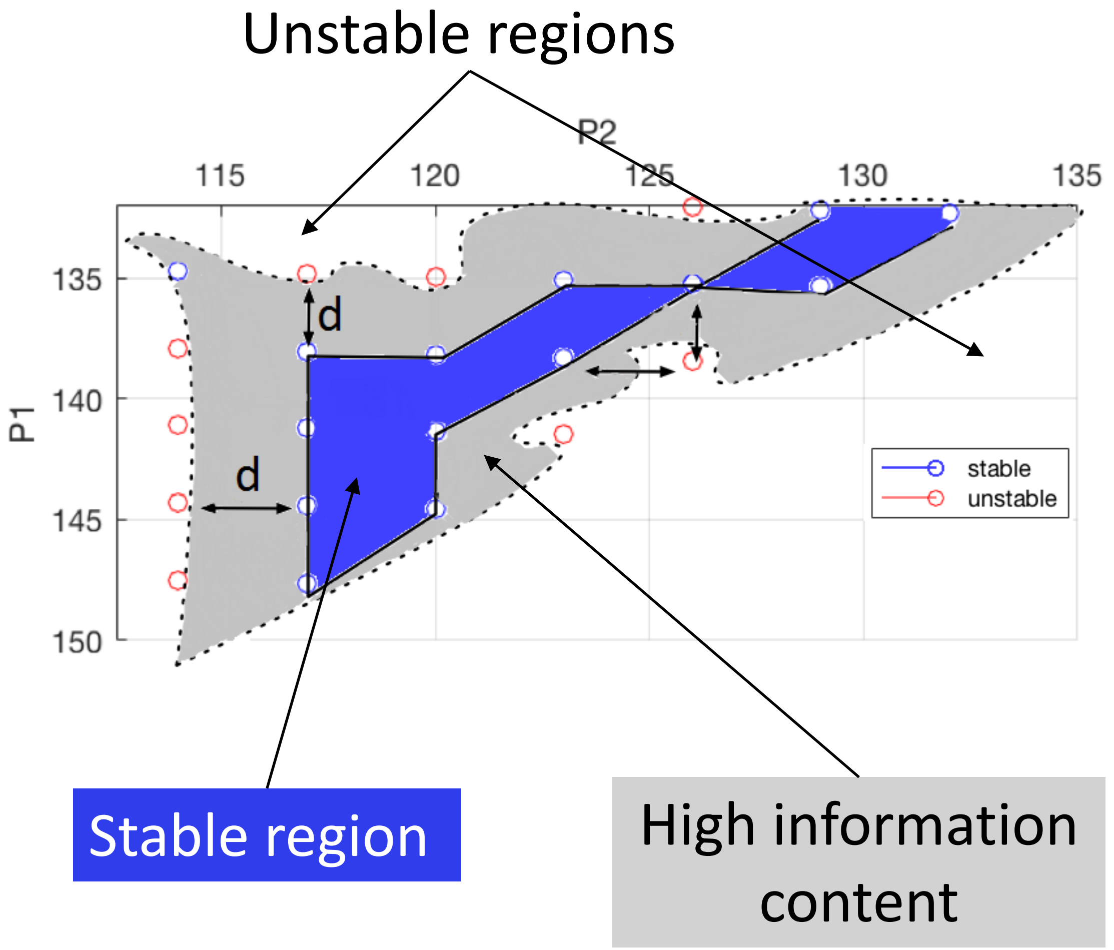
  http://www.chatziva.com/downloads.html#databases

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

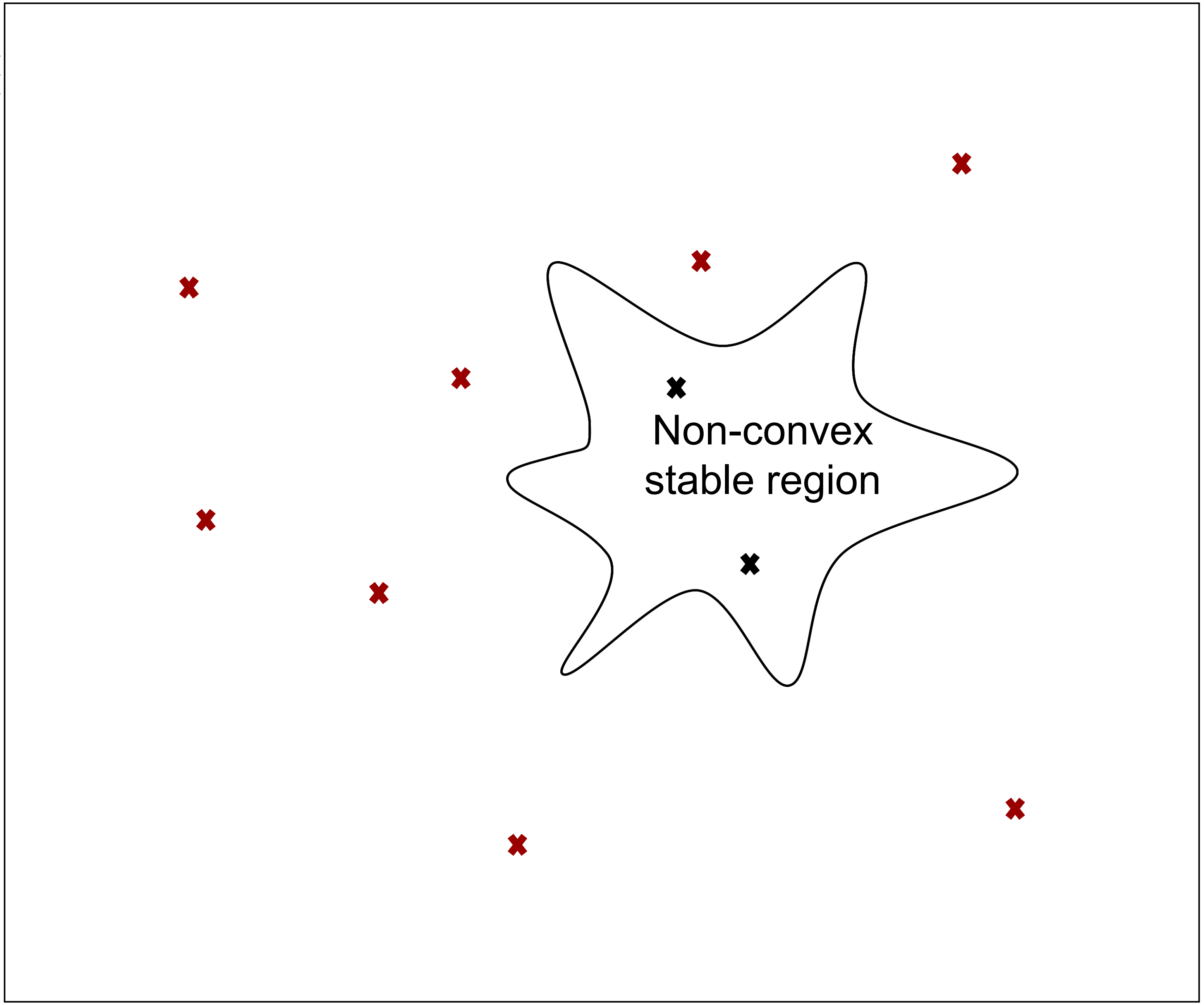
Sampling beyond Statistics: Efficient Database Generation

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

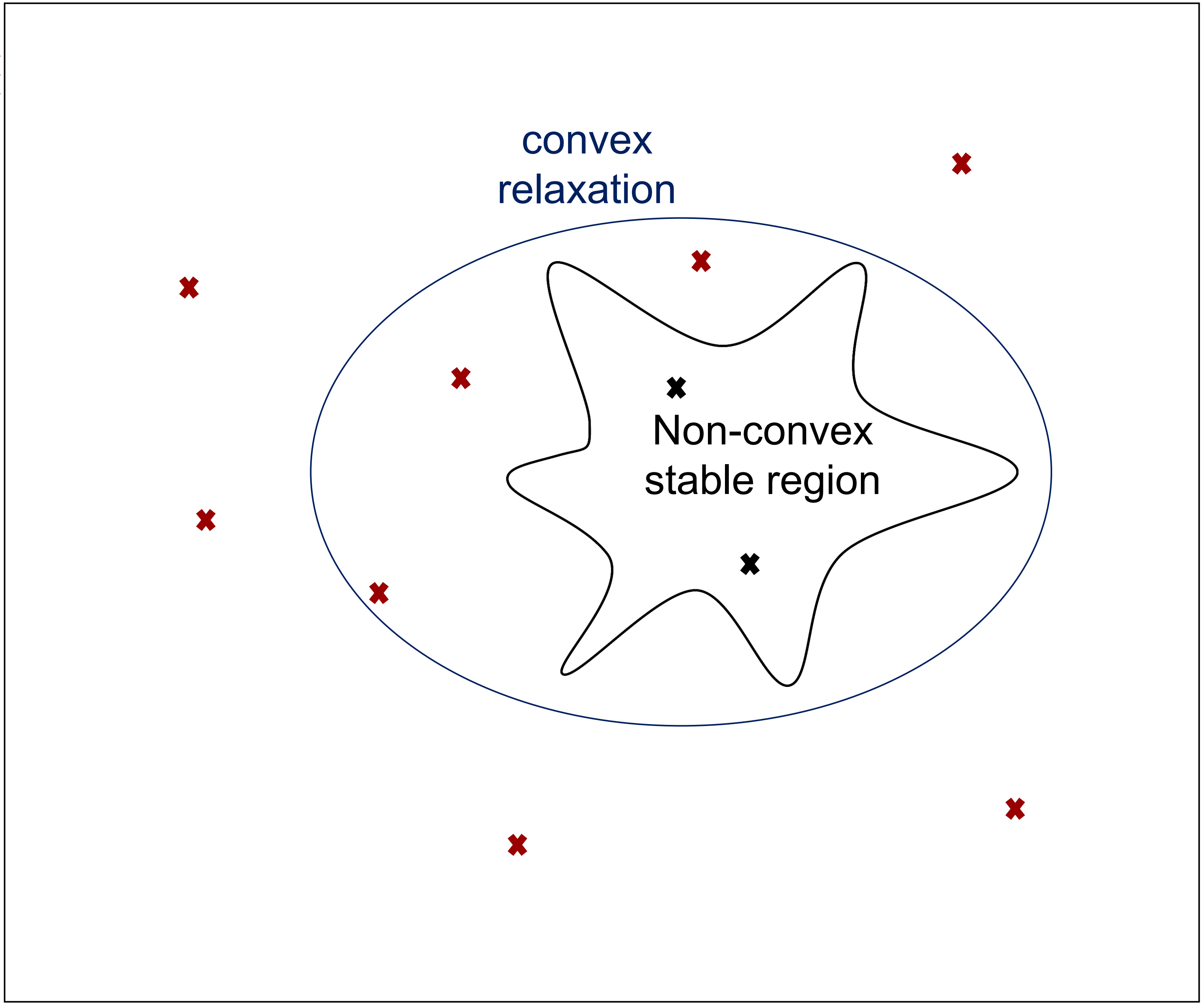
- How?
  1. Using convex relaxations
  2. And "Directed Walks"



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

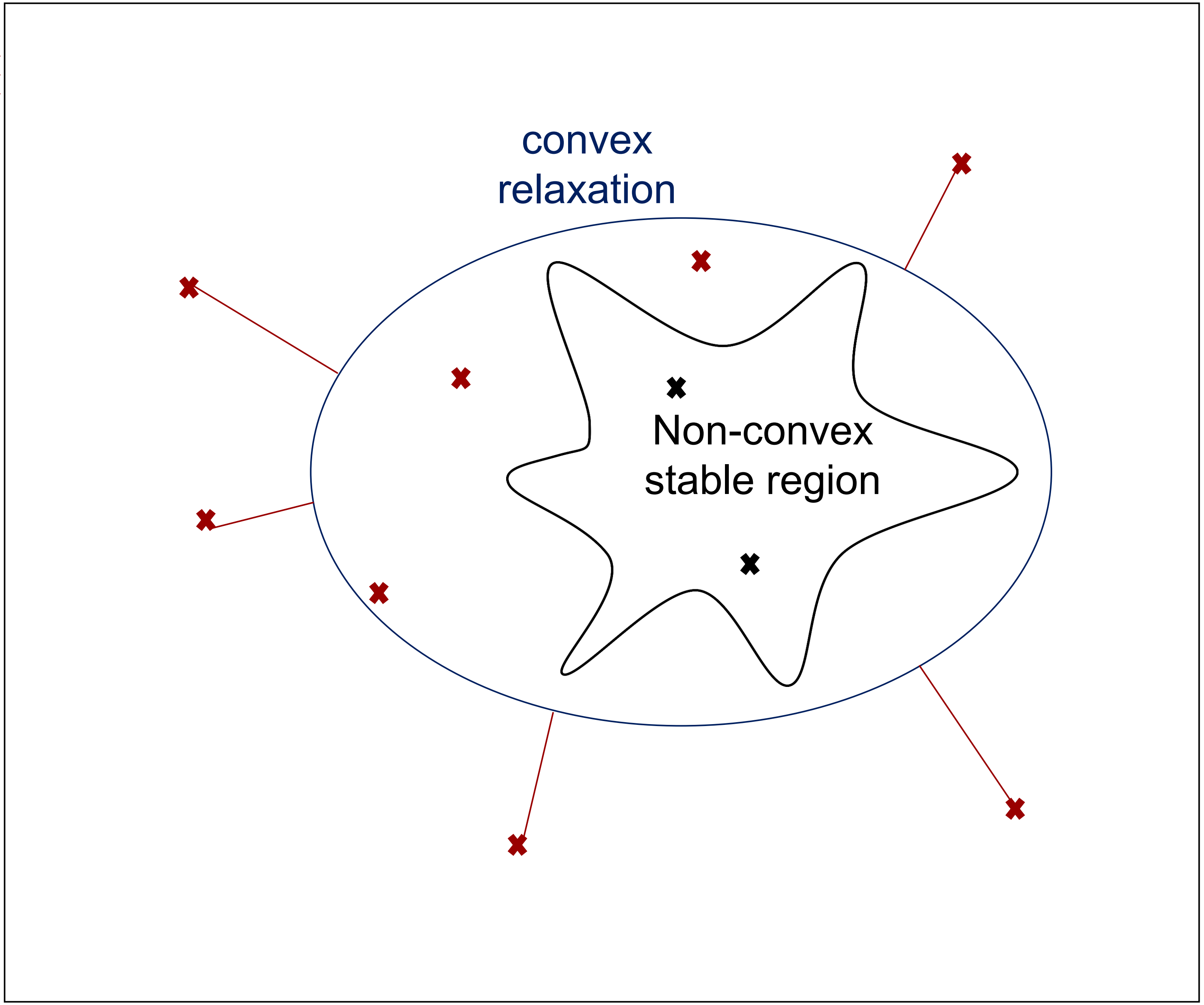


Convex relaxations to discard infeasible regions



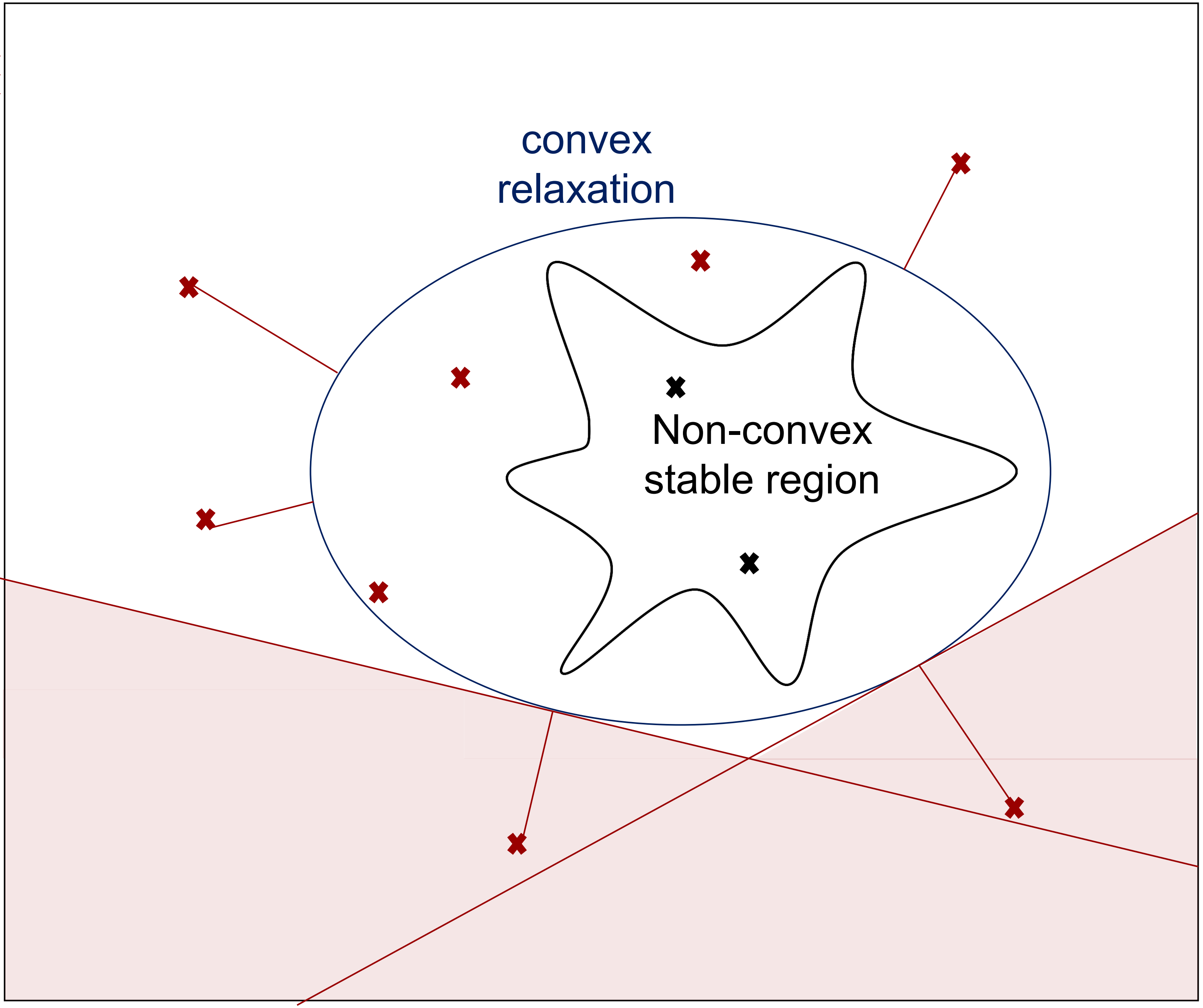
Convex relaxations to discard infeasible regions

- 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



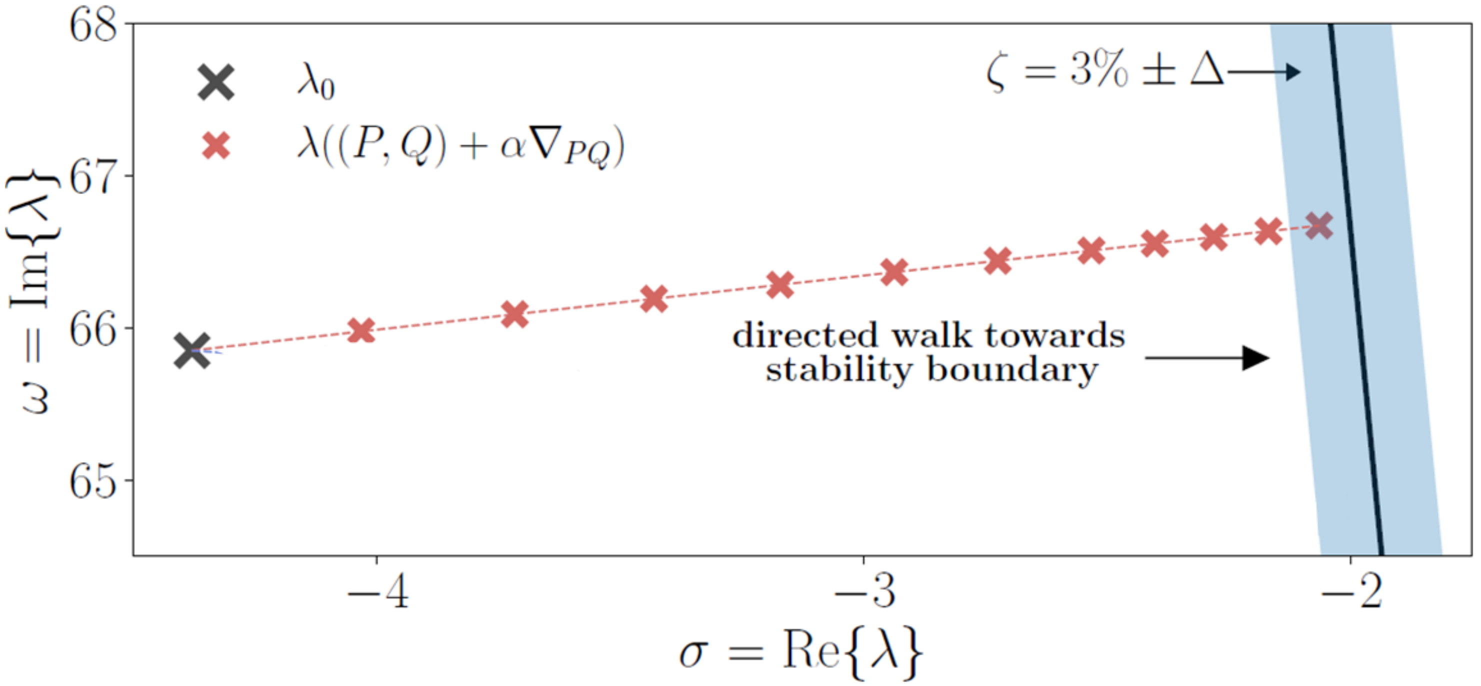
Convex relaxations to discard infeasible regions

- Certificate: if point infeasible for semidefinite relaxation → infeasible for the original problem
- If infeasible point: find minimum radius to feasibility
- Discard all points on one side of the hyperplane

- 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

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

Sampling beyond Statistics: NN-Informed Sampling

- Ideally: enrich the database with points near the stability boundary during NN training
  - But: impossible to know a priori which are these points

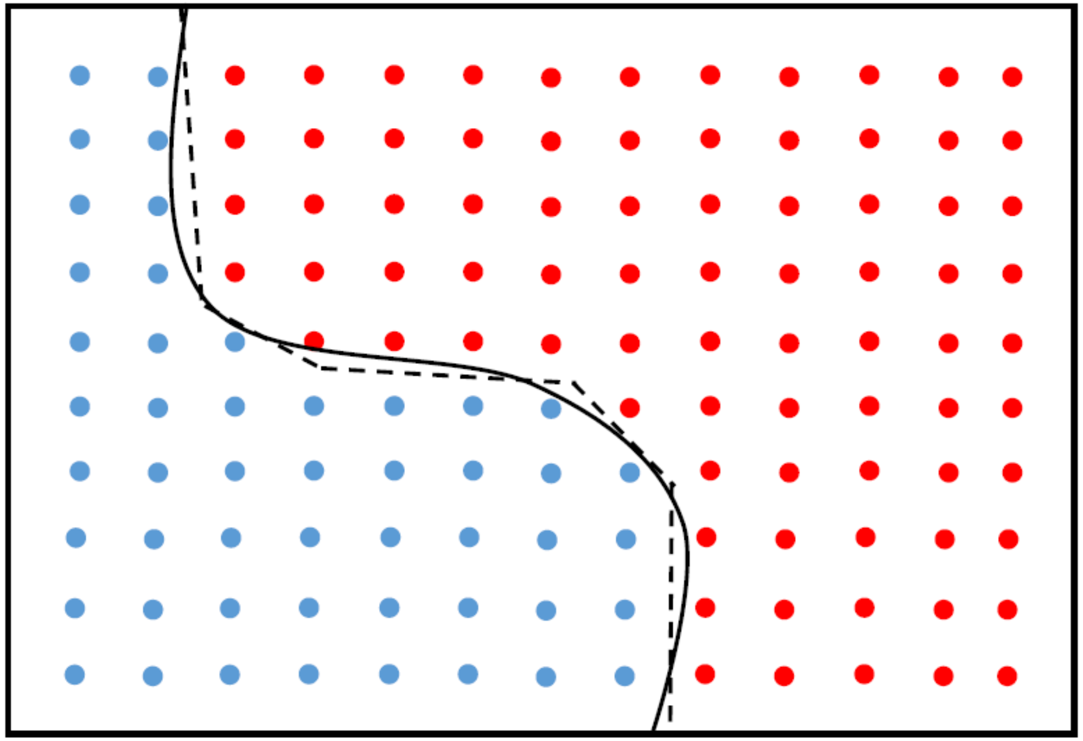
- What do we do?

NN-Informed Sampling

- Ideally: enrich the database with points near the stability boundary
  - But: impossible to know a priori which are these points

- What do we do?

1. Sample 1’000’000 random points and have the NN assess them
   - Extremely fast → NN will take some minutes to assess all of them



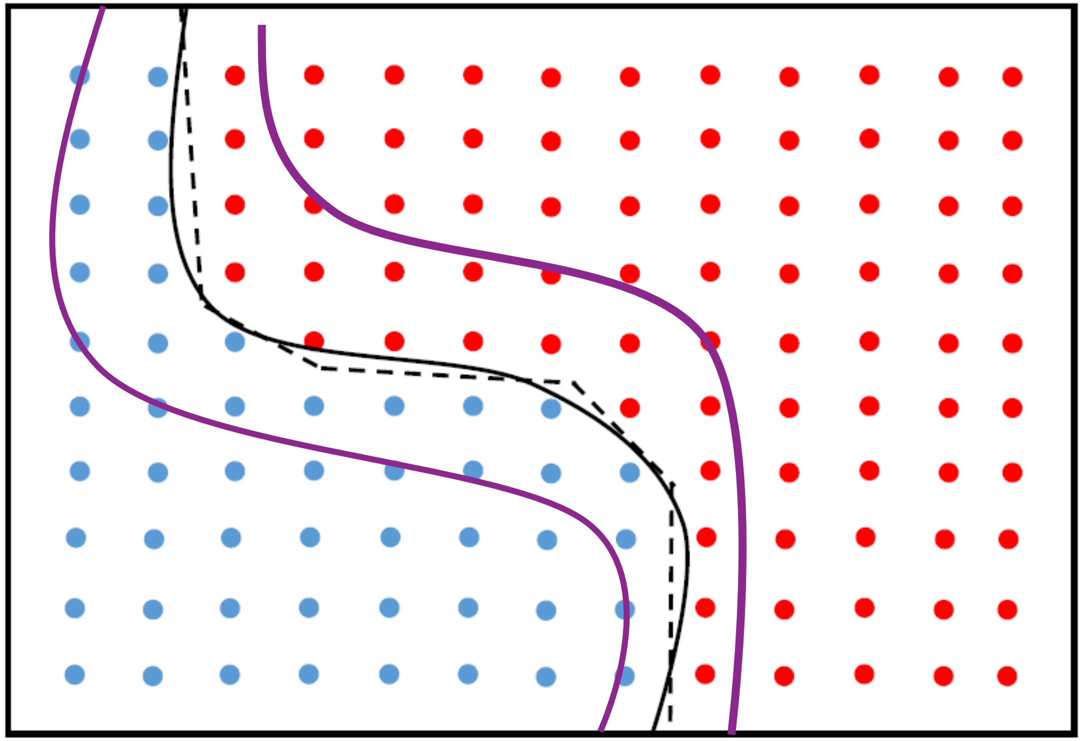
NN-Informed Sampling

- Ideally: enrich the database with points near the stability boundary
  - But: impossible to know a priori which are these points

- What do we do?

1. Sample 1’000’000 random points and have the NN assess them
   - Extremely fast → NN will take some minutes to assess all of them

2. From the NN assessment: identify the region close to the stability boundary



NN-Informed Sampling

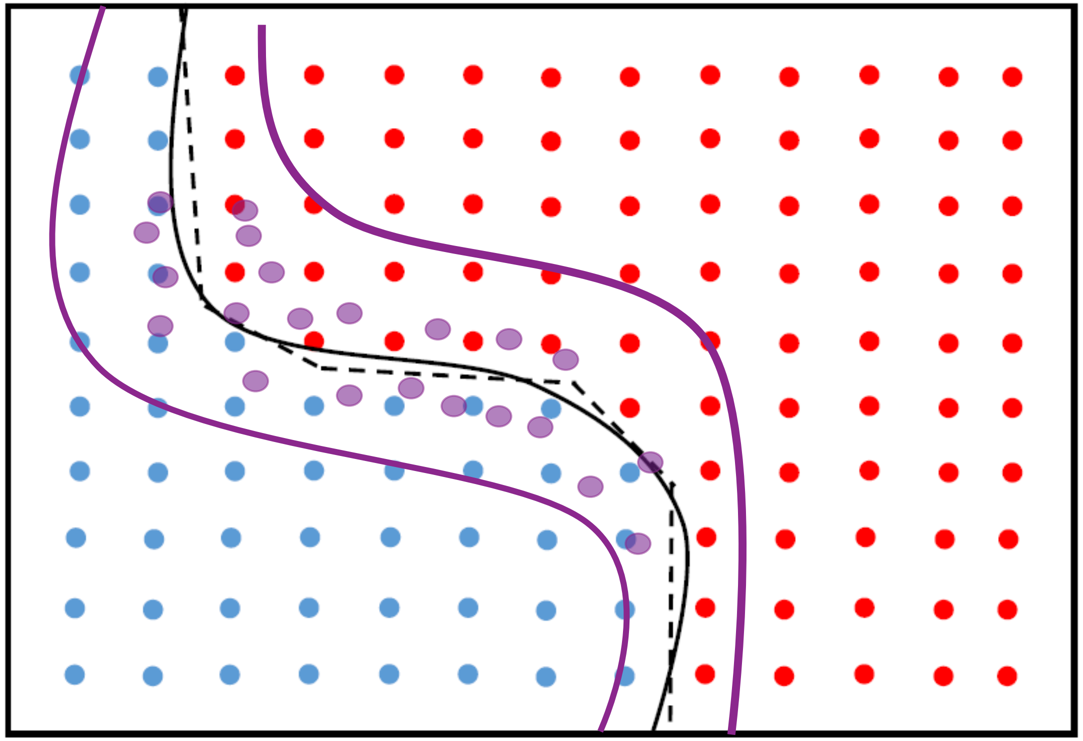
- Ideally: enrich the database with points near the stability boundary
  - But: impossible to know a priori which are these points

- What do we do?

1. Sample 1’000’000 random points and have the NN assess them
   - Extremely fast → NN will take some minutes to assess all of them

2. From the NN assessment: identify the region close to the stability boundary

3. Sample 200 points in this region, compute the ground truth (=run N-1 and small signal stability), and enrich the database

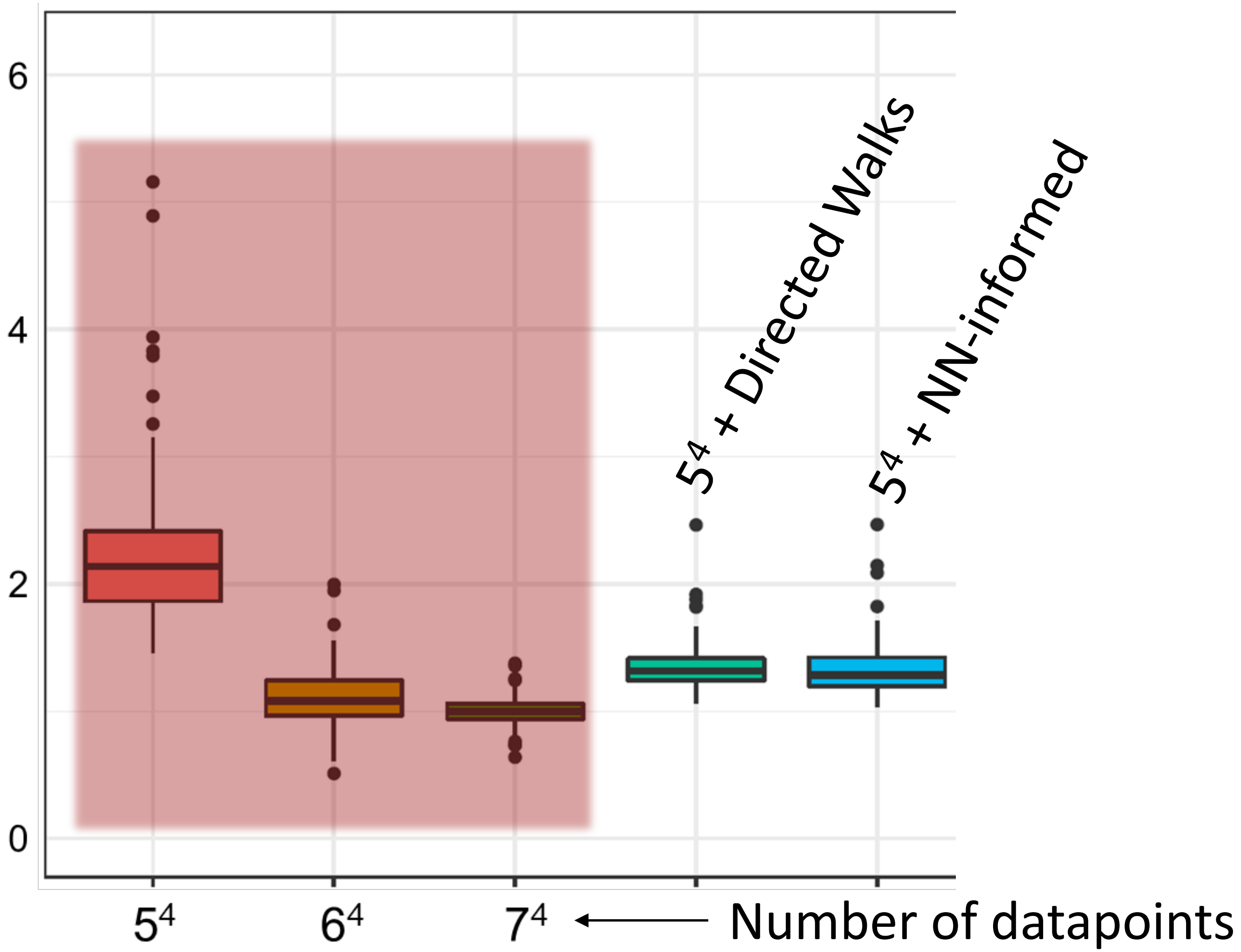


Sampling beyond statistics: Better results with less data

- Larger datasets achieve lower error
  - 6^4 : ~2x more data than 5^4
  - 7^4 : ~4x more data than 5^4

- The directed walks and the NN-informed resampling achieve the same performance with half the datapoints

Mean squared error (test set loss)



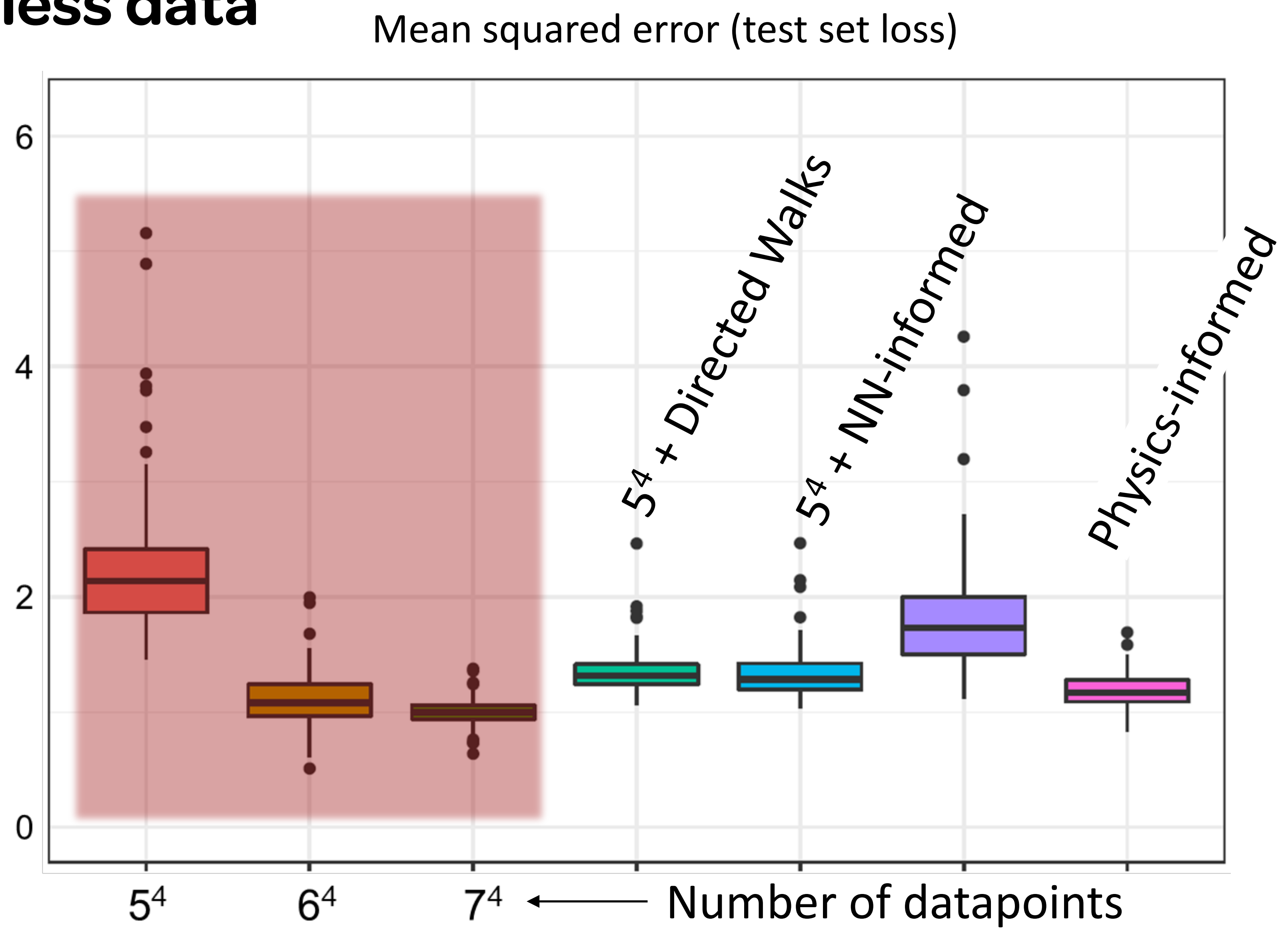
Note: Actual performance of DW and NI depends on the case study. But the trend remains the same across all our experiments

Sampling beyond statistics: Better results with less data

- Larger datasets achieve lower error
  - 6^4 : ~2x more data than 5^4
  - 7^4 : ~4x more data than 5^4

- The directed walks and the NN-informed resampling achieve the same performance with half the datapoints

- Physics-Informed Neural Networks can achieve similar results



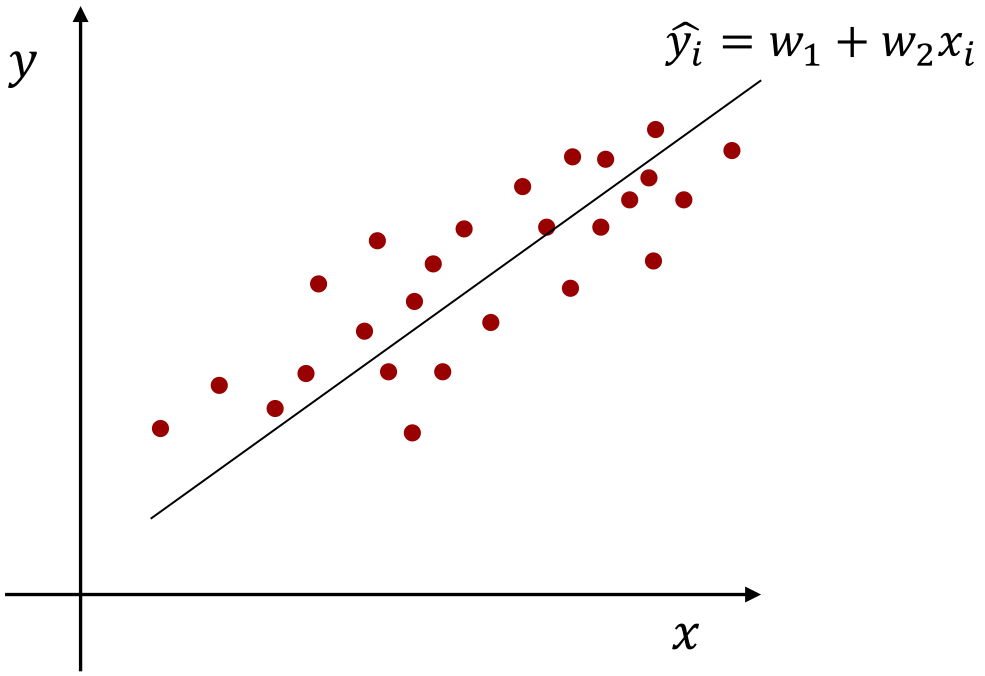
Note: Actual performance of DW, NI, and PINNs depends on the case study. But the trend remains the same across all our experiments

Physics-Informed Neural Networks for Power Systems

y_i: actual/correct value

$\hat{y}_i$: estimated value

Neural Networks: An advanced form of non-linear regression



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

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

s.t.

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

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 with respect to the input 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 (e.g. Pytorch, 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

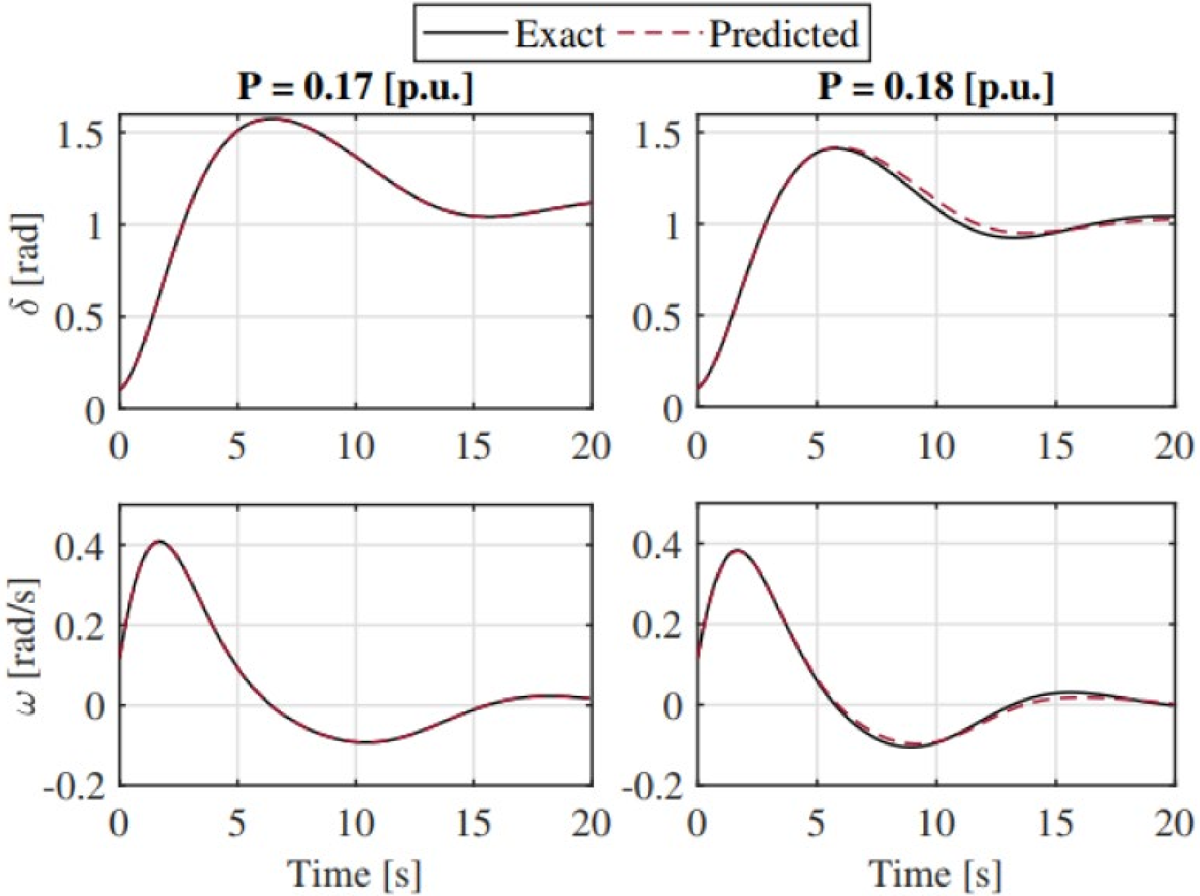
“Original” Loss function

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

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

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

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



G. S. Misyris, A. Venzke, S. Chatzivasileiadis, **Physics-Informed Neural Networks for Power Systems**. Presented at the **Best Paper Session** of IEEE PES GM 2020. https://arxiv.org/pdf/1911.03737.pdf

Physics-Informed Neural Networks for Power Systems

"Original" Loss function

"Physics-Informed" term

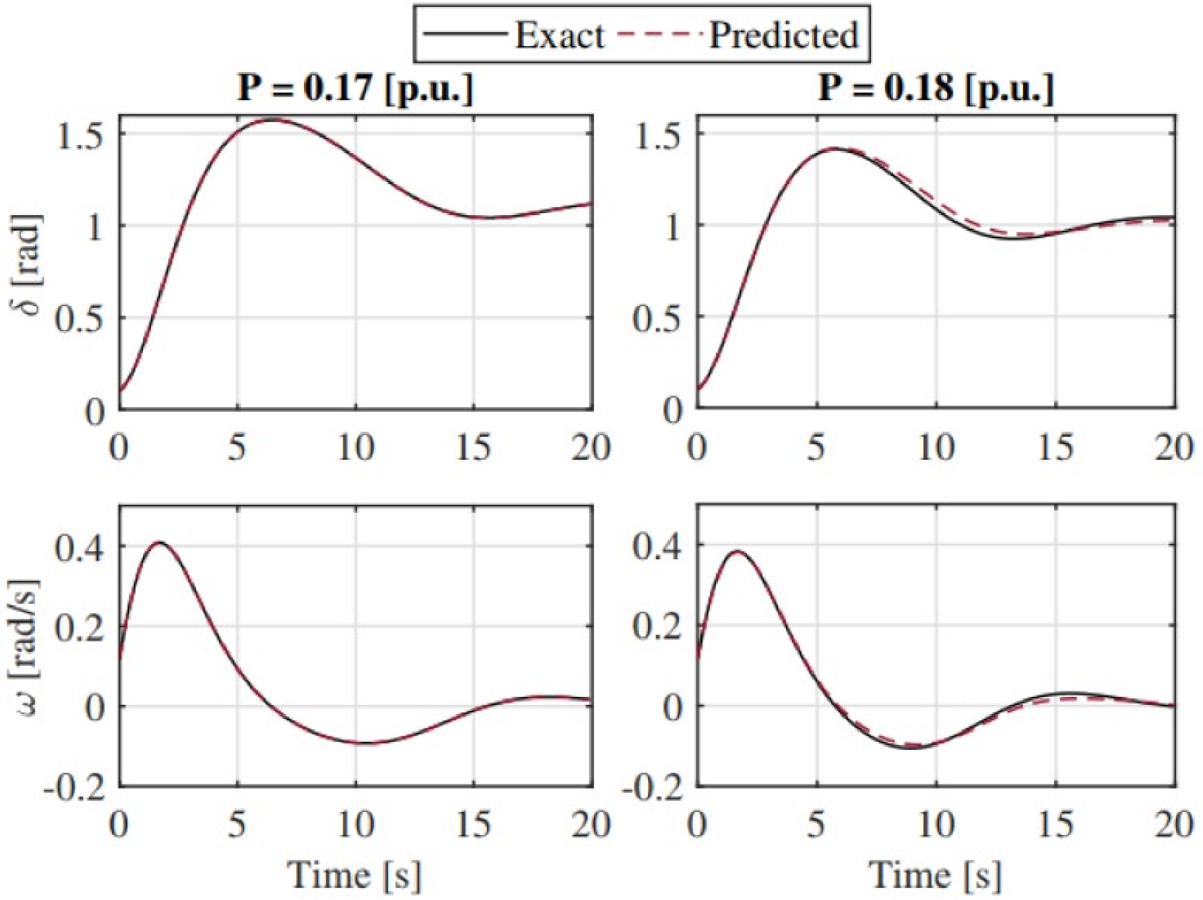
$$\min_{\mathbf{W},\mathbf{b}} \quad \frac{1}{|N_\delta|}\sum_{i\in N_\delta}|\hat{\delta}-\delta^i|^2 + \boxed{\frac{1}{|N_f|}\sum_{i\in N_f}|f(\hat{\delta})|^2} \tag{6a}$$

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

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

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

Swing equation



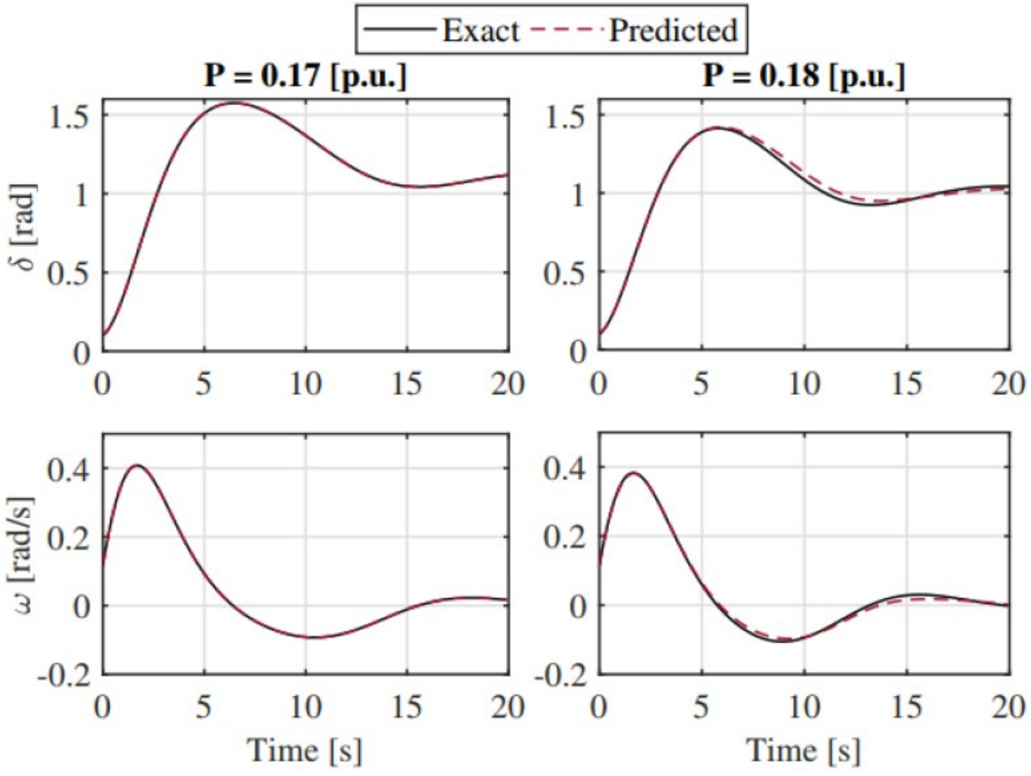
G. S. Misyris, A. Venzke, S. Chatzivasileiadis, **Physics-Informed Neural Networks for Power Systems**. Presented at the **Best Paper Session** of IEEE PES GM 2020. https://arxiv.org/pdf/1911.03737.pdf

Physics-Informed Neural Networks for Power Systems

- Physics-Informed Neural Networks (PINN) could potentially replace solvers for systems of differential-algebraic equations in the long-term
  - **Probable power system application: Extremely fast screening of critical contingencies**

- In our example: PINN 87 times faster than ODE solver

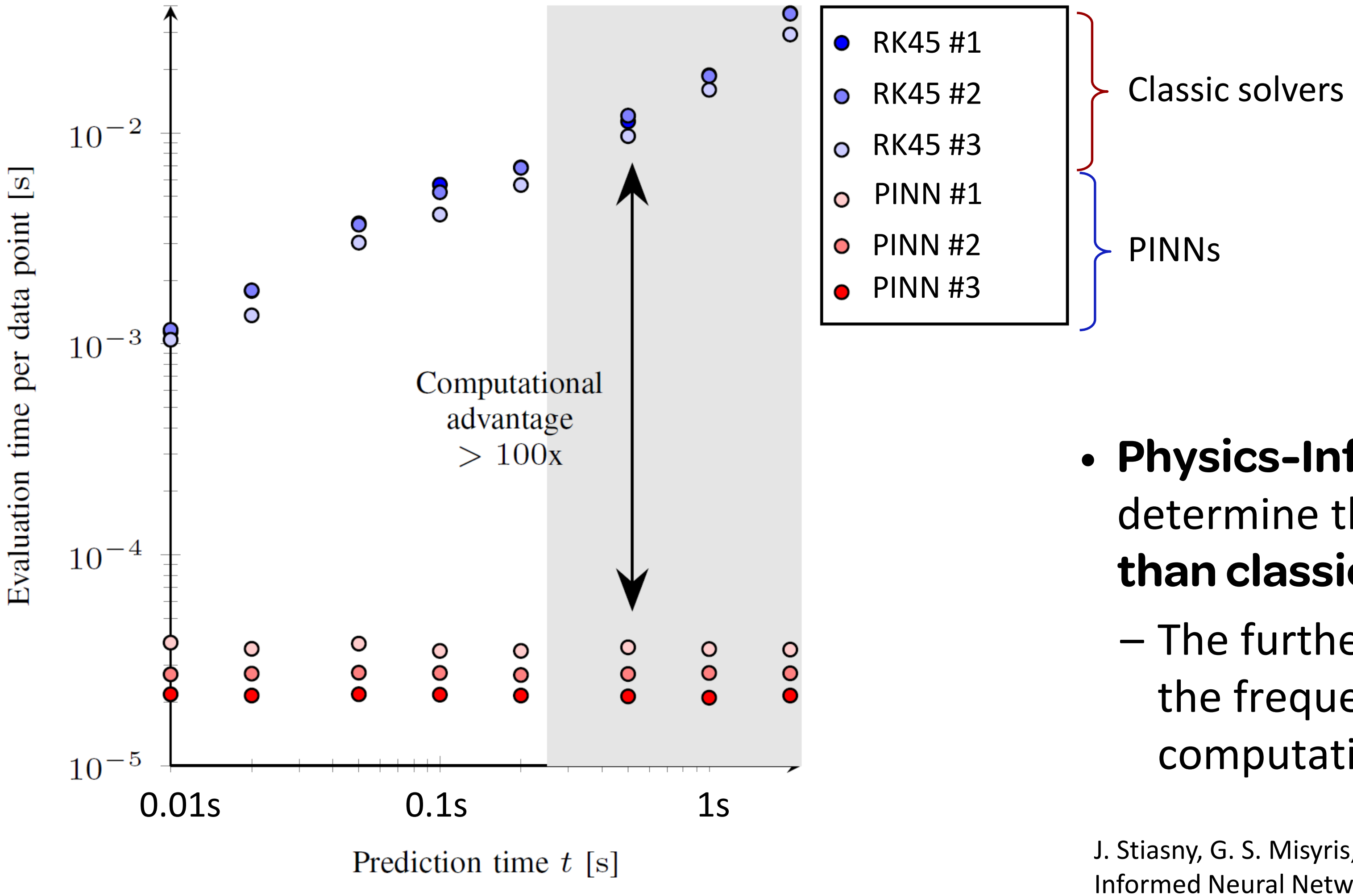
- Can **directly estimate** the rotor angle at **any** time instant



Code is available on GitHub: https://github.com/jbesty

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

Computation time: Classical numerical solvers vs. Physics-Informed NNs



- **Physics-Informed Neural Networks** can determine the outputs more than **100x faster than classical numerical solvers**
  - The further ahead we look in time, e.g. what is the frequency at t=1s, the larger the computational advantage is

J. Stiasny, G. S. Misyris, S. Chatzivasileiadis, Transient Stability Analysis with Physics-Informed Neural Networks. https://arxiv.org/abs/2106.13638 [code]

Thoughts on Physics-Informed Neural Networks (PINNs)

- PINNs convert NN training from *supervised learning* to *unsupervised learning*

- **PINNs show a clear benefit for problems that include PDEs and ODEs**
  - The benefit is less distinct for problems with algebraic equations only
    - Trade-off between training database size and PINN training time

- We believe that **it is possible to develop a NN-based simulator for time-domain simulations**
  - Currently working with NVIDIA to accelerate PINNs, and
  - Ørsted (the largest offshore wind developer) to develop a tool for real wind-farm electrical design problems

- **How do we generate NNs that are valid for a wide range of topologies?**
  - ”**Decompose** them[1]” → PINNs for single components? Or for sub-graphs of the whole system
  - For steady-state (algebraic) problems → **Graph Neural Networks?**

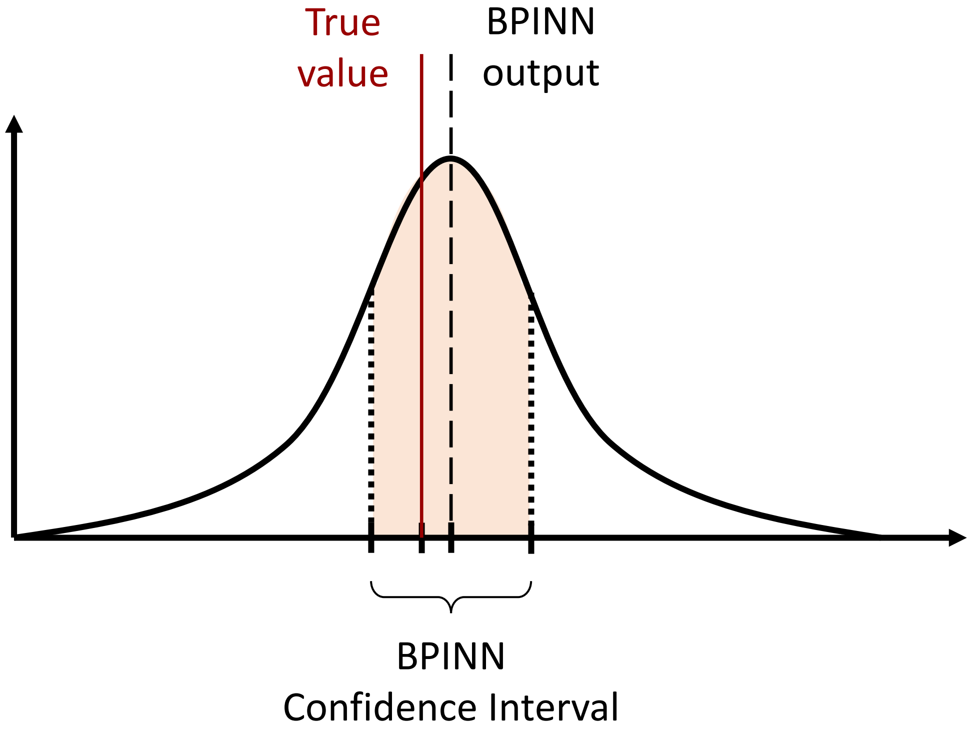
- How do we verify PINNs for dynamic systems? (with ODEs and PDEs)

- Can we have a **confidence measure** of the PINN output?
  - Bayesian PINNs?

[1] *M. Chatzos, T. W. K. Mak and P. V. Hentenryck, "Spatial Network Decomposition for Fast and Scalable AC-OPF Learning,“ 2022*

Bayesian Physics-Informed Neural Networks

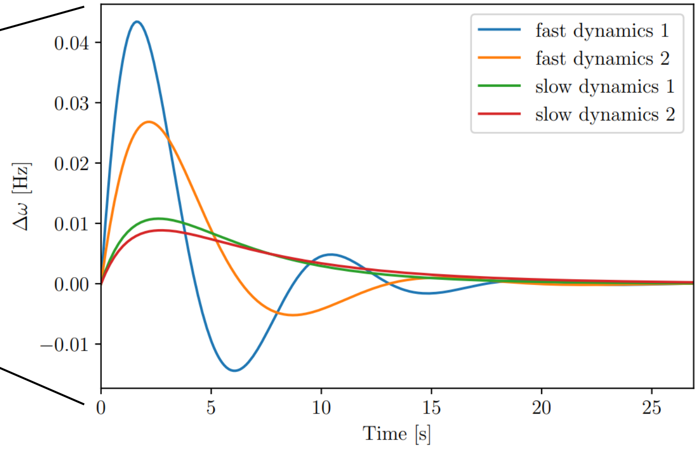
- Why Bayesian?
  - Add a **confidence measure** to the NN output
  - Very useful for forecasting, **system identification**, and many others



L. Yang, X. Meng, and G. E. Karniadakis, “B-PINNs: Bayesian physicsinformed neural networks for forward and inverse PDE problems with noisy data,” Journal of Computational Physics, vol. 425, p. 109913, 2021.

Bayesian Physics-Informed Neural Networks

Input



- Example:
  - Receive **frequency** time series as input
  - What is the **inertia** and **damping** coefficient of the system?

Output

$$\overbrace{\boxed{\hat{M}}\dot{\omega}(t) + \boxed{\hat{D}}}\omega(t) + A\sin\delta(t) - P_m = 0$$

How much is inertia and damping?

Compare:

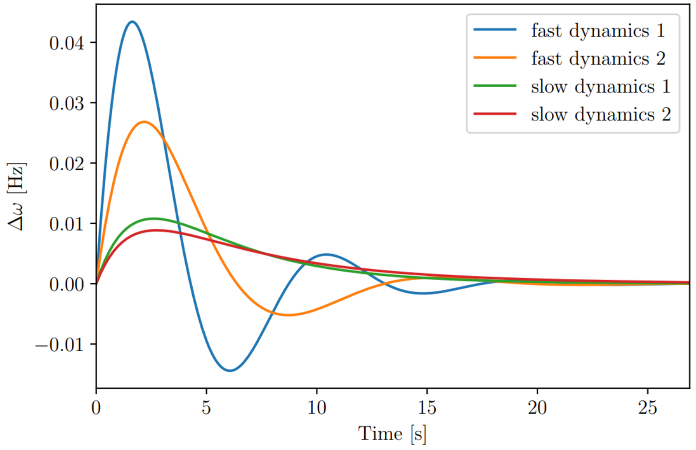
1. **SINDy** = among the recent most popular non-linear system identification methods
2. **PINNs**
3. Bayesian PINNs (**BPINNs**)

S. Stock, J. Stiasny, D. Babazadeh, C. Becker, S. Chatzivasileiadis, Bayesian Physics-Informed Neural Networks for Robust System Identification of Power Systems. https://arxiv.org/abs/2212.11911

Bayesian Physics-Informed Neural Networks (BPINNs)

- Example:
  - Receive **frequency** time series as input
  - What is the **inertia** and **damping** coefficient of the system?

Input



	Inertia Estim. Error (%)	Damping Estim. Error (%)
SINDy	3.80%	0.14%
PINN	0.34%	0.84%
BPINN	**1.20% ± 9.26%**	**0.01% ± 0.011%**

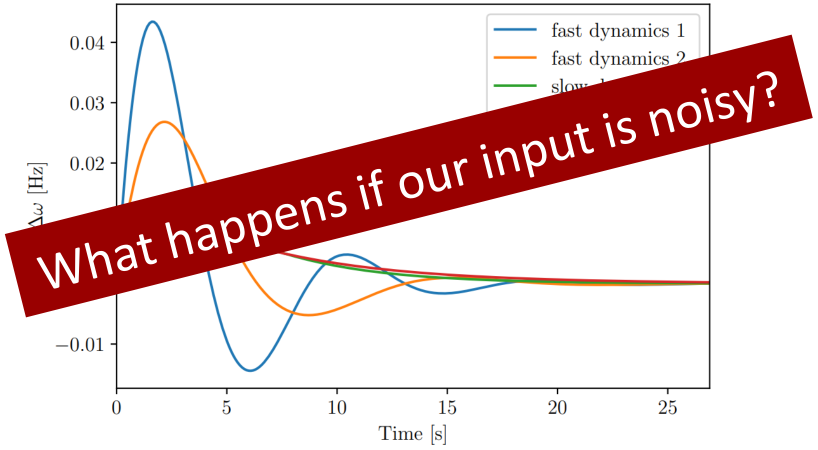
1. All approaches perform well
2. **BPINN is the only with confidence interval**

Note: We used default parameters for SINDy and BPINNs; for PINNs, we used tailored parameters, based on our experience with PINNs over the past years.

S. Stock, J. Stiasny, D. Babazadeh, C. Becker, S. Chatzivasileiadis, Bayesian Physics-Informed Neural Networks for Robust System Identification of Power Systems. https://arxiv.org/abs/2212.11911

Bayesian Physics-Informed Neural Networks (BPINNs)

Input



- Example:
  - Receive **frequency** time series as input
  - What is the **inertia** and **damping** coefficient of the system?

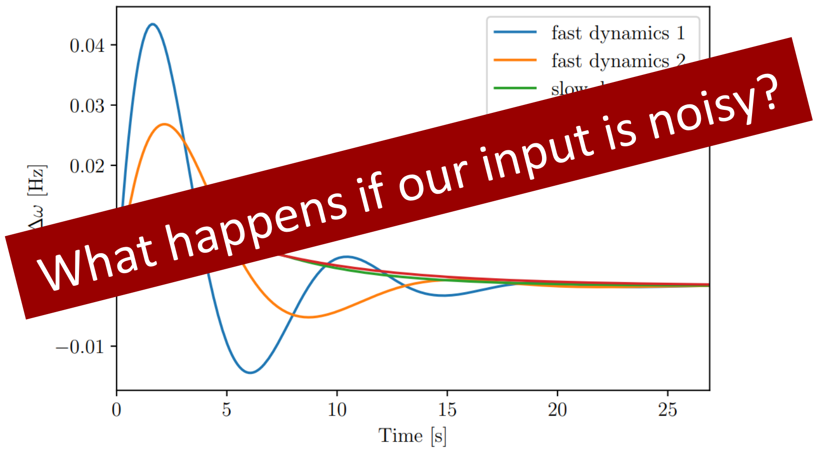
	Inertia Estim. Error (%)	Damping Estim. Error (%)
SINDy	3.80%	0.14%
PINN	0.34%	0.84%
BPINN	1.20% ± 9.26%	0.01% ± 0.011%

S. Stock, J. Stiasny, D. Babazadeh, C. Becker, S. Chatzivasileiadis, Bayesian Physics-Informed Neural Networks for Robust System Identification of Power Systems. https://arxiv.org/abs/2212.11911

Bayesian Physics-Informed Neural Networks (BPINNs)

- Example:
  - Receive **frequency** time series as input
  - What is the **inertia** and **damping** coefficient of the system?

Input



	Inertia Estim. Error (%)	Damping Estim. Error (%)
SINDy	3.80%	0.14%
PINN	0.34%	0.84%
BPINN	1.20% ± 9.26%	0.01% ± 0.011%

Noise = 5%

	Inertia Estim. Error (%)	Damping Estim. Error (%)
SINDy	37.88%	9.67%
PINN	0.41%	3.99%
BPINN	2.03% ± 21.59%	0.02% ± 0.011%

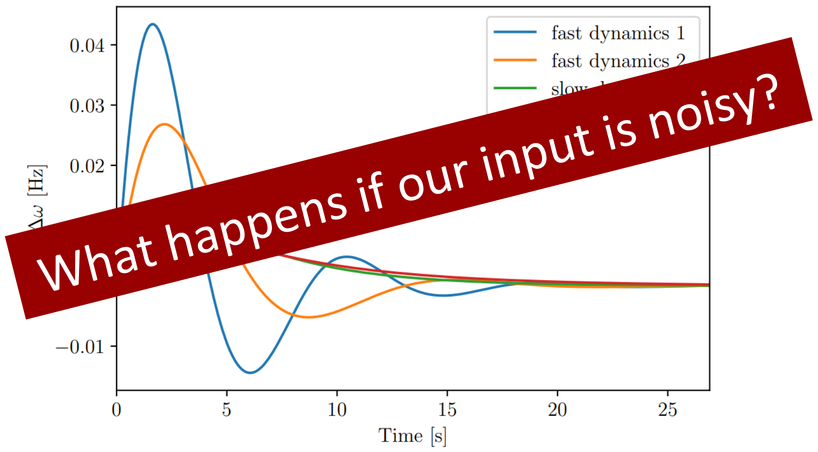
If there is noise, **SINDy** results to: **10x-90x larger error**

S. Stock, J. Stiasny, D. Babazadeh, C. Becker, S. Chatzivasileiadis, Bayesian Physics-Informed Neural Networks for Robust System Identification of Power Systems. https://arxiv.org/abs/2212.11911

Bayesian Physics-Informed Neural Networks (BPINNs)

- Example:
  - Receive **frequency** time series as input
  - What is the **inertia** and **damping** coefficient of the system?

Input



	Inertia Estim. Error (%)	Damping Estim. Error (%)
SINDy	3.80%	0.14%
PINN	0.34%	0.84%
BPINN	1.20% ± 9.26%	0.01% ± 0.011%

Noise = 5%

	Inertia Estim. Error (%)	Damping Estim. Error (%)
SINDy	37.88%	9.67%
PINN	0.41%	3.99%
BPINN	2.03% ± 21.59%	0.02% ± 0.011%

If there is noise, **SINDy** results to: **10x-90x larger error**

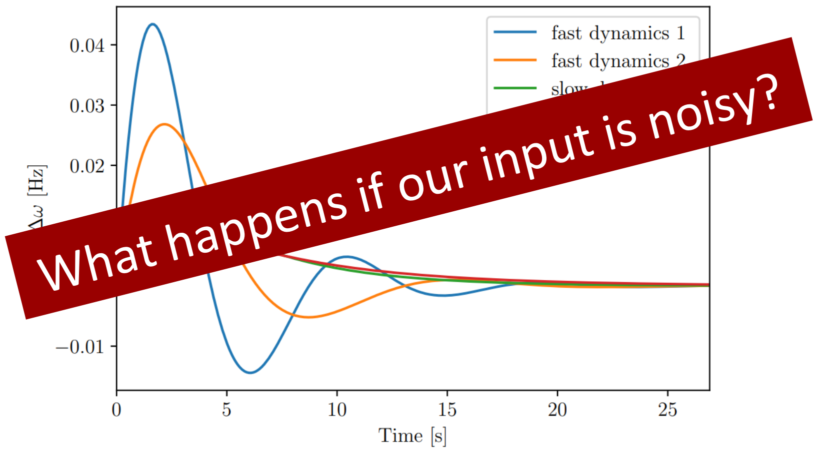
PINN and BPINN maintain good performance

S. Stock, J. Stiasny, D. Babazadeh, C. Becker, S. Chatzivasileiadis, Bayesian Physics-Informed Neural Networks for Robust System Identification of Power Systems. https://arxiv.org/abs/2212.11911

Bayesian Physics-Informed Neural Networks (BPINNs)

- Example:
  - Receive **frequency** time series as input
  - What is the **inertia** and **damping** coefficient of the system?

Input



	Inertia Estim. Error (%)	Damping Estim. Error (%)
SINDy	3.80%	0.14%
PINN	0.34%	0.84%
BPINN	1.20% ± 9.26%	0.01% ± 0.011%

Noise = 5%

	Inertia Estim. Error (%)	Damping Estim. Error (%)
SINDy	37.88%	9.67%
PINN	0.41%	3.99%
BPINN	2.03% ± 21.59%	0.02% ± 0.011%

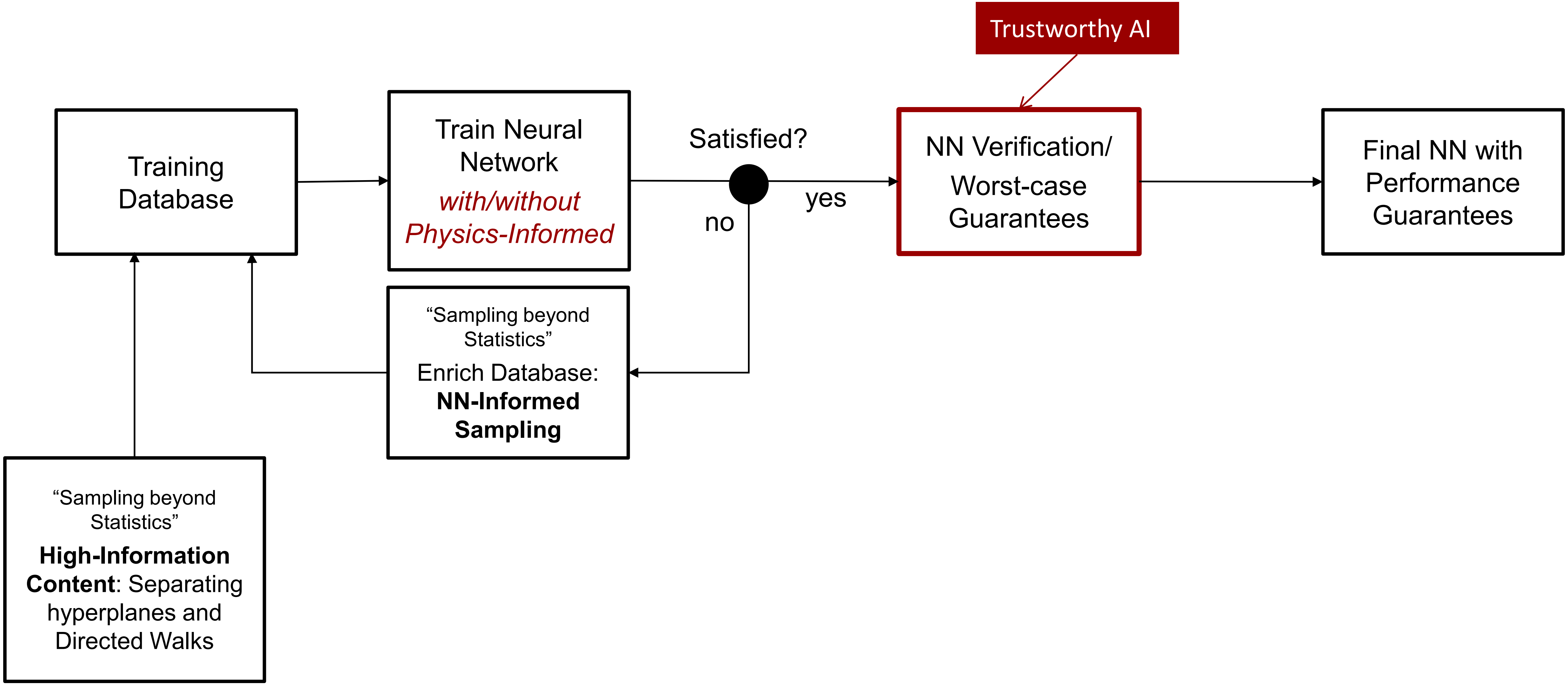
If there is noise, **SINDy** results to: **10x-90x larger error**

PINN and BPINN maintain good performance

BPINN is the only with confidence interval

S. Stock, J. Stiasny, D. Babazadeh, C. Becker, S. Chatzivasileiadis, Bayesian Physics-Informed Neural Networks for Robust System Identification of Power Systems. https://arxiv.org/abs/2212.11911

Closing the Loop: Trustworthy ML for Power Systems



Neural Network Verification

for classification NNs in Power Systems

A. Venzke, S. Chatzivasileiadis. Verification of Neural Network Behaviour: Formal Guarantees for Power System Applications. In *IEEE Transactions on Smart Grid*, vol. 12, no. 1, pp. 383-397, Jan. 2021, 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

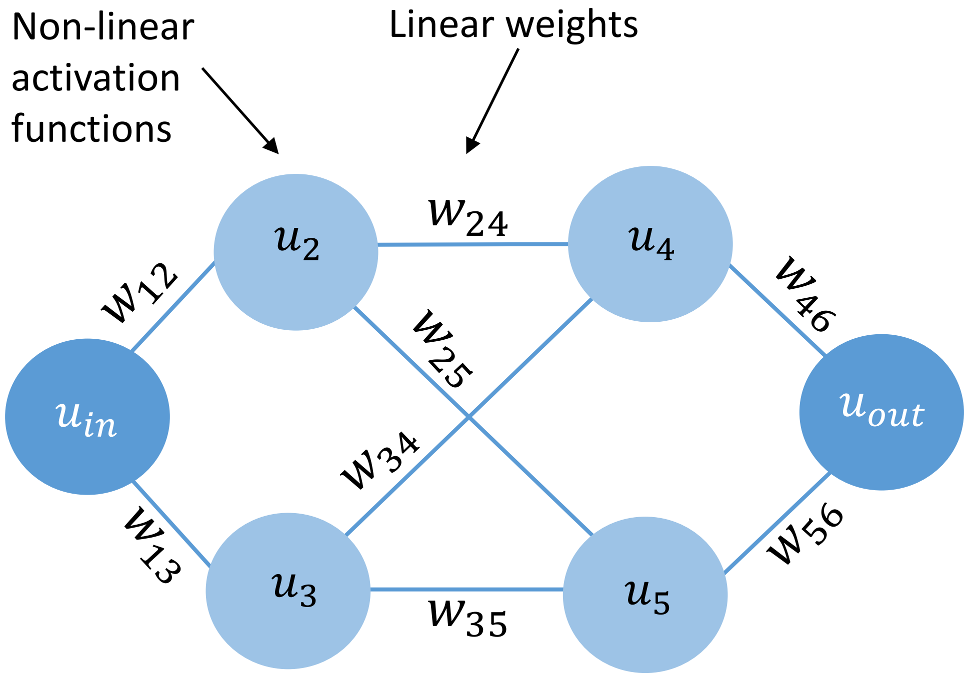
Neural Network Verification: HOW?

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

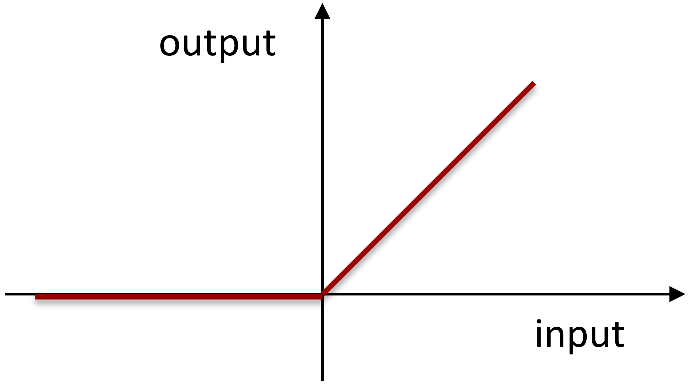
2. Formulate an **optimization** problem (MILP) and solve it → certificate for NN behavior

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

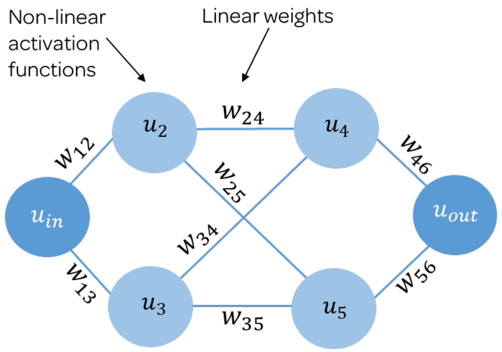
From Neural Networks to Mixed-Integer Linear Programming



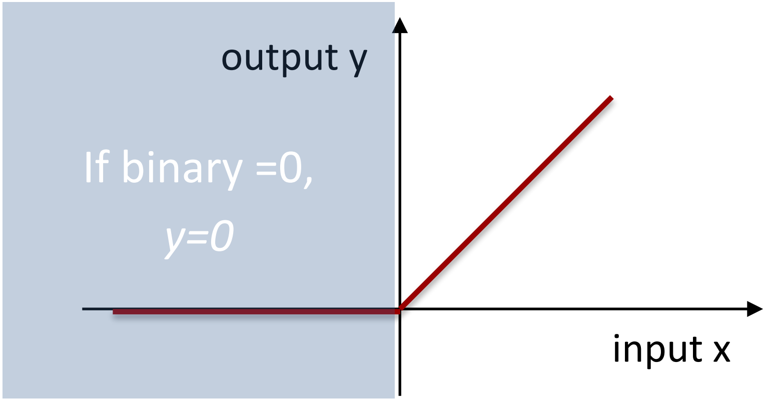
- Most usual activation function: ReLU
- **ReLU:** Rectifier Linear Unit



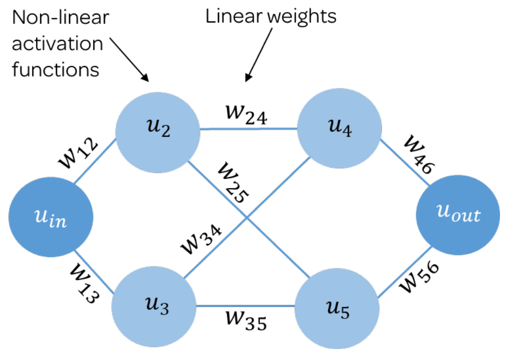
From Neural Networks to Mixed-Integer Linear Programming



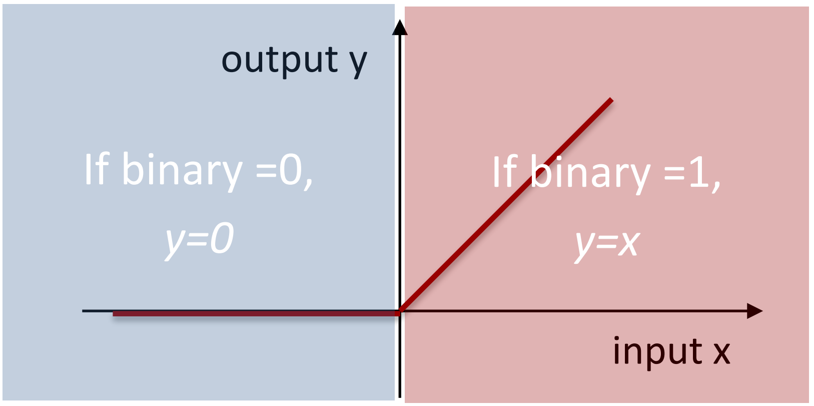
1. But **ReLU** can be transformed to a **piecewise linear function with binaries**



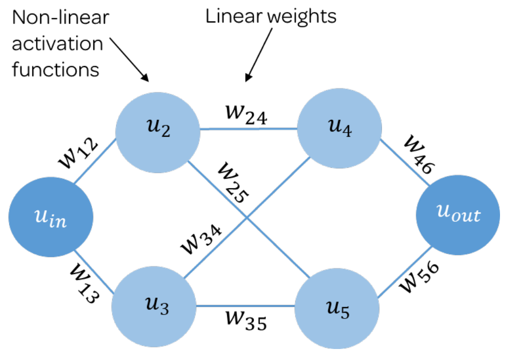
From Neural Networks to Mixed-Integer Linear Programming



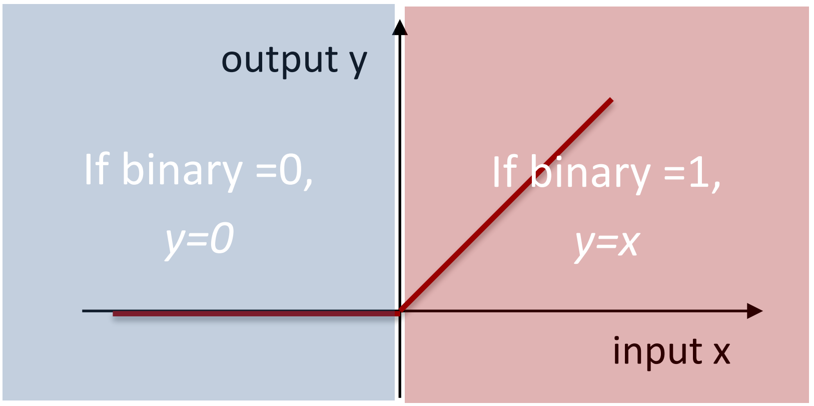
1. But **ReLU** can be transformed to a **piecewise linear function with binaries**



From Neural Networks to Mixed-Integer Linear Programming



1. But **ReLU** can be transformed to a **piecewise linear function with binaries**



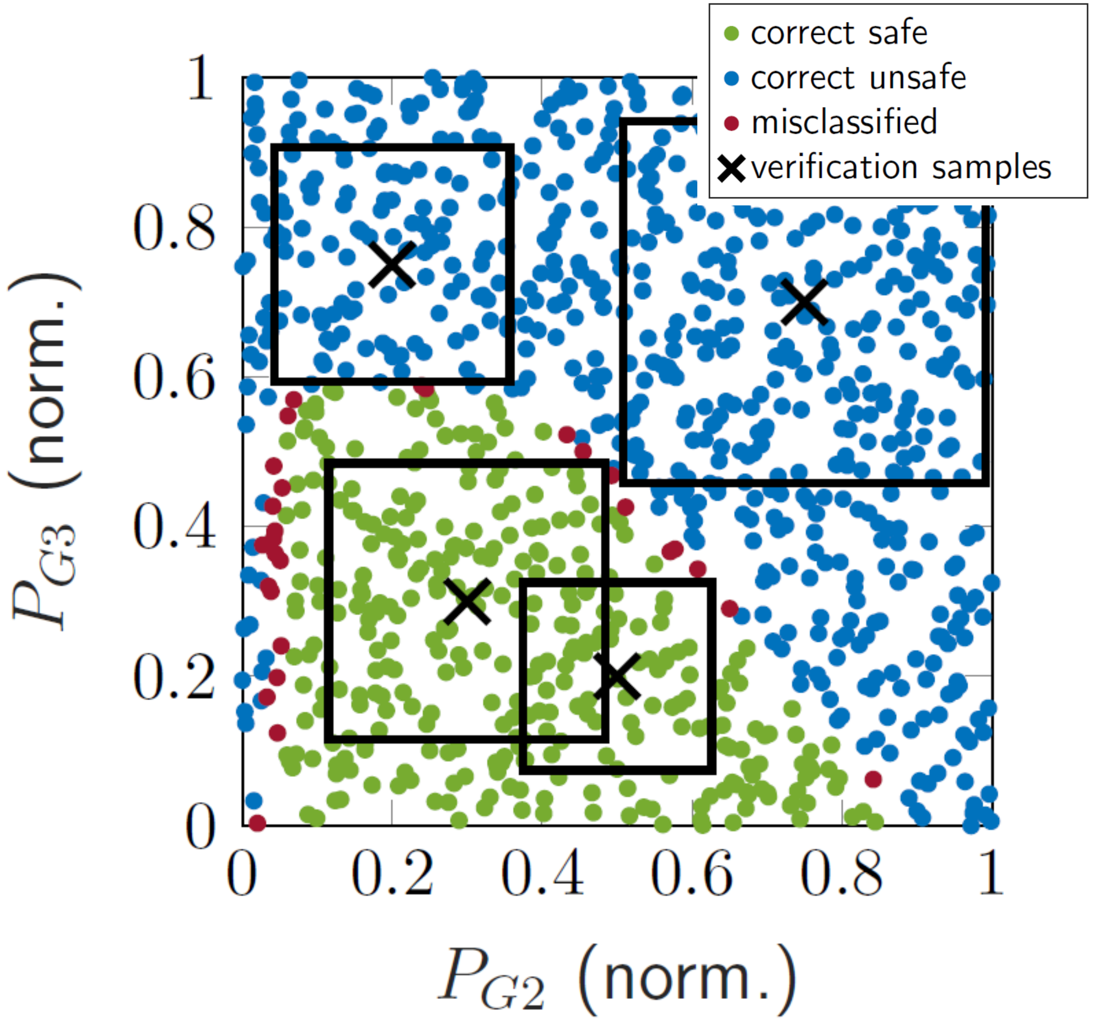
$$y = \max(0, x)$$

ReLU in a NN: $u_j = \max\left(0, w_{ij}u_i + b_i\right)$

2. I can encode all operations of a Neural Network to a system of linear equations with continuous and binary variables

3. I can **integrate** all information encoded in a **neural network inside a -** Mixed-Integed Linear optimization Program - **MILP**

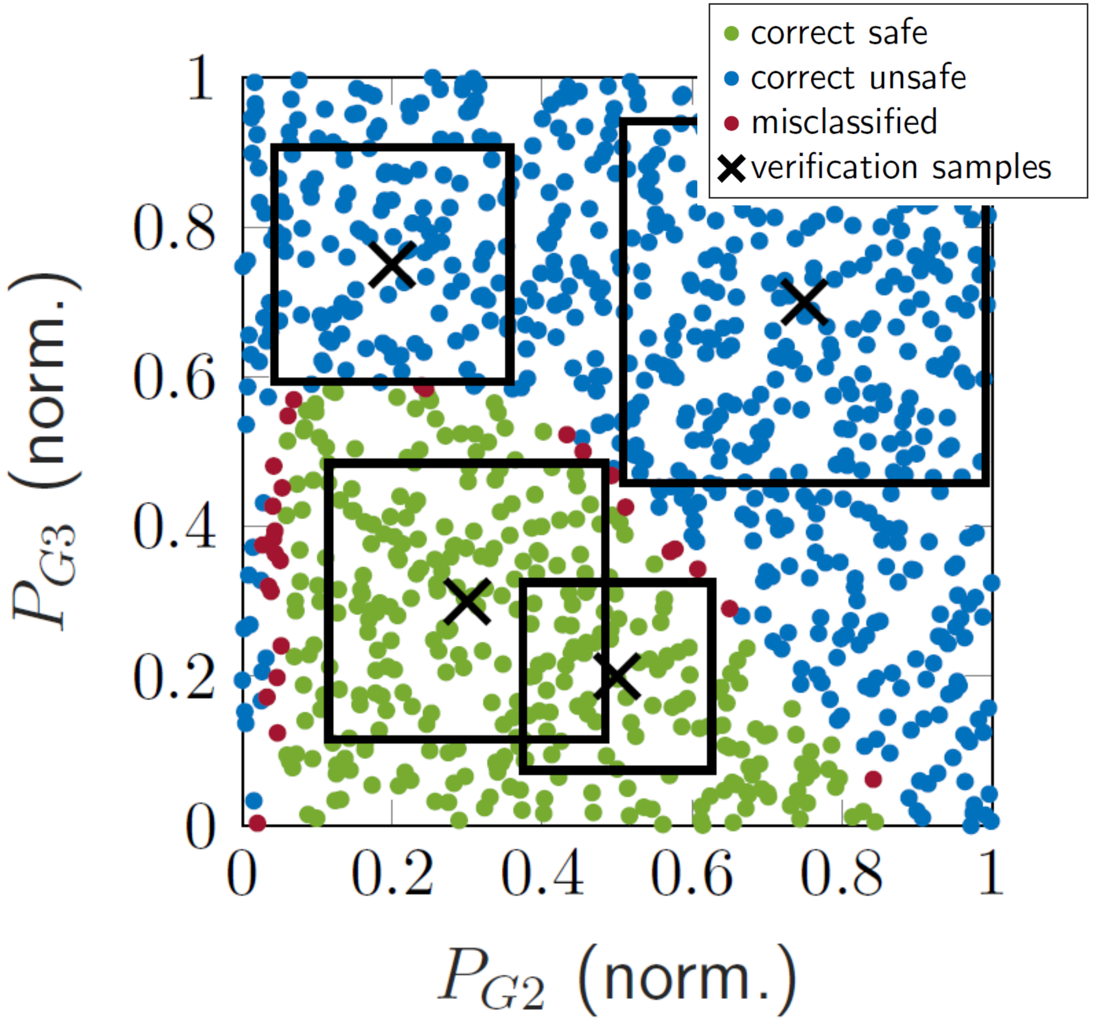
Certify the output for a continuous range of inputs



1. We assume a given input x_{ref} with classification "safe"

A. Venzke, S. Chatzivasileiadis. Verification of Neural Network Behaviour: Formal Guarantees for Power System Applications. *IEEE Transactions on Smart Grid*, Jan. 2021. https://arxiv.org/pdf/1910.01624.pdf

Certify the output for a continuous range of inputs



1. We assume a given input x_{ref} with classification “safe”
2. Solve optimization problem: **Does classification change for *any* input within distance ε from x_{ref}?**
3. If not, then **I can certify** that my neural network will classify the whole continuous region as “safe”
4. I can repeat this for other regions and different classifications

A. Venzke, S. Chatzivasileiadis. Verification of Neural Network Behaviour: Formal Guarantees for Power System Applications. *IEEE Transactions on Smart Grid*, Jan. 2021. https://arxiv.org/pdf/1910.01624.pdf

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.

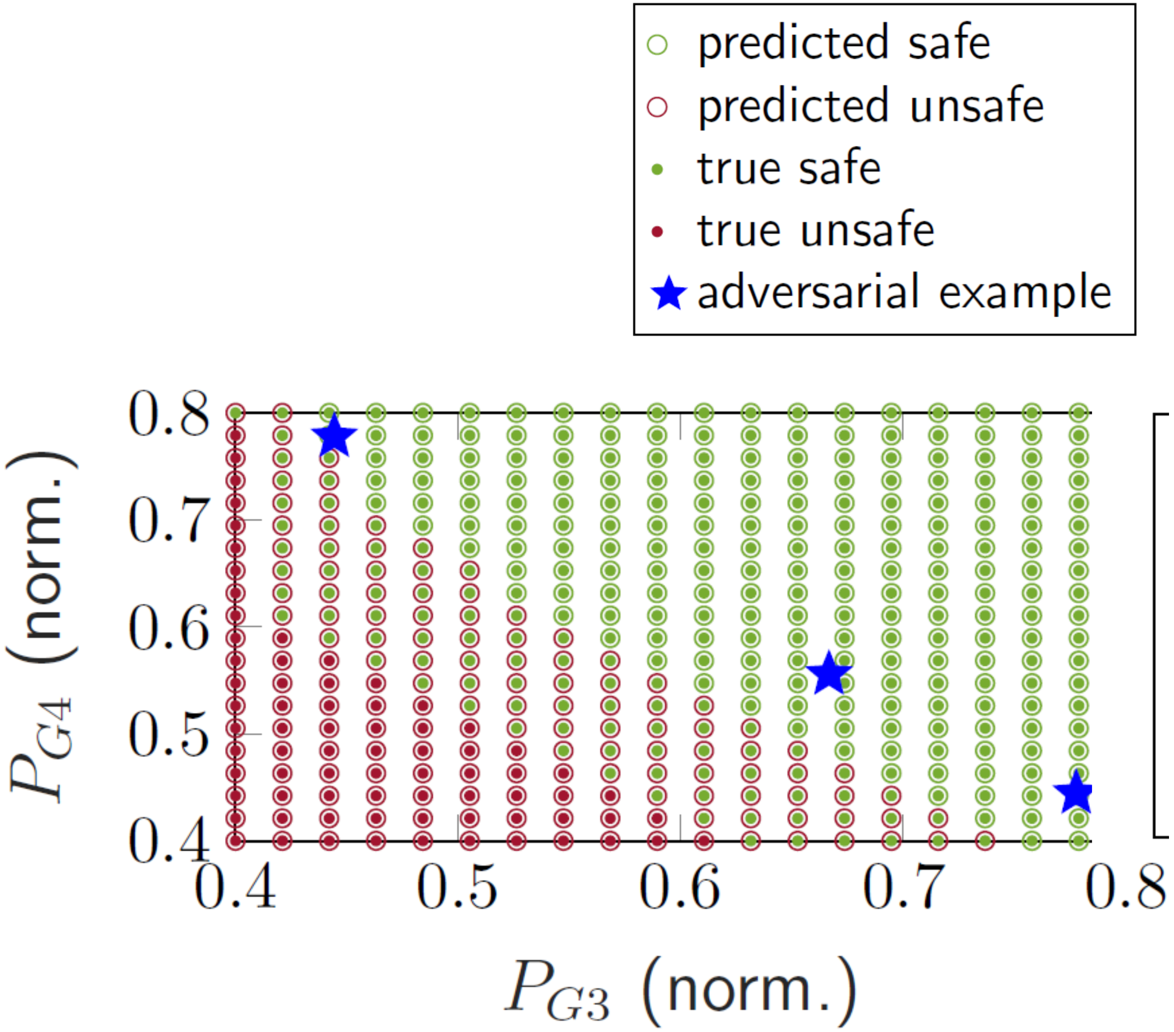
- Adversarial examples exist in many (deep) learning applications
- Major barrier for adoption of machine learning techniques in safety-critical systems!

Systematically identify adversarial examples

1. We assume a given input x_{ref} with classification "safe"

2. Solve optimization problem: What is the minimum distance ε for which the classification changes to "unsafe"

3. This point either is on the other side of the classification boundary (correct classification) **or is an adversarial point.**



Provable Worst-case Guarantees

Venzke, G. Qu, S. Low, S. Chatzivasileiadis, Learning Optimal Power Flow: Worst-case Guarantees for Neural Networks. Best Student Paper Award at IEEE SmartGridComm 2020. https://arxiv.org/pdf/2006.11029.pdf

S. Chevalier, S. Chatzivasileiadis. Global Performance Guarantees for Neural Network Models of AC Power Flow. https://arxiv.org/pdf/2211.07125.pdf

R. Nellikkath, S. Chatzivasileiadis. Minimizing Worst-Case Violations of Neural Networks. https://arxiv.org/pdf/2212.10930.pdf

Key Enabler:
our ability to represent the underlying ground truth

Main idea:

- Take advantage of the ground truth representation we have, i.e. the power system models
- Measure the performance of the NN against the ground truth → here the result of an OPF
  - Does the NN output violate constraints?
  - How close is the NN output to the optimal point?
- **Determine the worst-case performance**
  - **Across the continuous input domain**
  - No Sampling
  - Instead, we MILP or MINLP

Quick Reminder: DC Optimal Power Flow

- **Objective**: find the minimum cost generation dispatch
- **Input**: Varying load demand at different nodes

- Considered constant: generator costs; system topology

$$
\begin{aligned}
\min_{\mathbf{p}_\mathrm{g},\boldsymbol{\theta}} \quad & \mathbf{c}^T \mathbf{p}_\mathrm{g} && \text{Minimizes generation cost} \\
\text{s.t.} \quad & \mathrm{M}_\mathrm{g}\mathbf{p}_\mathrm{g} - \mathrm{M}_\mathrm{d}\mathbf{p}_\mathrm{d} = \mathrm{B}_\mathrm{bus}\boldsymbol{\theta} && \text{Nodal power balance} \\
& -\mathbf{p}_\mathrm{line}^\mathrm{max} \leq \mathrm{B}_\mathrm{line}\boldsymbol{\theta} \leq \mathbf{p}_\mathrm{line}^\mathrm{max} && \text{Transmission line limits} \\
& \mathbf{p}_\mathrm{g}^\mathrm{min} \leq \mathbf{p}_\mathrm{g} \leq \mathbf{p}_\mathrm{g}^\mathrm{max} && \text{Generator limits}
\end{aligned}
$$

Quick Reminder: DC Optimal Power Flow

- **Objective**: find the minimum cost generation dispatch
- **Input**: Varying load demand at different nodes

- Considered constant: generator costs; system topology

Several recent approaches in the literature that apply Neural Networks for solving the DC-OPF

- Demonstrate up to **100x speedup**
- But **no performance guarantees**

$$
\begin{aligned}
\min_{\mathbf{p}_{\mathrm{g}},\boldsymbol{\theta}} \quad & \mathbf{c}^T \mathbf{p}_{\mathrm{g}} && \text{Minimizes generation cost} \\
\text{s.t.} \quad & \mathbf{M}_{\mathrm{g}}\mathbf{p}_{\mathrm{g}} - \mathbf{M}_{\mathrm{d}}\mathbf{p}_{\mathrm{d}} = \mathbf{B}_{\mathrm{bus}}\boldsymbol{\theta} && \text{Nodal power balance} \\
& -\mathbf{p}_{\mathrm{line}}^{\mathrm{max}} \leq \mathbf{B}_{\mathrm{line}}\boldsymbol{\theta} \leq \mathbf{p}_{\mathrm{line}}^{\mathrm{max}} && \text{Transmission line limits} \\
& \mathbf{p}_{\mathrm{g}}^{\mathrm{min}} \leq \mathbf{p}_{\mathrm{g}} \leq \mathbf{p}_{\mathrm{g}}^{\mathrm{max}} && \text{Generator limits}
\end{aligned}
$$

Part I: Maximum limit-violations

1. Maximum violation of generator limits

$$\nu_{\mathrm{g}} = \max(\hat{\mathbf{p}}_{\mathrm{g}} - \mathbf{p}_{\mathrm{g}}^{\mathsf{max}}, \mathbf{p}_{\mathrm{g}}^{\mathsf{min}} - \hat{\mathbf{p}}_{\mathrm{g}}, \mathbf{0})$$

$$\begin{aligned} \max \quad & \nu_{\mathrm{g}} \\ \text{s.t.} \quad & \mathbf{A}_{\mathrm{d}}\mathbf{p}_{\mathrm{d}} \leq \mathbf{b}_{\mathrm{d}} && \text{Convex polytope as input domain } \mathcal{D} \\ & \hat{\mathbf{p}}_{\mathrm{g}} = NN(\mathbf{p}_{\mathrm{d}}) && \text{Mixed-integer reformulation of trained NN} \end{aligned}$$

Example:

$$0.6\, \boldsymbol{p}_{\mathrm{d}}^{\mathsf{max}} \leq \boldsymbol{p}_{\mathrm{d}} \leq 1.0\, \boldsymbol{p}_{\mathrm{d}}^{\mathsf{max}}$$

Part I: Maximum limit-violations

1. Maximum violation of generator limits

$$\nu_{\mathrm{g}} = \max(\hat{\mathbf{p}}_{\mathrm{g}} - \mathbf{p}_{\mathrm{g}}^{\max}, \mathbf{p}_{\mathrm{g}}^{\min} - \hat{\mathbf{p}}_{\mathrm{g}}, \mathbf{0})$$

$\max$	ν_{g}	
s.t.	$\mathbf{A}_{\mathrm{d}}\mathbf{p}_{\mathrm{d}} \leq \mathbf{b}_{\mathrm{d}}$	Convex polytope as input domain $\mathcal{D}$
	$\hat{\mathbf{p}}_{\mathrm{g}} = NN(\mathbf{p}_{\mathrm{d}})$	Mixed-integer reformulation of trained NN

Example:

$$0.6\, \boldsymbol{p}_{\mathrm{d}}^{\max} \leq \boldsymbol{p}_{\mathrm{d}} \leq 1.0\, \boldsymbol{p}_{\mathrm{d}}^{\max}$$

2. Maximum violation of line limits

$$\nu_{\mathrm{line}} = \max(\underbrace{|\mathbf{B}_{\mathrm{line}}\tilde{\mathbf{B}}_{\mathrm{bus}}^{-1}(\mathbf{M}_{\mathrm{g}}\hat{\mathbf{p}}_{\mathrm{g}} - \mathbf{M}_{\mathrm{d}}\mathbf{p}_{\mathrm{d}})^{\mathrm{nsb}}|}_{\text{Line flow equations for DC-OPF based on PTDFs}} - \mathbf{p}_{\mathrm{line}}^{\max}, \mathbf{0})$$

$\max$	ν_{line}	
s.t.	$\mathbf{A}_{\mathrm{d}}\mathbf{p}_{\mathrm{d}} \leq \mathbf{b}_{\mathrm{d}}$	Convex polytope as input domain $\mathcal{D}$
	$\hat{\mathbf{p}}_{\mathrm{g}} = NN(\mathbf{p}_{\mathrm{d}})$	Mixed-integer reformulation of trained NN

Worst violation over the **whole training dataset** (training+test set)

Our algorithm: **provable** worst-case guarantee over the **whole input domain**

	Empirical lower bound		Exact worst-case guarantee	
Test cases	ν_{g} (MW)	ν_{line} (MW)	ν_{g} (MW)	ν_{line} (MW)
case9				
case30				
case39				
case57				
case118				
case162				
case300				

ν_{g} Maximum violation of generator limits

ν_{line} Maximum violation of line limits

Worst violation over the **whole training dataset** (training+test set)

Our algorithm: **provable** worst-case guarantee over the **whole input domain**

	Empirical lower bound		Exact worst-case guarantee	
Test cases	ν_g (MW)	ν_{line} (MW)	ν_g (MW)	ν_{line} (MW)
case9	2.5	1.8	2.8	1.9
case30	1.7	0.6	3.6	3.1
case39	51.9	37.2	270.6	120.0
case57	4.2	0.0	23.7	0.0
case118	149.4	15.6	997.8	510.8
case162	228.0	180.0	1563.3	974.1
case300	474.5	692.7	3658.5	3449.3

ν_g Maximum violation of generator limits

ν_{line} Maximum violation of line limits

Over the whole input domain **violations can be much larger** (here ~7x) compared to what has been estimated empirically on the dataset

Worst violation over the **whole training dataset** (training+test set)

New algorithm: **provable** worst-case guarantee over the **whole input domain**

Test cases	Empirical lower bound ν_g (MW)	Empirical lower bound ν_{line} (MW)	Exact worst-case guarantee ν_g (MW)	Exact worst-case guarantee ν_{line} (MW)
case9	2.5	1.8	2.8	1.9
case30	1.7	0.6	3.6	3.1
case39	51.9	37.2	270.6	120.0
case57	4.2	0.0	23.7	$\boxed{0.0}$
case118	149.4	15.6	997.8	510.8
case162	228.0	180.0	1563.3	974.1
case300	474.5	692.7	3658.5	3449.3

ν_g Maximum violation of generator limits

ν_{line} Maximum violation of line limits

We can now provide **guarantees that no NN output will violate the line limits** over the whole input domain

How can we reduce the worst-case violations?

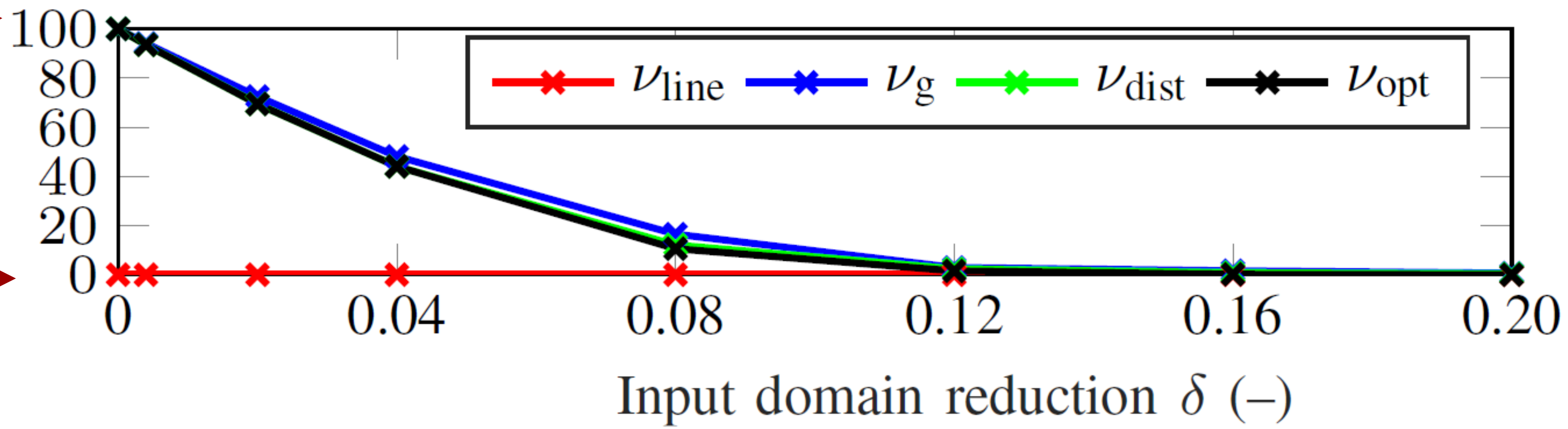
- From our experiments with DC-OPF in 7 different test power systems, we observed that the **worst-case violations occur at the boundary of the input domain**

- Possible solution:
  1. Train on a larger input domain
  2. Use the NN on a subdomain of the original training input

Reducing the worst-case violations

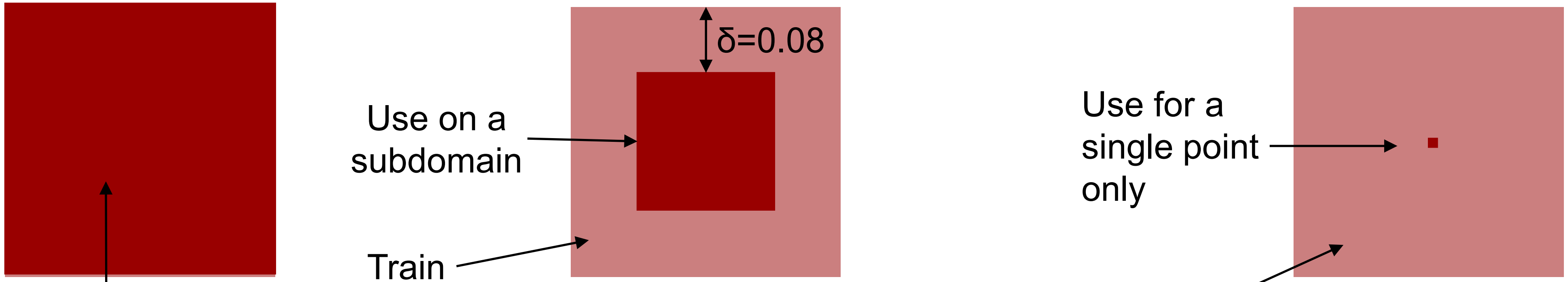
Max violation

No violation



Training Dataset:

Load Domain [60%-100%]



Train and use on the same domain

Use on a subdomain

Train

Use for a single point only

Train

Neural Networks for non-linear problems: How can we determine the worst-case violations?

- DC-OPF: convex linear problem
  - Determining the worst-case violations : MILP
- AC-OPF: non-linear
  - Determining the worst-case violations : MIQP = challenge!

*"we were **unable to compute** the **worst-case line flow** constraint violation since the **MIQCQP** problem could **not be solved** to zero optimality gap **within 5 hours.** This highlights the computational challenges associated with the extraction of the worst-case guarantees for AC-OPF..."*

R. Nellikkath and S. Chatzivasileiadis, "Physics-informed neural networks for AC optimal power flow," Electric Power Systems Research, 2022. (presented at PSCC)

S. Chevalier, S. Chatzivasileiadis. Global Performance Guarantees for Neural Network Models of AC Power Flow. https://arxiv.org/pdf/2211.07125.pdf

Neural Networks for non-linear problems: How can we determine the worst-case violations?

- AC-OPF: Determining the worst-case violations = MIQP

What can we do?

1. Use **SDP** to relax the binaries and the quadratic terms of the MIQP → too loose

2. Use **Sherali-Adams cuts** to tighten the relaxation → tight but too many constraints (N^2)

3. Sequential Targeted Tightening = iterative tightening

S. Chevalier, S. Chatzivasileiadis. Global Performance Guarantees for Neural Network Models of AC Power Flow. https://arxiv.org/pdf/2211.07125.pdf

Neural Networks for non-linear problems: How can we determine the worst-case violations?

- AC-OPF: Determining the worst-case violations = MIQP

What can we do?

1. Use **SDP** to relax the binaries and the quadratic terms of the MIQP → too loose
2. Use **Sherali-Adams cuts** to tighten the relaxation → tight but too many constraints (N^2)
3. Sequential Targeted Tightening = iterative tightening

- K. D. Dvijotham, R. Stanforth, S. Gowal, C. Qin, S. De, and P. Kohli, "Efficient neural network verification with exactness characterization," in Proceedings of The 35th Uncertainty in Artificial Intelligence Conference, 2020.
- Z. Ma and S. Sojoudi, "Strengthened sdp verification of neural network robustness via non-convex cuts," arXiv preprint arXiv:2010.08603, 2020.
- J. Lan, Y. Zheng, and A. Lomuscio, "Tight neural network verification via semidefinite relaxations and linear reformulations," in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 36, no. 7, 2022, pp. 7272–7280.
- S. Fattahi, M. Ashraphijuo, J. Lavaei, and A. Atamturk, "Conic ¨ relaxations of the unit commitment problem," Energy, vol. 134, pp. 1079–1095, 2017.
- S. Gopinath, H. Hijazi, T. Weisser, H. Nagarajan, M. Yetkin, K. Sundar, and R. Bent, "Proving global optimality of acopf solutions," Electric Power Systems Research, vol. 189, p. 106688, 2020

Our NN verification problem: includes the ground truth in the optimization

This approach: exploits iterative tightening to not only query the NN, but to also tighten the SDP variable associated with the ground truth

S. Chevalier, S. Chatzivasileiadis. Global Performance Guarantees for Neural Network Models of AC Power Flow. https://arxiv.org/pdf/2211.07125.pdf

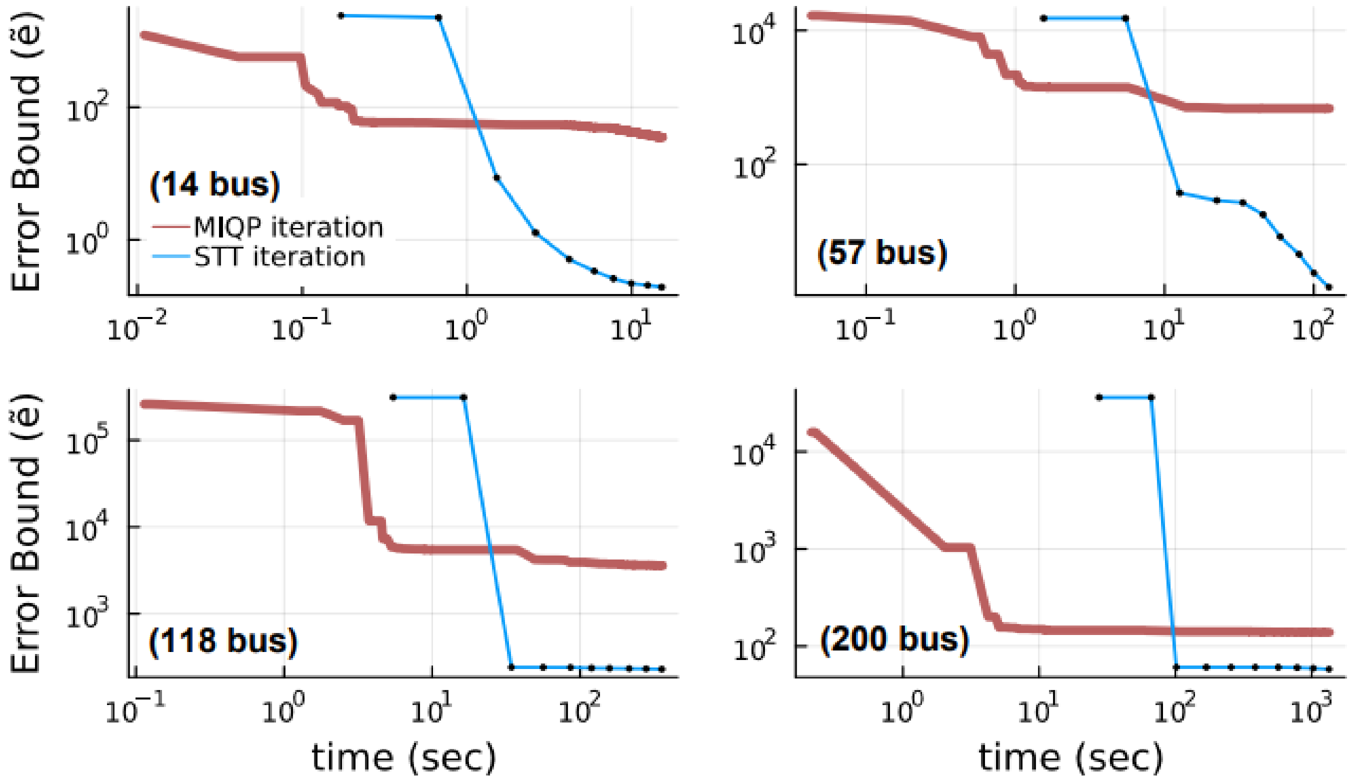
Neural Networks for non-linear problems: How can we determine the worst-case violations?

- AC-OPF: Determining the worst-case violations = MIQP

What can we do?

1. Use **SDP** to relax the binaries and the quadratic terms of the MIQP → too loose
2. Use **Sherali-Adams cuts** to tighten the relaxation → tight but too many constraints (N^2)
3. Sequential Targeted Tightening = iterative tightening

Gurobi MIQP vs Sequential Targeted Tightening (STT)



STT achieves much tighter bounds

Tight bounds = Guarantees (loose bounds = no guarantees)

S. Chevalier, S. Chatzivasileiadis. Global Performance Guarantees for Neural Network Models of AC Power Flow. https://arxiv.org/pdf/2211.07125.pdf

Worst-case Violations: What is the next natural step?

Integrate the worst-case violations *inside* the neural network training procedure

Our ”Holy Grail”: Design a **Neural Network training procedure** that:

- produces a Neural Network with best average performance,
- ***and*** delivers guarantees about its worst-case performance

Worst-case Violations: What is the next natural step?

Integrate the worst-case violations *inside* the neural network training procedure

Our ”Holy Grail”: Design a Neural Network training procedure that:

- produces a Neural Network with best average performance,
- ***and*** delivers guarantees about its worst-case performance

(Random) Example of an imaginary final message:

- ”Neural Network Training finished. Accuracy 99.2%. Worst-case violation of critical constraints: 10%.”

Wouldn’t that create a good level of trust for applying NNs on any safety-critical system?

Extends beyond power systems → drones, air-traffic control, robots, control of inverters, and others

How can we integrate worst-case violations in NN training?

- Standard NN training

$$\min_{\mathbf{w},\mathbf{b}} \mathcal{L}_0 \equiv \min_{\mathbf{w},\mathbf{b}} \frac{1}{N} \sum_i \| x_i - \hat{x}_i \|$$

How can we integrate worst-case violations in NN training?

- Standard NN training

$$\min_{\mathbf{w},\mathbf{b}} \mathcal{L}_0 \equiv \min_{\mathbf{w},\mathbf{b}} \frac{1}{N} \sum_i \| x_i - \hat{x}_i \|$$

- NN training which penalizes constraint violations
  – Reduces the violations for the training dataset

    See Fioretto, Mak, Van Hentenryck, AAAI, 2020, and others

$$\min_{\mathbf{w},\mathbf{b}} \quad \Lambda_0 \mathcal{L}_0 + \Lambda_p \mathcal{L}_p$$

e.g. $\mathcal{L}_p = v_g = (p_g - p_g^{max})$ *for generator constraint violations*

How can we integrate worst-case violations in NN training?

- Standard NN training

$$\min_{\mathbf{w},\mathbf{b}} \mathcal{L}_0 \equiv \min_{\mathbf{w},\mathbf{b}} \frac{1}{N} \sum_i \| x_i - \hat{x}_i \|$$

- NN training which penalizes constraint violations
  - Reduces the violations for the training dataset

    See Fioretto, Mak, Van Hentenryck, AAAI, 2020, and others

$$\min_{\mathbf{w},\mathbf{b}} \quad \Lambda_0 \mathcal{L}_0 + \Lambda_p \mathcal{L}_p$$

e.g. $\mathcal{L}_p = v_g = (p_g - p_g^{max})$ *for generator constraint violations*

- **NN training which penalizes worst-case violations**
  - **Worst-case violations might be on datapoints that do not belong to the training dataset.** And we might just discover it when we deploy the NN in a real application
    - this is a major fear of any power system operator (and a main barrier for the NNs in safety-critical applications)

$$\min_{\mathbf{w},\mathbf{b}} \quad \Lambda_0 \mathcal{L}_0 + \Lambda_w \mathcal{L}_{wc}$$

$$\mathcal{L}_{wc} = \max_{\mathbf{D}} \quad v_g$$

Hard bilevel optimization problem

1. Lower level is a MILP
2. **The MILP must be differentiable** so that the NN training can backpropagate

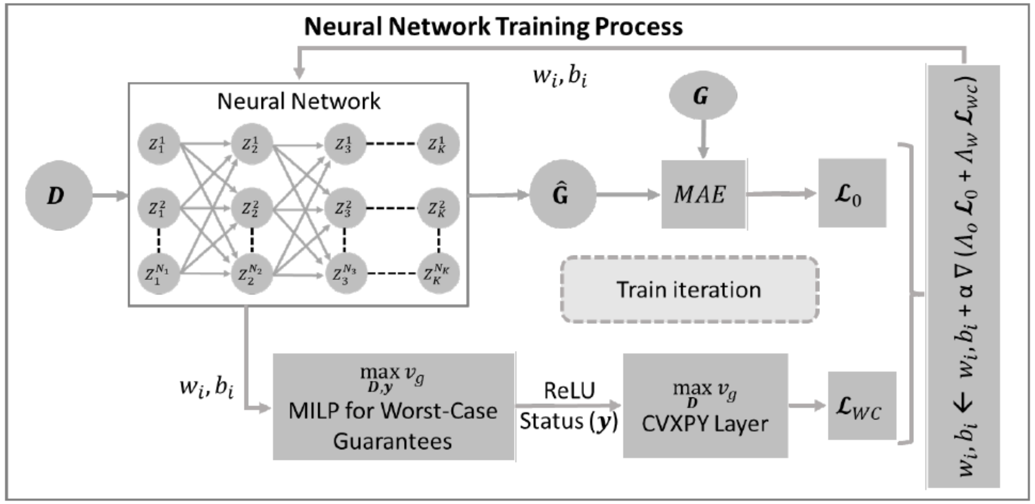
Some thoughts

on how to design an NN training that minimizes worst-case violations

1. Fix the binaries
   - Arbitrary assumption (but it works): for small perturbations of weights&biases, binaries remain constant
   - Solve the lower level MILP by itself, find the binary values for the max constraint violation and fix them

2. MILP is converted to an LP → it is now differentiable

3. Cast it as a diffentiable optimization layer (we use CVXPY) → NN training can now backpropagate through it

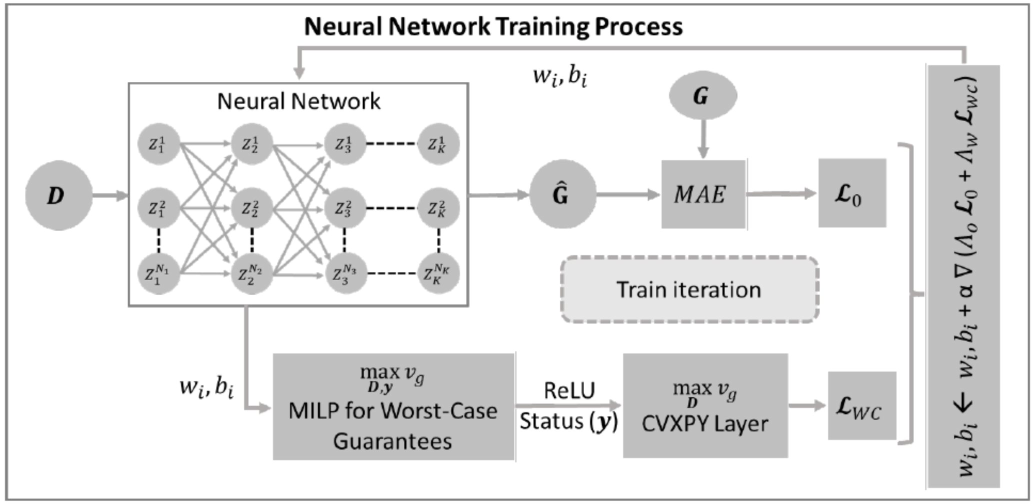


R. Nellikkath, S. Chatzivasileiadis. Minimizing Worst-Case Violations of Neural Networks. https://arxiv.org/pdf/2212.10930.pdf

Some thoughts

on how to design an NN training that minimizes worst-case violations

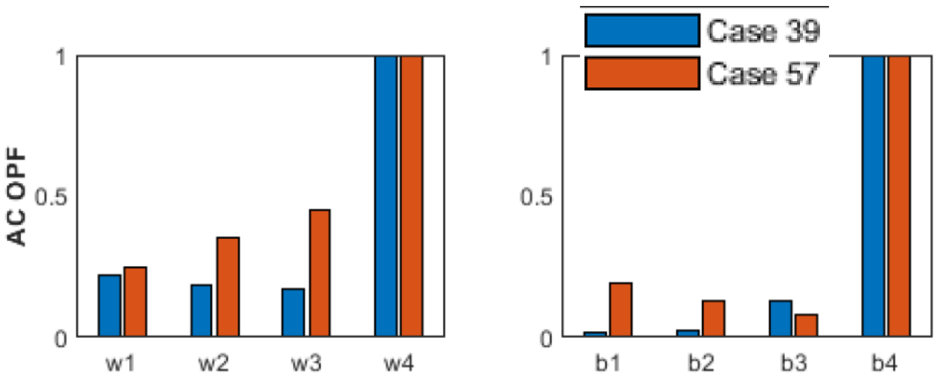
1. Fix the binaries
   - Arbitrary assumption (but it works): for small perturbations of weights&biases, binaries remain constant
   - Solve the lower level MILP by itself, find the binary values for the max constraint violation and fix them



2. MILP is converted to an LP → it is now differentiable

3. Cast it as a diffentiable optimization layer (we use CVXPY) → NN training can now backpropagate through it

4. Reduce complexity: reduce the weights and biases to adjust → **w, b of last layer had the largest impact**



Derivatives of weights and biases on each of the 4 layers w.r.t. worst-case violations

R. Nellikkath, S. Chatzivasileiadis. Minimizing Worst-Case Violations of Neural Networks.. https://arxiv.org/pdf/2212.10930.pdf

DTU

Test Cases		MAE (%)	Worst case Guarantees Generation violation w.r.t. max loading
case39	NN		
	GenNN		
	WCNN		
case57	NN		
	GenNN		
	WCNN		
case118	NN		
	GenNN		
	WCNN		
case162	NN		
	GenNN		
	WCNN		

AC-OPF

NN: standard NN

GenNN: penalizing violations in the Loss Function

WCNN: our approach; penalizing *worst-case* violations

R. Nellikkath, S. Chatzivasileiadis. Minimizing Worst-Case Violations of Neural Networks.. https://arxiv.org/pdf/2212.10930.pdf

DTU

Test Cases		MAE (%)	Worst case Guarantees Generation violation w.r.t. max loading
case39	NN	0.56 %	0.67 %
	GenNN	0.55 %	0.67 %
	WCNN	0.47 %	0.00 %
case57	NN	1.02 %	0.65 %
	GenNN	1.01 %	0.67 %
	WCNN	1.00 %	0.29 %
case118	NN	0.42 %	204.60 %
	GenNN	0.42 %	213.80 %
	WCNN	0.42 %	109.83 %
case162	NN	1.10 %	184.30 %
	GenNN	1.06 %	181.52 %
	WCNN	1.06 %	142.35 %

NN: standard NN

GenNN: penalizing violations in the Loss Function

WCNN: our approach; penalizing *worst-case* violations

1. Good average performance and minimum worst-case violations are **not necessarily competing objectives**

2. Surprising: **WCNN** not only eliminates all violations, but manages to find a lower minimum for the average performance as well

R. Nellikkath, S. Chatzivasileiadis. Minimizing Worst-Case Violations of Neural Networks.. https://arxiv.org/pdf/2212.10930.pdf

DTU

Test Cases		MAE (%)	Worst case Guarantees Generation violation w.r.t. max loading
case39	NN	0.56 %	0.67 %
	GenNN	0.55 %	0.67 %
	WCNN	0.47 %	0.00 %
case57	NN	1.02 %	0.65 %
	GenNN	1.01 %	0.67 %
	WCNN	1.00 %	0.29 %
case118	NN	0.42 %	204.60 %
	GenNN	0.42 %	213.80 %
	WCNN	0.42 %	109.83 %
case162	NN	1.10 %	184.30 %
	GenNN	1.06 %	181.52 %
	WCNN	1.06 %	142.35 %

R. Nellikkath, S. Chatzivasileiadis. Minimizing Worst-Case Violations of Neural Networks.. https://arxiv.org/pdf/2212.10930.pdf

NN: standard NN

GenNN: penalizing violations in the Loss Function

WCNN: our approach; penalizing *worst-case* violations

1. For larger systems, the worst-case violations are large
2. **WCNN manages to reduce them by 50%**
3. Reducing Worst-Case Violations does not affect average performance!

A lot more work is needed to improve scalability and performance!

Thoughts on Minimizing Worst-Case Violations of Neural Networks

- For the first time, create a NN training procedure that can not only determine but also reduce the worst-case violations *during training*

- Why does it work?
  - Because we have a physical model of the process that our NN emulates

- What are the challenges?
  - Computational performance → it takes too much time
  - Scalability → how can we verify larger neural networks (or consider more complex ground truth representations)
  - How can we achieve the zero MILP gap = obtain the performance guarantee?

- Solutions?
  - …

Wrap-up

1. **Sampling beyond statistics** can yield high quality training databases with smaller amounts of data

2. **Physics-informed neural networks** exploit the underlying physics in the training procedure.

3. **Neural network verification** builds the missing trust; necessary in safety-critical systems.

4. Combine NN verification with physics-informed (ground truth representation) → **NN training that delivers worst-case performance guarantees**

"Data-centric AI movement"
(Andrew Ng, Stanford, and others)

"Small [data] is the new big"
(IEEE Spectrum, Apr. 2022)

Exploit the prior knowledge

Some Final Thoughts

- **If we want to accelerate processes by 10x-100x-1000x we need to think differently**
  - Conventional methods reach their limits (?)
  - Could Machine Learning become the disruptive technology?

- **Neural Network Verification is an optimization problem**. Can we address its challenges?
  - **If yes, we remove barriers** for a wide range of safety-critical applications
    - Power systems, robots, self-driving cars, control of critical infrastructure, and many others

- **Can we model the ground truth? If yes, use it!**
  - Physics-Informed Neural Networks
  - Sampling Beyond Statistics
  - Neural Network Training with Worst-Case Performance Guarantees

- Federated/Distributed Learning
  - Do not need a single NN for the whole problem
  - **Let's work with "Libraries of Neural Networks"**, similar to "Libraries of Models"

- **For Power Systems: Major Challenge = Topology**
  - Solution: Graph Neural Networks?
  - Transfer Learning?

What did I not talk about

Exploring a wide range of research directions

1. Contracting Neural-Newton Solver: Derive convergence guarantees for Neural Networks that can replace conventional Newton solvers [https://arxiv.org/pdf/2106.02543.pdf , L4DC 2022]

2. Accelerating MILPs: using Decision Trees to estimate the active set and drastically reduce the number of binary variables [https://arxiv.org/pdf/2010.06344.pdf , IEEE Trans. Power Systems]

3. Interpretable Machine Learning: Direct association of the SHAP Values with the Power Transfer Distribution Factors (PTDFs) [https://arxiv.org/pdf/2209.05793.pdf , submitted]

4. Input Convex NNs for convex approximations of non-convex optimization problems [https://arxiv.org/pdf/2209.08645.pdf , submitted]

5. Physics-Informed Neural Networks for Fast Dynamic Security Assessment [https://arxiv.org/pdf/2106.13638.pdf, code: https://github.com/jbesty/PINNs_transient_stability_analysis]

and others…

Interested in a postdoc or PhD?

- Come work with us!

- Wide range of topics around ML and beyond:
  - Trustworthy Machine Learning, Physics-Informed Neural Networks, capturing intractrable constraints with NNs, and more!
  - Working with real datasets, and industry collaboration
  - Opportunities for open academic research and/or toolbox development for practical applications

- Open positions online!

- **Deadline: 31st January 2023**

- Contact: spchatz@dtu.dk

Thank you!

Spyros Chatzivasileiadis

Assoc. Prof, Head of Section

www.chatziva.com

spchatz@dtu.dk

- A. Venzke, S. Chatzivasileiadis. Verification of Neural Network Behaviour: Formal Guarantees for Power System Applications. Accepted at IEEE Trans. on Smartgrid. 2020. https://arxiv.org/pdf/1910.01624.pdf
- A. Venzke, G. Qu, S. Low, S. Chatzivasileiadis, Learning Optimal Power Flow: Worst-case Guarantees for Neural Networks. **Best Student Paper Award** at IEEE SmartGridComm 2020.[.pdf | slides | video]
- G. S. Misyris, A. Venzke, S. Chatzivasileiadis, Physics-Informed Neural Networks for Power Systems. Presented at the **Best Paper Session** of IEEE PES GM 2020. https://arxiv.org/pdf/1911.03737.pdf
- S. Chevalier, S. Chatzivasileiadis. Global Performance Guarantees for Neural Network Models of AC Power Flow. https://arxiv.org/pdf/2211.07125.pdf
- R. Nellikkath, S. Chatzivasileiadis. Minimizing Worst-Case Violations of Neural Networks. https://arxiv.org/pdf/2212.10930.pdf
- J. Stiasny, S. Chevalier, R. Nellikkath, B. Sævarsson, S. Chatzivasileiadis. **Closing the Loop: A Framework for Trustworthy Machine Learning in Power Systems**. Accepted to *2022 iREP Symposium - Bulk Power System Dynamics and Control - XI (iREP)*. Banff, Canada. 2022. [paper | code]

Article without any equations ☺

S. Chatzivasileiadis, A. Venzke, J. Stiasny and G. Misyris, **"Machine Learning in Power Systems: Is It Time to Trust It?**," in *IEEE Power and Energy Magazine*, vol. 20, no. 3, pp. 32-41, May-June 2022 [.pdf]

All publications available at:

www.chatziva.com/publications.html

Some code available at:

www.chatziva.com/downloads.html