# DYNAMIC STABILITY OF POWER GRIDS - NEW DATASETS FOR GRAPH NEURAL NETWORKS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

One of the key challenges for the success of the energy transition is the analysis of the dynamic stability of power grids. Graph Neural Networks (GNNs) are a promising method to reduce the computational effort of predicting dynamic stability of power grids, however datasets of appropriate complexity and size do not yet exist. In this paper we introduce new datasets of synthetic power grids and node-wise dynamic stability based on Monte-Carlo simulations. The datasets consist of a total of 20,000 grids instead of previously published work that has 2,000 grids. This enables the training of more complex models and can significantly increase the performance. The investigated grids have two sizes (20 and 100 nodes), which enables the application of out-of-distribution evaluation and transfer learning from a small to a large domain. Lastly, we provide several benchmark models to establish the feasibility of predicting dynamic stability from graph features. These models achieve surprisingly high performance and even generalize to out-of-distribution settings, which opens the door for future application on real power grids. All Code and Data will be made available upon publication. We invite the community to improve on our benchmark models and thus aid the energy transition with better tools.

## 1 INTRODUCTION

Increasing the share of renewable energies in total energy production is one of the key targets on the path to carbon-neutral societies. In contrast to conventional power plants, renewable energies are more decentralized, have less inertia and their production is more volatile. Those aspects pose challenges to the current power grid infrastructure, both in terms of grid expansion and stable operation with large shares of renewable energies. Classical "static" approaches of computational modeling and analysis of power grids, like load flow analysis, are no longer sufficient. In the near future, they have to be complemented by dynamical simulations, which rely on graph theory as well as differential equations. Since such dynamical simulations are computationally much more demanding, a reliable and efficient prediction of dynamical behaviour of power grids using machine learning (ML) methods might be a better alternative. Furthermore, the analysis of the decision process of ML-models trained to predict dynamic stability might lead to new unknown relations. For these purposes, we introduce new datasets that consist of synthetic models of power grids and statistical results of dynamic simulations. The task is to predict the dynamic stability of power grids, which characterizes the resilience of the power grid towards non-linear perturbations.

**Dynamic stability of power grids** We quantify dynamic stability with single-node basin stability (SNBS)(Menck et al., 2013), a probabilistic measure that captures non-linear, dynamical effects after perturbations at a single node. Crucially, SNBS is not the result of a single simulation but describes a statistical behaviour (expected value of a Bernoulli experiment), where higher values indicate more stability. Due to its probabilistic nature, all values of SNBS are $\in [0, 1]$. Furthermore, despite its name, it captures the reaction of the entire grid and hence is not a purely local property.

**Related work** Since power grids have an underlying graph structure, the recent development of graph representation learning (Bronstein et al., 2021; Hamilton, 2020) makes it possible to use machine learning to analyze power grids. There are a number of applications using Graph Neural Networks (GNNs) for different flow-related tasks: (Donon et al., 2019; Kim et al., 2019; Bolz et al.,

2019; Retiére et al., 2020; Wang et al., 2020; Owerko et al., 2020; Gama et al., 2020; Misyris et al., 2020; Liu et al., 2021). In (Nauck et al., 2022) GNNs are used to predict the dynamic stability of power grids and we extend this work by introducing datasets that have ten times as many grids to enable the training of more complex models. In this work, we show that by increasing the complexity of the GNN-models and training on a larger data set, high accuracies can be achieved, which demonstrates the feasibility of using ML to aid power grid modeling.

**Potential of our datasets** The newly introduced, large datasets deal with a very complex problem and set an exciting challenge for GNNs. Even though we applied several simplifications regarding the modelling of power grids, the task is still challenging and the datasets are an important step towards the analysis of the dynamics of power grids. By reducing some of the complexity, new methods can be explored more easily. Working with these datasets opens an opportunity for researchers in the field of graph representation learning to contribute to the energy transition.

**Our main contributions are:** First, we introduce new synthetic datasets based on dynamic modeling of power grids as new challenges for GNN models. Second, we briefly explain the underlying physics to generate the datasets and we present selected properties of the datasets. Third, we compare the performance of multiple GNN models and set benchmark performances, which include an out-of-distribution evaluation.

## 2 GENERATION OF THE DATASETS

### 2.1 PHYSICAL MODELING OF DYNAMICAL STABILITY OF POWER GRIDS

To generate the datasets, we conduct dynamical simulations. Power grids are complex systems consisting of different nodes and edges. It is not possible to consider all properties of real power grid to conduct dynamical simulations, so we rely on simplified models that are commonly used. In our case, all nodes are represented by the $2^{nd}$-order-Kuramoto model (Kuramoto, 2005; Rodrigues et al., 2016), which is also called swing equation and we use the following notation:

$$\ddot{\phi}_i = P_i - \alpha\dot{\phi}_i - \sum_j^n K_{ij}\sin(\phi_i - \phi_j), \tag{1}$$

where $\phi, \dot{\phi}, \ddot{\phi}$ denotes the voltage angle and its time derivatives. The parametrization is as follows: the injected power $P_i \in \{-1, 1\}$ where $\sum_i P_i = 0$ to guarantee power balance, the damping coefficient $\alpha = 0.1$ , the coupling matrix $K$ is based on the graphs' adjacency matrix $A$ which encodes the graph topology and we use a uniform coupling strength, that is $K_{ij} = 9A_{ij}$. Using homogeneous coupling strength can be interpreted as considering power grids that only have one type of power line and equal distances between all nodes.

To estimate SNBS we use the same approach as in Nauck et al. (2022): "[F]or every node in a graph, $M = 10,000$ samples of perturbations per node are constructed by sampling a phase and frequency deviation from a uniform distribution with $(\phi, \dot{\phi}) \in [-\pi, \pi] \times [-15, 15]$ and adding them to the synchronized state. Each such single-node perturbation serves as an initial condition of a dynamic simulation of our power grid model, [cf. equation 1]. At $t = 500$ the integration is terminated and the outcome of the Bernoulli trial is derived from the final state. A simulation outcome is referred to as *stable* if at all nodes $\dot{\phi}_i < 0.1$. Otherwise it is referred to as *unstable*. The classification threshold of $0.1$ is chosen accounting for minor deviations due to numerical noise and slow convergence rates within a finite time-horizon."

### 2.2 DESIGN CHOICES AND PROPERTIES OF THE DATASETS

We pursue several goals during the design process of the datasets. First, to obtain a somewhat realistic representation of power grids, we use the package Synthetic Networks (Schultz et al., 2014) [1] to generate topologies with basic features of power grids. The generated graphs are sparse (degree distribution has a maximum at two and an exponential tail) similar to real-world power grids. To

[1] This tool is available on Github (Schultz, 2020)

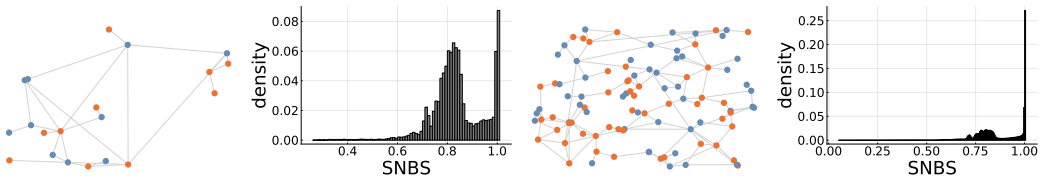


Figure 1: Examples of the power grids of the datasets and the corresponding distributions of SNBS of the entire datasets with 20 nodes (left) and 100 nodes (right). The blue color denotes sources and the orange sinks. The distributions are normalized so that bin heights sum to 1.

investigate different topological properties of differently sized grids, we generate two datasets with either 20 or 100 nodes per grid, referred to as dataset20 and dataset100. Second, we use a power grid model with reduced complexity to achieve a solvable problem. The most important simplifications are homogeneous edges, fixed magnitudes of sources/sinks and modeling all nodes by the swing equation. Third, we fine-tune parameters such as the coupling constant and perturbation strength to obtain a bi-modular shape of the SNBS distribution. Examples of each dataset as well as the distributions of SNBS are given in fig. 1. Fourth, we reduce the numerical and statistical errors to a minimum by using higher order Runge-Kutta methods with low tolerances and adaptive time steps and we conduct 10,000 simulations per node, resulting in standard errors of only $\pm 0.01$. This ensures the reliability of the results. Lastly, to enable the training of complex models, both datasets consist of 10,000 graphs.
Overall, the power grid datasets consist of the adjacency matrix and the binary injected power $P$ per node as inputs, and nodal SNBS as target values.

## 3 Performance of selected GNN models

In this section we show the performance of different GNN models. The GNN models are based on the following types of convolution: GNNs with ARMA filters by Bianchi et al. (2021), Graph Convolutional Networks (GCN) by Kipf & Welling (2017), SAmple and aggreGatE (SAGE) by Hamilton et al. (2017) and Topology Adaptive Graph Convolution (TAG) by Du et al. (2017). We refer to the models by ArmaNet, GCNNet, SAGENet and TAGNet. We conduct hyperparameter studies to set model properties such as the number of layers and channels as well as model-specific parameters e.g. the number of stacks in case of ArmaNets. As baseline, we use the best model from (Nauck et al., 2022) referred to as ArmaNet-bench. The only adjustment to that model is the removal of the fully connected layer after the second Arma-Convolution and before applying the Sigmoid-layer, which improves the training. The details of the models are given in appendix B.1.

### 3.1 Metrics for evaluation

To evaluate the performance, the coefficient of determination ($R^2$-score) and a self-defined *discretized accuracy* is used. The detailed computation of $R^2$ is given in appendix B.4. For the discretized accuracy, we rephrase the evaluation as a classification problem. Predictions are considered to be correct, if the predicted output $y$ is within a certain threshold to the target value, i.e., $t$: $|y - t| <$ threshold. We set this threshold to $0.1$, because this is small enough to differentiate between the modes in the distributions, compare fig. 1). We compute the accuracy of the classifier based on this discretization of the predictor as discretized accuracy $= 1 - \frac{\text{true positives+true negatives}}{\text{number of samples.}}$ and refer to it as discretized accuracy.

### 3.2 Experiment Setup and Results

The GNNs are trained on a nodal regression task. The power grids are represented by the adjacency matrix and a binary feature vector representing sources and sinks. Both are fed into the GNN as input, the GNN is trained to predict SNBS for each node. We split the datasets in training, validation and testing sets (70:15:15). The validation set is used for the hyperparameter studies, we report the performance on the test set. Details of the training process including the used software is given in

Table 1: Results represented by $R^2$ score and discretized accuracy in %

| model | dataset20 | | dataset100 | | tr20ev100 | |
|---|---|---|---|---|---|---|
| | $R^2$ | discr. accu | $R^2$ | discr. accu | $R^2$ | discr. accu |
| ArmaNet-bench | 51.82 | 88.93 | 54.19 | 83.58 | 38.80 | 70.32 |
| ArmaNet | 80.63 | 94.90 | 85.66 | 94.31 | 66.75 | 83.57 |
| GCNNet | 70.90 | 93.13 | 75.49 | 91.09 | 58.30 | 80.19 |
| SAGENet | 65.65 | 90.18 | 75.69 | 90.01 | 52.27 | 76.98 |
| TAGNet | 83.27 | 95.61 | 88.33 | 95.07 | 65.78 | 84.15 |

For dataset20 and dataset100, the models are both trained on their training and evaluated on their test sections. To evaluate the out-of-distribution generalization capabilities, we use the term *tr20ev100* meaning that the model is trained on the dataset20, but evaluated on the dataset100.

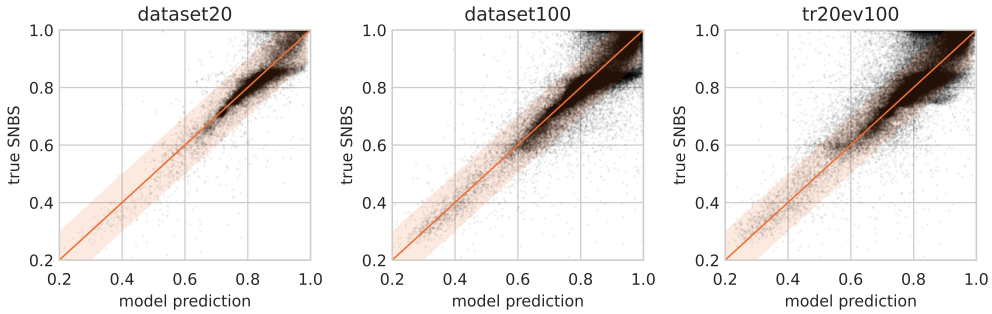


Figure 2: SNBS over predicted output of the ArmaNet model for dataset20, dataset100 and trained on dataset20, but evaluated on dataset100. The diagonal represents a perfect model (R2 = 1), the band indicates the region for correct predictions based on the discretized accuracy.

appendix B.2. To minimize the effect of initializations we use 5 different initializations per model and consider the three best only (selection of the models based on the maximum of $R^2$ and validation set, reported values are from the test set) to compute average performances.

The results are given in table 1 and visualized in fig. 2. The key result is the surprisingly high performance across all datasets. SNBS is a highly nonlinear property and the obtained performance exceeds expectations. By training on the newly introduced large datasets and using more complex models, we significantly outperform previous work. In particular, the modalities of the datasets (c.f. fig. 1) are clearly separated, as the high discretized accuracy across all experiments indicates. Most interestingly, the high performance is relatively stable even under transfer from grids of size 20 to such of size 100. We would like to emphasize the significance of that finding. Given sufficient size and complexity in the source dataset, GNNs can robustly predict highly nonlinear stability metrics for grids several times larger than the source. We did not expect grids of size 20 to be large enough to contain enough relevant structures to generalize to larger grids. However, we expect grids of size 100 to generalize well to even larger grids and are encouraged by the out-of-distribution results. Generalizing from small, numerically solvable grids to large grids is key for real world application.

## 4 Conclusion and Outlook

In this work, we introduce two new datasets of synthetical powergrids with nodal dynamical stability values. We train several benchmark GNN models to estimate the feasibility of predicting dynamic stability using GNNs. Our results show that i) dynamic stability can be predicted to a surprising degree of accuracy; ii) the modes in dynamic stability are recovered with high precision; and iii) prediction generalize from small grids of size 20 to grids of size 100. Those successes might quickly lead to real world applications, because there is no shortcut to reliably predict the dynamic stability of power grids up to today. In future we focus on reducing the simplifications step by step to get closer to real power grids. Furthermore, applications of interpretabilty methods on the GNNs might help to identify unknown patterns in power grids which are relevant for their dynamic stability.

## REFERENCES

Filippo Maria Bianchi, Daniele Grattarola, Lorenzo Livi, and Cesare Alippi. Graph Neural Networks with Convolutional ARMA Filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2021. ISSN 1939-3539. doi: 10.1109/TPAMI.2021.3054830. Conference Name: IEEE Transactions on Pattern Analysis and Machine Intelligence.

Valentin Bolz, Johannes Rueß, and Andreas Zell. Power Flow Approximation Based on Graph Convolutional Networks. In *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pp. 1679–1686, December 2019. doi: 10.1109/ICMLA.2019.00274.

Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges. *arXiv:2104.13478 [cs, stat]*, May 2021. URL http://arxiv.org/abs/2104.13478. arXiv: 2104.13478.

Balthazar Donon, Benjamin Donnot, Isabelle Guyon, and Antoine Marot. Graph Neural Solver for Power Systems. In *Proceedings of the International Joint Conference on Neural Networks*, volume 2019-July. Institute of Electrical and Electronics Engineers Inc., July 2019. ISBN 978-1-72811-985-4. doi: 10.1109/IJCNN.2019.8851855.

Jian Du, Shanghang Zhang, Guanhang Wu, Jose M. F. Moura, and Soummya Kar. Topology Adaptive Graph Convolutional Networks. October 2017. URL http://arxiv.org/abs/1710.10370. _eprint: 1710.10370.

Matthias Fey and Jan E Lenssen. FAST GRAPH REPRESENTATION LEARNING WITH PYTORCH GEOMETRIC. pp. 9, 2019. URL http://arxiv.org/abs/1903.02428.

Fernando Gama, Ekaterina Tolstaya, and Alejandro Ribeiro. Graph Neural Networks for Decentralized Controllers. March 2020. URL http://arxiv.org/abs/2003.10280. _eprint: 2003.10280.

William L. Hamilton. Graph Representation Learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(3):1–159, September 2020. ISSN 1939-4608, 1939-4616. doi: 10.2200/S01045ED1V01Y202009AIM046. URL https://www.morganclaypool.com/doi/10.2200/S01045ED1V01Y202009AIM046.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, 2017. ISSN: 10495258 _eprint: 1706.02216.

Cheolmin Kim, Kibaek Kim, Prasanna Balaprakash, and Mihai Anitescu. Graph Convolutional Neural Networks for Optimal Load Shedding under Line Contingency. In *2019 IEEE Power Energy Society General Meeting (PESGM)*, pp. 1–5, August 2019. doi: 10.1109/PESGM40551.2019.8973468. ISSN: 1944-9933.

Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv:1609.02907 [cs, stat]*, February 2017. URL http://arxiv.org/abs/1609.02907. arXiv: 1609.02907.

Yoshiki Kuramoto. Self-entrainment of a population of coupled non-linear oscillators. In *International Symposium on Mathematical Problems in Theoretical Physics*. 2005. doi: 10.1007/bfb0013365.

Yuxiao Liu, Ning Zhang, Dan Wu, Audun Botterud, Rui Yao, and Chongqing Kang. Searching for Critical Power System Cascading Failures with Graph Convolutional Network. *IEEE Transactions on Control of Network Systems*, pp. 1–1, 2021. ISSN 2325-5870. doi: 10.1109/TCNS.2021.3063333. Conference Name: IEEE Transactions on Control of Network Systems.

Peter J. Menck, Jobst Heitzig, Norbert Marwan, and Jürgen Kurths. How basin stability complements the linear-stability paradigm. *Nature Physics*, 9(2):89–92, February 2013. ISSN 1745-2481. doi: 10.1038/nphys2516. URL https://www.nature.com/articles/nphys2516. Number: 2 Publisher: Nature Publishing Group.

George S. Misyris, Andreas Venzke, and Spyros Chatzivasileiadis. Physics-Informed Neural Networks for Power Systems. In *2020 IEEE Power Energy Society General Meeting (PESGM)*, pp. 1–5, August 2020. doi: 10.1109/PESGM41954.2020.9282004. ISSN: 1944-9933.

Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I. Jordan, and Ion Stoica. Ray: A Distributed Framework for Emerging AI Applications. *arXiv:1712.05889 [cs, stat]*, September 2018. URL http://arxiv.org/abs/1712.05889. arXiv: 1712.05889.

Christian Nauck, Michael Lindner, Konstantin Schürholt, Haoming Zhang, Paul Schultz, Jürgen Kurths, Ingrid Isenhardt, and Frank Hellmann. Predicting basin stability of power grids using graph neural networks. *New Journal of Physics*, 2022. ISSN 1367-2630. doi: 10.1088/1367-2630/ac54c9. URL http://iopscience.iop.org/article/10.1088/1367-2630/ac54c9.

Damian Owerko, Fernando Gama, and Alejandro Ribeiro. Optimal Power Flow Using Graph Neural Networks. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5930–5934, May 2020. doi: 10.1109/ICASSP40776.2020.9053140. ISSN: 2379-190X.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

Nicolas Retiére, Dinh Truc Ha, and Jean-Guy Caputo. Spectral Graph Analysis of the Geometry of Power Flows in Transmission Networks. *IEEE Systems Journal*, 14(2):2736–2747, June 2020. ISSN 1937-9234. doi: 10.1109/JSYST.2019.2928852. Conference Name: IEEE Systems Journal.

Francisco A. Rodrigues, Thomas K. DM. Peron, Peng Ji, and Jürgen Kurths. The Kuramoto model in complex networks. *Physics Reports*, 610:1–98, January 2016. ISSN 0370-1573. doi: 10.1016/j.physrep.2015.10.008. URL https://www.sciencedirect.com/science/article/pii/S0370157315004408.

Paul Schultz. *luap-pik/SyntheticNetworks*. 2020. URL https://github.com/luap-pik/SyntheticNetworks.

Paul Schultz, Jobst Heitzig, and Jürgen Kurths. A random growth model for power grids and other spatially embedded infrastructure networks. *The European Physical Journal Special Topics*, 223 (12):2593–2610, October 2014. ISSN 1951-6401. doi: 10.1140/epjst/e2014-02279-6. URL https://doi.org/10.1140/epjst/e2014-02279-6.

Dawei Wang, Kedi Zheng, Qixin Chen, Gang Luo, and Xuan Zhang. Probabilistic Power Flow Solution with Graph Convolutional Network. In *2020 IEEE PES Innovative Smart Grid Technologies Europe (ISGT-Europe)*, pp. 650–654, October 2020. doi: 10.1109/ISGT-Europe47291.2020.9248786.

## A Data and source code availability

We took great care to ensure reproducibility. Code and data will be made available upon publication. This includes code to generate the dataset, as well as reproducing training results and also the code to generate the figures.

## B Training of GNNs

This section covers more information regarding the used models, the training parameters and the hyperparameter optimization.

Table 2: Properties of models. Number of parameters denotes the number of learnable weights of the model.

| name | type of convolution | number of layers | number of parameters |
|---|---|---|---|
| ArmaNet-bench | ARMA | 2 | 1,050 |
| ArmaNet | ARMA | 3 | 189,048 |
| GCNNet | GCN | 7 | 523,020 |
| SAGENet | SAGE | 8 | 728869 |
| TAGNet | TAG | 13 | 415,320 |

### B.1 Model details

Details of the models such as number of layers and parameters are provided in table 2.

### B.2 Training of models

The training is implemented in Pytorch (Paszke et al., 2019). For the graph handling and graph convolutional layers we rely on the additional library PyTorch Geometric (Fey & Lenssen, 2019). As loss function we use the mean squared error [2]. Furthermore ray (Moritz et al., 2018) is used for parallelizing the hyperparameter study.

### B.3 Hyperparameter study

For different architectures, we investigate multiple hyperparameters to find appropriate models. For all models we investigated the influence of different numbers of layers and the numbers of channel between multiple layers. We limited the model size to little above four million parameters, so we did not investigate the full presented space, but limited for example the number of channels when adding more layers. T

### B.4 Computation of $R^2$

The score $R^2$ is computed by $R^2 = 1 - \frac{mse(y,t)}{mse(t_{mean},t)}$, where $mse$ denotes the mean squared error, $y$ the output of the model, $t$ the target value and $t_{mean}$ the mean of all considered targets of the test dataset. $R^2$ captures the mean square error relative to a null model that predicts the mean of the test-dataset for all points. The $R^2$-score is used to measure the portion of explained variance in a dataset.

[2] corresponds to MSELoss in Pytorch