Projected Neural Differential Equations for Power Grid Modeling with Constraints

Alistair White[1,2*] **Anna Büttner**[2] **Maximilian Gelbrecht**[1,2] **Niki Kilbertus**[1,3,4]
Frank Hellmann[2] **Niklas Boers**[1,2,5]

[1]Technical University of Munich
[2]Potsdam Institute for Climate Impact Research
[3]Helmholtz Munich
[4]Munich Center for Machine Learning (MCML)
[5]University of Exeter

Abstract

Neural differential equations offer a powerful approach for data-driven simulation. However, many applications in science and engineering possess known constraints that should be obeyed by the learned model. We introduce *projected neural differential equations* (PNDEs), a new method for constraining neural differential equations based on projection of the learned vector field to the tangent space of the constraint manifold. In tests on two challenging examples from power grid modeling, PNDEs outperform existing methods while requiring fewer hyperparameters. Our approach demonstrates significant potential for enhancing the modeling of constrained dynamical systems, particularly in complex domains like power grid dynamics, where accuracy and reliability are essential.

1 Introduction and Related Work

Numerical simulation of dynamical systems plays a pivotal role in science and engineering. Across fields, simulations extend the reach of scientific inquiry far beyond the limitations of direct observation and experimentation. Deep learning, enabled by the emergence of efficient model architectures and specialized accelerator hardware, has further advanced the use of simulations for scientific inquiry [20, 23, 1, 34, 18, 31].

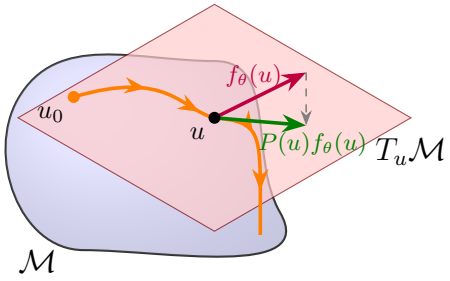

Figure 1: Schematic of projected neural differential equations (PNDEs). The vector field of the unconstrained NDE (red arrow) is projected to the tangent space $T_u\mathcal{M}$ of the constraint manifold $\mathcal{M}$. For initial conditions $u_0 \in \mathcal{M}$, solutions (orange) of the projected vector field (green arrow) remain on the manifold and thereby satisfy the constraints.

In power grid modeling, simulations allow grid operators and researchers to examine the stability of the complex electrical networks that underpin modern life. However, the transition to renewable energy sources presents unprecedented challenges. As wind and solar generation increasingly contribute to the energy mix, grid operators must contend with distributed and highly variable production, alongside the loss of system inertia that was previously provided by conventional generators [6, 32]. Meanwhile, existing methods for modeling grid dynamics are computationally infeasible at the scale and accuracy required for realistically modeling future scenarios dominated by renewable energy sources [29]. This challenge is further exacerbated by legal mandates requiring grid operators to perform large-scale

*alistair.white@tum.de

D3S3: Data-driven and Differentiable Simulations, Surrogates, and Solvers @ NeurIPS 2024.

probabilistic simulations [13]. By leveraging large amounts of operational data to learn and predict grid dynamics, machine learning can enable grid operators to efficiently forecast system states, mitigate faults, and maintain stability in future power grids. As with all machine learning, however, considering appropriate inductive biases and constraints will be essential for the robustness of this approach.

Neural ordinary differential equations (NODEs) offer a promising approach for data-driven modeling of dynamical systems, combining the expressiveness of neural networks with the interpretability of differential equations [9]. Building upon NODEs, universal differential equations (UDEs) offer a hybrid framework that integrates neural networks with differentiable physical models [38]. We refer to these approaches collectively as *neural differential equations* (NDEs). NDEs have demonstrated success across various scientific disciplines, including weather and climate modeling [21], robotics [41], epidemiology [12] and bioengineering [33].

Despite excellent performance within the training data, NDEs often generalize poorly, for example, to new initial conditions or future times. Inductive biases play a crucial role in avoiding such issues. In scientific machine learning, inductive biases can take the form of physical constraints, conservation laws, or other domain-specific knowledge that enhances model performance and interpretability [39, 2, 28, 15, 11, 14]. For power grids, in particular, the dynamics are typically modeled with hard constraints on the power consumed at certain nodes. In this paper, we study two challenging and impactful power grid examples, requiring significantly higher dimensions and using more constraints than previous, comparable works.

A large number of related works have studied neural ODEs on Riemannian manifolds, primarily in the context of continuous normalizing flows on non-Euclidean geometries [27, 4, 30, 40, 42, 3]. In contrast to the work presented here, they typically deal with prototypical Riemannian manifolds, such as spheres and tori, and are not easily adapted to the more general manifolds studied in this paper. White et al. [44], whose work is most closely related to ours in terms of purpose and method, incorporate hard constraints in NDE models by stabilizing the differential equation's vector field, such that the learned dynamics are "nudged" to remain on the constraint manifold.

Contribution In this paper, we develop an alternative method called *projected neural differential equations* (PNDEs). Our method is based on projection of the NDE's vector field to the tangent space of the constraint manifold, which is defined by arbitrary algebraic constraints on the state of the system (see Figure 1). We demonstrate empirically on two elaborate examples from power grid modeling that our approach satisfies the constraints more accurately, while also requiring fewer hyperparameters and permitting larger timesteps of the numerical solver.

2 Methods

Neural Differential Equations A first order neural differential equation is given by

$$\dot{u} = f_\theta(u(t), t) \qquad u(0) = u_0, \tag{1}$$

where $u : \mathbb{R} \to \mathbb{R}^n$, $\dot{u} = du/dt$, the vector field $f_\theta \colon \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$ is at least partially parameterized by a neural network with parameters $\theta \in \mathbb{R}^d$, and $u_0 \in \mathbb{R}^n$ is an initial condition.

Given observations $(t_i, u(t_i))_{i=1}^N$, the parameters θ can be optimized using stochastic gradient descent by integrating a trajectory $\hat{u}(t) = \mathrm{ODESolve}(u_0, f_\theta, t)$ and taking gradients of the loss $\mathcal{L}(u, \hat{u})$ with respect to θ. For ease of notation, but without loss of generality, we will drop the time-dependence in Equation (1) and consider only autonomous systems from now on.

Constraints Assume we have $m < n$ explicit algebraic constraints on solutions $u(t)$ of Equation (1), defined as the zero-set of the function $g(u) = \big[g_1(u), \ldots, g_m(u)\big]$, where $g \colon \mathbb{R}^n \to \mathbb{R}^m$ and each g_i is an independent constraint. In other words, we want to constrain the state $u(t)$ of our system to the set

$$\mathcal{M} = \{u \in \mathbb{R}^n \,;\, g(u) = 0\}. \tag{2}$$

As long as the Jacobian $Dg(u)$ exists and has full rank for all $u \in \mathcal{M}$, then $\mathcal{M}$ is an embedded submanifold of $\mathbb{R}^n$ with dimension $n - m$ and local defining function $g(u) = 0$. In this case, our goal of satisfying the constraints is equivalent to constraining solutions of Equation (1) to the embedded submanifold $\mathcal{M}$. From now on, we will assume this is true and refer to $\mathcal{M}$ as the *constraint manifold*.

Stabilized Neural Differential Equations White et al. [44] constrain NDE solutions using stabilized neural differential equations (SNDEs), given by

$$\dot{u} = f_\theta(u) - \gamma F(u) g(u), \tag{3}$$

where $\gamma \geq 0$ is a scalar hyperparameter and $F : \mathbb{R}^n \to \mathbb{R}^{n \times m}$ is the stabilization matrix. Under mild conditions on γ and F, it can be shown that Equation (3) admits all solutions of Equation (1) while rendering the constraint manifold provably asymptotically stable. In other words, Equation (3) is able to learn the correct ground truth dynamics while guaranteeing that the constraints are satisfied.

Projected Neural Differential Equations Given the constraints $g(u) = 0$, we can derive (see Appendix A) a projection operator of the form

$$\mathrm{Proj}_u(v) = v - \mathrm{D}g(u)^* \left[\left(\mathrm{D}g(u)^* \right)^\dagger [v] \right], \tag{4}$$

where $Dg(u)$ is the Jacobian of $g(u)$ and the dagger denotes the Moore-Penrose pseudoinverse. Projecting the NDE in Equation (1) using the projection operator defined in Equation (4) gives the following *Projected Neural Differential Equation* (PNDE),

$$\boxed{\dot{u} = \mathrm{Proj}_u(f_\theta(u)) \in \mathrm{T}_u\mathcal{M},} \tag{5}$$

which is a differential equation on the constraint manifold $\mathcal{M}$.

Proposition 1. *Solutions to the PNDE 5 with $u_0 \in \mathcal{M}$ satisfy $g(u(t)) = 0$ for all $t \geq 0$.*

Proof. We note that

$$\frac{dg(u(t))}{dt} = Dg(u)[\dot{u}] = Dg(u)\left[\mathrm{Proj}_u(f_\theta(u))\right] = 0 \tag{6}$$

since $\mathrm{Proj}_u(f_\theta(u)) \in \mathrm{T}_u\mathcal{M} = \ker \mathrm{D}g(u)$. As a result, $g(u(t))$ remains constant in time and satisfies $g(u(t)) = g(u(0)) = 0$ for all $t \geq 0$ since $u_0 \in \mathcal{M}$. □

3 Data

IEEE Test Systems The IEEE test systems are standardized power system models widely used in electrical engineering research and education [10]. Representing key features of power grids such as topologies and operating conditions, these models enable researchers to benchmark algorithms and offer a common framework for analyzing power flow, stability, and reliability in electrical networks. In this paper, we will study the dynamics of two test systems in particular: the IEEE 14-Bus system and the IEEE Reliability Test System 1996 (hereafter RTS-96) [16].

We adapt the IEEE test cases to represent future grid dynamics by replacing conventional generators with renewable energy sources. Our renewables-based grids therefore consist of three types of buses: renewable energy sources, loads (that is, consumers), and slack buses (a constant voltage bus representing, for example, a large power plant or a connection to another grid). Buses, which form the nodes in a graph, are connected via transmission lines, which form the edges. The state of each bus in the grid is determined by a complex voltage, from which currents can be calculated using the line admittances, which are well understood and assumed to be known.

Renewable energy sources are connected to the grid using inverters, which we model using the normal form [22]. Loads are modeled using PQ-buses, meaning that the complex power at the PQ-bus m is set to a fixed, constant value given by $P_m + \mathrm{i}Q_m = v_m \cdot i_m^*$, where P_m and Q_m are the active and reactive power, respectively, v_m is the complex voltage, and i_m is the complex current [8]. These fixed power values will be the constraints in our data-driven simulators. For full details of how we constructed the dataset, as well as the training setup and hyperparameters, see Appendix B.

4 Results

We evaluate the trained models by solving each model with the initial conditions from the test set. Averaging over all test trajectories, we then measure how well each model predicts (1) the voltages of the entire system, and (2) the power at the PQ-buses (which should be constant). For the stabilized models, we report results for a range of values of the stabilization hyperparameter γ. Table 1 shows the mean-squared error for both systems, calculated over all test trajectories. Figure 2 shows the average relative error as a function of time, defined as $E(t) = \|u(t) - \hat{u}(t)\|_2 / \|u(t)\|_2$, where $u(t)$ is the ground truth and $\hat{u}(t)$ is the prediction, along with the average L^2-norm $\|g(u)\|_2$ of the constraint.

Unconstrained NDEs incur large errors in the PQ-bus power. For SNDEs, increasing the stabilization hyperparameter γ enforces the constraints more accurately, but larger values lead to stiffness and, ultimately, prohibitively expensive simulations. Meanwhile, PNDEs enforce the constraints exactly – close to machine precision and several orders of magnitude more accurately than the best SNDEs – without stiffness.

	IEEE 14-Bus			IEEE RTS-96		
Model	V ($\times 10^{-5}$)	PQ Power	f_θ Evals	V ($\times 10^{-4}$)	PQ Power	f_θ Evals
NDE	1.8 ± 1.8	$(3.0 \pm 2.0) \times 10^{-3}$	171	2.1 ± 0.8	$(0.9 \pm 1.1) \times 10^{-1}$	405
SNDE ($\gamma = 0.1$)	1.7 ± 1.8	$(1.5 \pm 1.3) \times 10^{-3}$	171	2.0 ± 0.8	$(5.0 \pm 6.0) \times 10^{-2}$	405
SNDE ($\gamma = 1$)	1.6 ± 1.8	$(2.0 \pm 3.0) \times 10^{-4}$	177	2.0 ± 0.8	$(0.5 \pm 1.1) \times 10^{-2}$	405
SNDE ($\gamma = 10$)	1.6 ± 1.8	$(0.5 \pm 1.3) \times 10^{-5}$	297	1.9 ± 0.8	$(1.7 \pm 7.0) \times 10^{-4}$	411
SNDE ($\gamma = 100$)	1.6 ± 1.8	$(0.5 \pm 1.6) \times 10^{-7}$	1929	1.9 ± 0.8	$(2.0 \pm 8.0) \times 10^{-6}$	1815
PNDE	1.6 ± 1.8	$\mathbf{(3.5 \pm 1.6) \times 10^{-11}}$	171	1.9 ± 0.8	$\mathbf{(1.9 \pm 1.5) \times 10^{-10}}$	405

Table 1: MSE of the predicted voltage V (all buses) and power constraint (PQ-buses only), along with the number of function evaluations required by the adaptive ODE solver. For both systems, the constrained models predict the voltages at least as well as the NDE. Unsurprisingly, the NDE incurs large errors in the PQ-bus power. SNDEs enforce the constraints more accurately for larger values of γ, but the largest values lead to stiffness, requiring many more function evaluations. PNDEs efficiently enforce the constraints near machine precision.

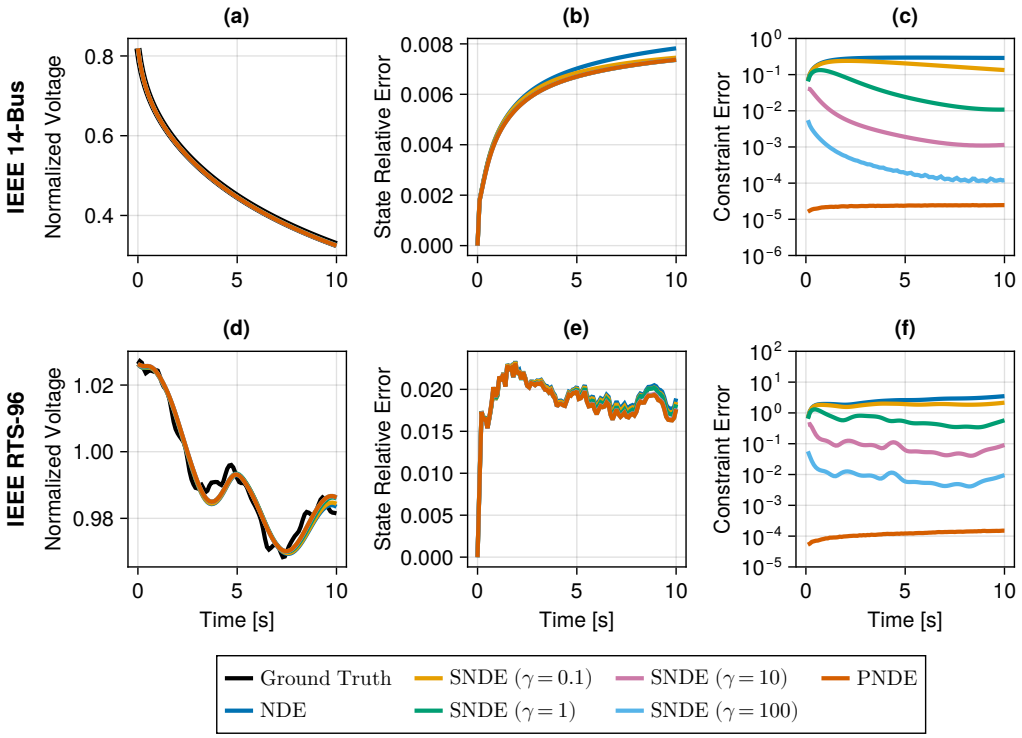

Figure 2: Test statistics for the IEEE 14-Bus System (top) and IEEE RTS-96 System (bottom). Panels (a) and (d) show a representative voltage for a single trajectory. All models accurately capture the shape of this voltage for the 14-bus system, but appear to struggle with the high frequency components of the RTS-96 system. Panels (b) and (e) show the average relative error of the voltages at all buses, calculated over the entire test set. Constrained models outperform unconstrained models, with the gap growing over time. Panels (c) and (f) show the average error in the power at the PQ-buses (that is, $\|g(u)\|_2$), with the PNDE outperforming the best SNDEs by several orders of magnitude.

5 Conclusion

We have introduced projected neural differential equations (PNDEs), a new method for enforcing arbitrary algebraic constraints in neural differential equation models. In experiments on two challenging examples from power grid modeling, PNDEs enforce the constraints orders of magnitude more accurately than existing methods and require fewer hyperparameters. Many promising directions remain for future research. For example, graph neural networks are also compatible with our method and a natural fit for modeling grid dynamics, thereby enabling critical out-of-distribution tests such as changing grid topologies.

Acknowledgments and Disclosure of Funding

This work was funded by the Volkswagen Foundation. The authors gratefully acknowledge the European Regional Development Fund (ERDF), the German Federal Ministry of Education and Research, and the Land Brandenburg for supporting this project by providing resources on the high performance computer system at the Potsdam Institute for Climate Impact Research.

References

[1] Kamyar Azizzadenesheli, Nikola Kovachki, Zongyi Li, Miguel Liu-Schiaffini, Jean Kossaifi, and Anima Anandkumar. Neural operators for accelerating scientific simulations and design. *Nature Reviews Physics*, pages 1–9, 2024.

[2] Nathan Baker, Frank Alexander, Timo Bremer, Aric Hagberg, Yannis Kevrekidis, Habib Najm, Manish Parashar, Abani Patra, James Sethian, Stefan Wild, Karen Willcox, and Steven Lee. Workshop report on basic research needs for scientific machine learning: Core technologies for artificial intelligence. Technical report, USDOE Office of Science (SC), Washington, D.C. (United States), 2019.

[3] Heli Ben-Hamu, Samuel Cohen, Joey Bose, Brandon Amos, Maximillian Nickel, Aditya Grover, Ricky T. Q. Chen, and Yaron Lipman. Matching normalizing flows and probability paths on manifolds. In *Proceedings of the 39th International Conference on Machine Learning*, pages 1749–1763. PMLR, 2022.

[4] Joey Bose, Ariella Smofsky, Renjie Liao, Prakash Panangaden, and Will Hamilton. Latent variable modelling with hyperbolic normalizing flows. In *Proceedings of the 37th International Conference on Machine Learning*, pages 1045–1055. PMLR, 2020.

[5] Nicolas Boumal. *An Introduction to Optimization on Smooth Manifolds*. Cambridge University Press, 2024.

[6] T. Breithaupt, D. Herwig, L. Hofmann, A. Mertens, R. Meyer, N. Farrokhseresht, B. Tuinema, D. Wang, J. Rueda Torres, S. Rüberg, and V. Sewdien. Deliverable D1.1: Report on systemic issues. Technical report, 2016.

[7] Anna Büttner, Jürgen Kurths, and Frank Hellmann. Ambient forcing: sampling local perturbations in constrained phase spaces. *New Journal of Physics*, 24(5):053019, 2022.

[8] Anna Büttner, Anton Plietzsch, Mehrnaz Anvari, and Frank Hellmann. A framework for synthetic power system dynamics. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 33(8):083120, 2023.

[9] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 31, pages 6571–6583. Curran Associates, Inc., 2018.

[10] Richard D. Christie. Power systems test case archive, 1999.

[11] Miles Cranmer, Sam Greydanus, Stephan Hoyer, Peter Battaglia, David Spergel, and Shirley Ho. Lagrangian neural networks. *arXiv preprint arXiv:2003.04630*, 2020.

[12] Raj Dandekar, Chris Rackauckas, and George Barbastathis. A machine learning-aided global diagnostic and comparative tool to assess effect of quarantine control in COVID-19 spread. *Patterns*, 1(9):100145, 2020.

[13] European Network of Transmission System Operators for Electricity (ENTSO-e). All continental Europe and Nordic TSOs' proposal for assumptions and a cost benefit analysis methodology in accordance with Article 156(11) of the Commission Regulation (EU) 2017/1485 of 2 August 2017 establishing a guideline on electricity transmission system operation. Technical report, 2018.

[14] Marc Finzi, Ke Alexander Wang, and Andrew G Wilson. Simplifying Hamiltonian and Lagrangian neural networks via explicit constraints. In *Advances in Neural Information Processing Systems*, volume 33, pages 13880–13889. Curran Associates, Inc., 2020.

[15] Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. In *Advances in Neural Information Processing Systems*, volume 32, pages 15379–15389. Curran Associates, Inc., 2019.

[16] C. Grigg, P. Wong, P. Albrecht, R. Allan, M. Bhavaraju, R. Billinton, Q. Chen, C. Fong, S. Haddad, S. Kuruganty, W. Li, R. Mukerji, D. Patton, N. Rau, D. Reppen, A. Schneider, M. Shahidehpour, and C. Singh. The IEEE Reliability Test System-1996. a report prepared by the Reliability Test System Task Force of the Application of Probability Methods Subcommittee. *IEEE Transactions on Power Systems*, 14(3):1010–1020, 1999.

[17] Ernst Hairer and Gerhard Wanner. *Solving Ordinary Differential Equations II*, volume 14 of *Springer Series in Computational Mathematics*. Springer, 1996.

[18] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021.

[19] Patrick Kidger. On neural differential equations. *arXiv preprint arXiv:2202.02435*, 2022.

[20] Dmitrii Kochkov, Jamie A. Smith, Ayya Alieva, Qing Wang, Michael P. Brenner, and Stephan Hoyer. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21):e2101784118, 2021.

[21] Dmitrii Kochkov, Janni Yuval, Ian Langmore, Peter Norgaard, Jamie Smith, Griffin Mooers, Milan Klöwer, James Lottes, Stephan Rasp, Peter Düben, Sam Hatfield, Peter Battaglia, Alvaro Sanchez-Gonzalez, Matthew Willson, Michael P. Brenner, and Stephan Hoyer. Neural general circulation models for weather and climate. *Nature*, pages 1–7, 2024.

[22] Raphael Kogler, Anton Plietzsch, Paul Schultz, and Frank Hellmann. Normal form for grid-forming power grid actors. *PRX Energy*, 1(1):013008, 2022.

[23] Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: Learning maps between function spaces with applications to PDEs. *Journal of Machine Learning Research*, 24(89):1–97, 2023.

[24] Serge Lang. *Fundamentals of Differential Geometry*, volume 191 of *Graduate Texts in Mathematics*. Springer, 1999.

[25] John M. Lee. *Introduction to Smooth Manifolds*, volume 218 of *Graduate Texts in Mathematics*. Springer New York, 2012.

[26] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2018.

[27] Aaron Lou, Derek Lim, Isay Katsman, Leo Huang, Qingxuan Jiang, Ser Nam Lim, and Christopher M De Sa. Neural manifold ordinary differential equations. In *Advances in Neural Information Processing Systems*, volume 33, pages 17548–17558. Curran Associates, Inc., 2020.

[28] Michael Lutter, Christian Ritter, and Jan Peters. Deep Lagrangian networks: Using physics as model prior for deep learning. In *International Conference on Learning Representations*, 2018.

[29] Julia Matevosyan, Babak Badrzadeh, Thibault Prevost, Eckard Quitmann, Deepak Ramasubramanian, Helge Urdal, Sebastian Achilles, Jason MacDowell, Shun Hsien Huang, Vijay Vital, Jon O'Sullivan, and Ryan Quint. Grid-forming inverters: Are they the key for high renewable penetration? *IEEE Power and Energy Magazine*, 17(6):89–98, 2019.

[30] Emile Mathieu and Maximilian Nickel. Riemannian continuous normalizing flows. In *Advances in Neural Information Processing Systems*, volume 33, pages 2503–2515. Curran Associates, Inc., 2020.

[31] Amil Merchant, Simon Batzner, Samuel S. Schoenholz, Muratahan Aykol, Gowoon Cheon, and Ekin Dogus Cubuk. Scaling deep learning for materials discovery. *Nature*, 624(7990):80–85, 2023.

[32] Federico Milano, Florian Dörfler, Gabriela Hug, David J. Hill, and Gregor Verbič. Foundations and challenges of low-inertia systems (invited paper). In *2018 Power Systems Computation Conference (PSCC)*, pages 1–25, 2018.

[33] Harini Narayanan, Martin Luna, Michael Sokolov, Alessandro Butté, and Massimo Morbidelli. Hybrid models based on machine learning and an increasing degree of process knowledge: Application to cell culture processes. *Industrial & Engineering Chemistry Research*, 61(25): 8658–8672, 2022.

[34] Jaideep Pathak, Shashank Subramanian, Peter Harrington, Sanjeev Raja, Ashesh Chattopadhyay, Morteza Mardani, Thorsten Kurth, David Hall, Zongyi Li, Kamyar Azizzadenesheli, Pedram Hassanzadeh, Karthik Kashinath, and Animashree Anandkumar. FourCastNet: A global data-driven high-resolution weather model using adaptive Fourier neural operators, 2022.

[35] Anton Plietzsch, Raphael Kogler, Sabine Auer, Julia Merino, Asier Gil-de Muro, Jan Liße, Christina Vogel, and Frank Hellmann. PowerDynamics.jl—an experimentally validated open-source package for the dynamical analysis of power grids. *SoftwareX*, 17:100861, 2022.

[36] Lev Semenovich Pontryagin. *The Mathematical Theory of Optimal Processes*. Interscience Publishers, 1962.

[37] Christopher Rackauckas and Qing Nie. DifferentialEquations.jl – a performant and feature-rich ecosystem for solving differential equations in julia. *Journal of Open Research Software*, 5(1), 2017.

[38] Christopher Rackauckas, Yingbo Ma, Julius Martensen, Collin Warner, Kirill Zubov, Rohit Supekar, Dominic Skinner, Ali Ramadhan, and Alan Edelman. Universal differential equations for scientific machine learning. *arXiv preprint arXiv:2001.04385*, 2021.

[39] M. Raissi, P. Perdikaris, and G. E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

[40] Danilo Jimenez Rezende, George Papamakarios, Sebastien Racaniere, Michael Albergo, Gurtej Kanwar, Phiala Shanahan, and Kyle Cranmer. Normalizing flows on tori and spheres. In *Proceedings of the 37th International Conference on Machine Learning*, pages 8083–8092. PMLR, 2020.

[41] Spencer Richards, Navid Azizan, Jean-Jacques Slotine, and Marco Pavone. Adaptive-control-oriented meta-learning for nonlinear systems. In *Robotics: Science and Systems XVII*. Robotics: Science and Systems Foundation, 2021.

[42] Noam Rozen, Aditya Grover, Maximilian Nickel, and Yaron Lipman. Moser flow: Divergence-based generative modeling on manifolds. In *Advances in Neural Information Processing Systems*, volume 34, pages 17669–17680. Curran Associates, Inc., 2021.

[43] Ch. Tsitouras. Runge–Kutta pairs of order 5(4) satisfying only the first column simplifying assumption. *Computers & Mathematics with Applications*, 62(2):770–775, 2011.

[44] Alistair White, Niki Kilbertus, Maximilian Gelbrecht, and Niklas Boers. Stabilized neural differential equations for learning dynamics with explicit constraints. In *Advances in Neural Information Processing Systems*, volume 36, pages 12929–12950, 2023.

A Derivation of the Projection Operator

A.1 Background

We first review some differential geometry definitions that are relevant to the proposed approach and refer the reader to [24, 25, 5] for a more detailed treatment. The set of tangent vectors to a manifold $\mathcal{M}$ at a point $u \in \mathcal{M}$ is a vector space called the *tangent space* to $\mathcal{M}$ at u and is denoted by $\mathrm{T}_u\mathcal{M}$. The disjoint union of all the tangent spaces to $\mathcal{M}$ forms the *tangent bundle* $\mathrm{T}\mathcal{M} = \{(u, v) \mid u \in \mathcal{M},\ v \in \mathrm{T}_u\mathcal{M}\}$ of $\mathcal{M}$. A *vector field* on a manifold $\mathcal{M}$ is a continuous map $X : \mathcal{M} \to \mathrm{T}\mathcal{M}$ with the property that $X(u) \in \mathrm{T}_u\mathcal{M}$ for all $u \in \mathcal{M}$. The *integral curve* at u of a vector field X on $\mathcal{M}$ is the smooth curve γ on $\mathcal{M}$ such that $\gamma(0) = u$ and $\gamma'(t) = X(\gamma(t))$ for all t in the interval of interest. The integral curve γ is simply the solution of a first-order ODE with right-hand side $X_{\gamma(t)}$ for t in the interval of interest, so theorems for the existence, uniqueness, and smoothness of ODE solutions translate directly to integral curves of vector fields on manifolds.

A.2 Problem Setting

Neural Differential Equations. Assume we have an ambient n-dimensional Euclidean phase space $\mathcal{E}$, and consider the neural differential equation

$$\dot{u} = f_\theta(u(t), t) \qquad u(0) = u_0, \tag{7}$$

where $u : \mathbb{R} \to \mathcal{E}$, $u_0 \in \mathcal{E}$ is an initial condition, $\dot{u} = du/dt$, and f_θ is parameterized by a neural network with parameters $\theta \in \mathbb{R}^d$.

Given observations $\{t_i, u(t_i)\}_{i=1}^N$, the parameters θ can be optimized using stochastic gradient descent by integrating a trajectory $\hat{u}(t) = \mathrm{ODESolve}(u_0, f_\theta, t)$ and taking gradients of the loss $\mathcal{L}(u, \hat{u})$ with respect to θ. Gradients of the $\mathrm{ODESolve}$ operation can be obtained using adjoint sensitivity analysis (sometimes called *optimize-then-discretize*) or via direct automatic differentiation of the solver operations (*discretize-then-optimize*) [9, 19].

Since any non-autonomous system can be equivalently represented as an autonomous system by adding time as an additional coordinate, for ease of notation we will drop the time-dependence in Equation (7) and consider only autonomous systems from now on, without loss of generality.

Constraints. Suppose that the state $u(t)$ of the system must be constrained to the set

$$\mathcal{M} = \{u \in \mathcal{E}\,;\, g(u) = 0\}, \qquad \text{where} \quad g\colon \mathcal{E} \to \mathbb{R}^m,\ \ g : u \mapsto \big(g_1(u), \ldots, g_m(u)\big), \tag{8}$$

given $m < n$ independent explicit algebraic constraints $g_i(u) = 0$. Assuming that the Jacobian $Dg(u)$ exists and has full rank for all $u \in \mathcal{M}$, the Regular Level Set Theorem [25, #5.14] implies that $\mathcal{M}$ is a smooth $(n - m)$-dimensional embedded submanifold of $\mathbb{R}^n$. To satisfy the constraints $g(u) = 0$, we seek solutions of Equation (7) that are constrained to the embedded submanifold $\mathcal{M}$, which we refer to as the *constraint manifold*. In other words, we seek integral curves on $\mathcal{M}$.

A.3 Proposed Approach

Proposed Approach. We want to learn $f_\theta : \mathcal{M} \to \mathrm{T}\mathcal{M}$ as a smooth vector field on the embedded submanifold $\mathcal{M}$ with local defining function g. We realize this by first parameterizing a smooth extension $\bar{f}_\theta\colon \mathcal{E} \to \mathcal{E}$ of f_θ in Euclidean space using a neural network, and then projecting $\bar{f}_\theta(u)$ using a suitable projection operator $\mathrm{Proj}_u\colon \mathcal{E} \to \mathrm{T}_\mathrm{u}\mathcal{M}$ to obtain the desired tangent vector,

$$f_\theta(u) = \mathrm{Proj}_u\big(\bar{f}_\theta(u)\big) \in \mathrm{T}_\mathrm{u}\mathcal{M}. \tag{9}$$

Since $\mathcal{M}$ is a smooth embedded submanifold of $\mathcal{E}$, the restriction $f_\theta = \bar{f}_\theta|_\mathcal{M}$ is also smooth. Note that the projection acts fibrewise (i.e. acting "point-by-point"); for each $u \in \mathcal{M}$, it projects the vector $\bar{f}_\theta(u)$ at u onto $\mathrm{T}_\mathrm{u}\mathcal{M}$, as opposed to projecting directly the full vector field $\bar{f}_\theta$ onto $\mathrm{T}\mathcal{M}$.

Construction of the projection operator. We now derive a suitable projection operator, following the approach of Boumal [5]. Given an inner product $\langle\cdot,\cdot\rangle$ and induced norm $\|\cdot\|$ on $\mathcal{E}$, the differential $\mathrm{D}g(u)$ and its adjoint $\mathrm{D}g(u)^*$ define linear maps

$$\mathrm{D}g(u)[v] = \big(\langle\nabla g_1(u), v\rangle, \ldots, \langle\nabla g_m(u), v\rangle\big), \qquad \mathrm{D}g(u)^*[\alpha] = \sum_{i=1}^m \alpha_i \nabla g_i(u). \tag{10}$$

Denoting the orthogonal projection from $\mathcal{E}$ to $\mathrm{T}_u\mathcal{M}$ by $\mathrm{Proj}_u : \mathcal{E} \to \mathrm{T}_u\mathcal{M}$, we can uniquely decompose any vector $v \in \mathcal{E}$ into its components parallel and perpendicular to $\mathrm{T}_u\mathcal{M}$,

$$v = \mathrm{Proj}_u(v) + \mathrm{D}g(u)^*[\alpha]. \tag{11}$$

The coefficients α are obtained as the solution of the least-squares problem

$$\alpha = \underset{\alpha \in \mathbb{R}^m}{\arg\min} \|v - \mathrm{D}g(u)^*[\alpha]\|^2 = \left(\mathrm{D}g(u)^*\right)^\dagger [v], \tag{12}$$

where $(\cdot)^\dagger$ denotes the Moore-Penrose pseudoinverse. Substituting Equation (12) into Equation (11), we obtain a formula for the orthogonal projection from $\mathcal{E}$ to $\mathrm{T}_u\mathcal{M}$,

$$\boxed{\mathrm{Proj}_u(v) = v - \mathrm{D}g(u)^* \left[\left(\mathrm{D}g(u)^*\right)^\dagger [v]\right].} \tag{13}$$

B Data and Training Details

Complex Numbers Given complex voltages and complex constraints, we treat the real and imaginary parts independently. Hence, for the IEEE 14-Bus system, with 14 buses in total and nine PQ-buses, we have $n = 28$ voltages and $m = 18$ constraints. Similarly, for the IEEE RTS-96 system, with 74 buses in total and 40 PQ-buses, we have $n = 128$ voltages and $m = 80$ constraints.

Initial Conditions Given a power grid at its operation point – a stable equilibrium with all variables at their desired steady-state values – we obtain perturbed initial conditions using the ambient forcing method of Büttner et al. [7], which ensures that the perturbed state satisfies the PQ-bus constraints. The transient behavior of the grid following the perturbation produces rich and complex dynamics that we aim to emulate using NDEs.

Generating the Data The grid models, which are differential-algebraic equations (DAEs), are implemented in the Julia package PowerDynamics.jl [35]. Given a random initial condition, we then generate a ground truth trajectory by solving the DAE for 10 seconds using a 4th order L-stable Rosenbrock method [17], implemented in the Julia package DifferentialEquations.jl [37] as `Rodas4`. We save the result every 0.1 seconds, yielding 101 observations per initial condition. We use 70 initial conditions for training, 30 for validation, and a further 100 for testing. For training and validation, each trajectory is split into contiguous chunks consisting of four observations, which are then randomized and combined along the batch dimension. Test statistics are calculated using the full trajectory, meaning that the test set is not split into chunks.

Training All models are trained for 10,000 epochs on NVIDIA H100 GPUs using the AdamW optimizer [26] with a constant learning rate of 10^{-3}, weight decay of 10^{-4}, and a batch size of 1024. All hyperparameters are optimized via grid search. Both systems are modeled using a multilayer perceptron with 3 hidden layers and ReLU activation functions. The hidden layers contain 512 and 1024 neurons for the IEEE 14-Bus and IEEE RTS-96 systems, respectively. During both training and evaluation, all models are solved using the 5(4) explicit Runge-Kutta algorithm of Tsitouras [43], implemented in DifferentialEquations.jl [37] as `Tsit5`, with absolute and relative tolerances of 10^{-4}. Gradients of ODE solutions are obtained using the adjoint sensitivity method [36, 9], implemented in SciMLSensitivity.jl as `BacksolveAdjoint` [38].