Unified Implementations of Recurrent Neural Networks in Multiple Deep Learning Frameworks

Francesco Martinuzzi MARTINUZZI@PKS.MPG.DE
Max Planck Institute for the Physics of Complex Systems, Dresden, Germany

Abstract

Recurrent neural networks (RNNs) are a cornerstone of sequence modeling across various scientific and industrial applications. Owing to their versatility, numerous RNN variants have been proposed over the past decade, aiming to improve the modeling of long-term dependencies and to address challenges such as vanishing and exploding gradients. However, no central library is available to test these variations, and reimplementing diverse architectures can be time-consuming and error-prone, limiting reproducibility and exploration. Here, we introduce three open-source libraries in Julia and Python that centralize numerous recurrent cell implementations and higher-level recurrent architectures. `torchrecurrent`, `RecurrentLayers.jl`, and `LuxRecurrentLayers.jl` offer a consistent framework for constructing and extending RNN models, providing built-in mechanisms for customization and experimentation. All packages are available under the MIT license and actively maintained on GitHub.

Keywords: Recurrent neural networks, recurrent layers, PyTorch, Julia

1 Introduction

Sequential data, such as time series, are pervasive across multiple scientific disciplines (Manning and Schütze, 1999; Goldberger et al., 2000; Kantz and Schreiber, 2003). With the increasing availability of large datasets and computational resources, modern modeling approaches increasingly rely on deep learning (DL) models (Längkvist et al., 2014; Lim and Zohren, 2021). Among DL architectures, recurrent neural networks (RNNs) stand out due to their explicit modeling of temporal dependencies (Jordan, 1986; Elman, 1990). Thanks to their embedded temporal inductive bias, RNNs have found applications in diverse fields, including Earth sciences (Kratzert et al., 2018; Arcomano et al., 2020; Martinuzzi et al., 2024), speech and audio processing (Graves et al., 2013; Hannun et al., 2014), neuroscience (Yang et al., 2019; Durstewitz et al., 2023), and dynamical system reconstruction (Pathak et al., 2017; Vlachas et al., 2020). This wide range of applications has, in turn, motivated the development of numerous RNN variants, each introducing unique design principles. Examples include architectures with gating mechanisms (Hochreiter and Schmidhuber, 1997; Cho et al., 2014; Zhou et al., 2016), physics-inspired dynamics (Rusch and Mishra, 2021; Rusch et al., 2022), and geometric constraints (Arjovsky et al., 2016; Chang et al., 2019; Kerg et al., 2019).

Despite the diversity of proposed architectures, only a small subset of RNNs are supported by mainstream DL libraries. Code accompanying individual papers may exist, but it is often hardcoded for specific tasks, not continuously maintained, or incompatible with current frameworks. Additionally, a central repository of unified implementations of RNNs

is lacking in any DL framework. The absence of centralized, streamlined, and up-to-date implementations restricts researchers' ability to test or extend alternative designs, concentrating empirical work on a few canonical architectures. In fact, virtually all the mainstream DL libraries limit their offering to the Elman RNN (Elman, 1990), the long short-term memory (LSTM; Hochreiter and Schmidhuber, 1997), and the gated recurrent unit (GRU; Cho et al., 2014).

To enable easier exploration of alternative RNN designs and increase the number of implementations available to researchers, we introduce three open-source libraries implementing a wide range of recurrent architectures: `torchrecurrent` (based on `PyTorch` (Paszke et al., 2019)), `RecurrentLayers.jl` (based on `Flux.jl` (Innes et al., 2018)), and `LuxRecurrentLayers.jl` (based on `Lux.jl` (Pal, 2023)). Collectively, we refer to these as `recurrent`. Each library provides unified primitives for constructing RNNs within its respective framework, along with multiple implementations of recurrent cells and layers. Models from `recurrent` can be invoked using the same interface as their native counterparts, allowing easy integration into existing workflows. At the same time, the layers in `recurrent` provide additional configuration options to support customization and exploration of novel RNN architectures. By standardizing these implementations, `recurrent` lowers the barrier to experimentation and accelerates the development of new recurrent models.

2 Software Overview

Installation All three libraries are registered in the main software repositories of their respective languages. The Julia packages are available in the General Registry[1], while `torchrecurrent` can be installed directly from PyPI[2]. Because the primary dependency of each library is its corresponding deep learning framework, no additional installation steps are required for components such as graphical processing unit (GPU) support.

Models The libraries implement a diverse set of recurrent architectures, organized into three levels of abstraction: *cells*, which represent single computational units; *layers*, which apply these computations across full sequences; and *wrappers*, which extend either cells or layers with additional behaviors. Wrapped components retain the same interface as their unwrapped counterparts, allowing them to be used interchangeably within existing workflows. Currently, `recurrent` includes approximately 30 recurrent cells, each with corresponding layers and optional wrappers. The available cells span a wide range of designs, including gated architectures (Kusupati et al., 2018; Ravanelli et al., 2018; Landi et al., 2021), physics-inspired formulations (Rusch and Mishra, 2021; Rusch et al., 2022), and models that use alternative integration schemes (Krause et al., 2017; Li et al., 2018).

While the libraries currently provide largely overlapping model implementations, the development of each library progresses independently. Each library may gradually specialize in a specific niche, shaped by the interests of its user community. For instance, given the strong focus on scientific machine learning within the Julia ecosystem (Rackauckas et al., 2019; Zubov et al., 2021; Martinuzzi et al., 2022), it is likely that `LuxRecurrentLayers.jl`

1. github.com/JuliaRegistries/General/tree/master/R/RecurrentLayers and github.com/JuliaRegistries/General/tree/master/L/LuxRecurrentLayers
2. pypi.org/project/torchrecurrent/

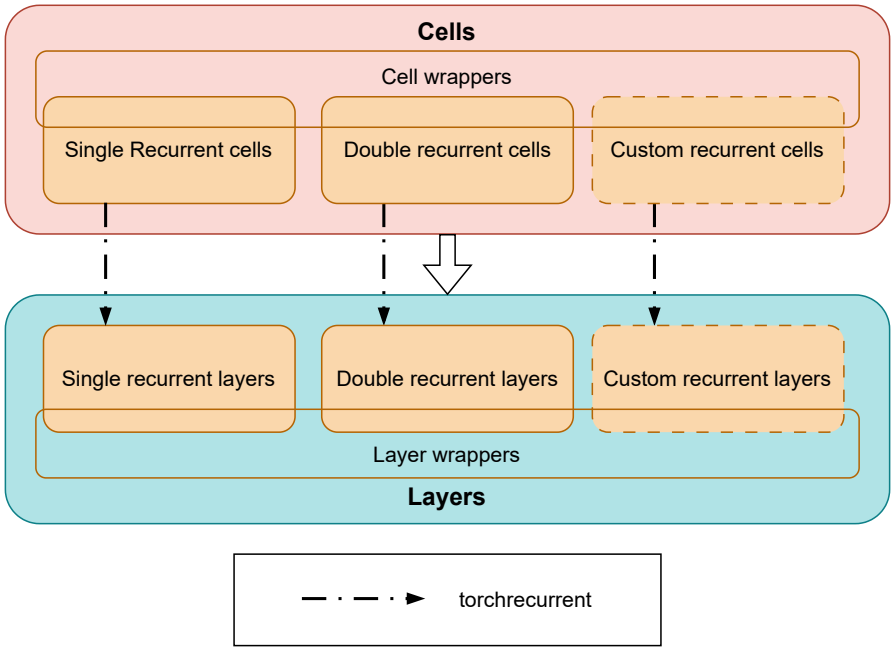

Figure 1: **Structure of recurrent components in the `recurrent` libraries.** Each library defines recurrent *cells* and corresponding *layers*, which can be extended using *wrappers*. Single, double, and custom recurrent architectures share a unified interface, while framework-specific differences (e.g., in `torchrecurrent`) are indicated with dashed arrows.

will evolve toward providing implementations of continuous-time RNNs (De Brouwer et al., 2019; Lechner and Hasani, 2020). Conversely, because of the widespread use of `PyTorch` for large-scale models, `torchrecurrent` may expand to include hybrid architectures based, for instance, on state-space models (Gu et al., 2022a,b).

Modularity and Extensibility In addition to pre-defined models, `recurrent` provides lightweight primitives to simplify the implementation of custom recurrent layers. By standardizing standard boilerplate code for recurrent models, such as parameter initialization, the libraries allow users to focus primarily on defining the forward computation. Once a custom cell is specified using the provided types and interfaces, it can automatically leverage higher-level utilities such as the layers and wrappers described above. Conversely, the packages' modularity allows users to apply custom wrappers to all the provided cells and layers. Figure 1 illustrates the high-level structure that the packages follow. All models in the three libraries follow the identical function signatures as those of the underlying deep learning frameworks. This design choice ensures that new components can be easily integrated into existing workflows without introducing breaking changes.

3 Code Quality

The codebase for all three libraries is open source and hosted on GitHub [3], under a permissive MIT license. `recurrent`'s codebase follows the standards for each language, with clear formatting guidelines, enforced through black in Python and JuliaFormatter in Julia. The libraries can be accessed at the following URLs:

- `torchrecurrent`: github.com/MartinuzziFrancesco/torchrecurrent
- `RecurrentLayers.jl`: github.com/MartinuzziFrancesco/RecurrentLayers.jl
- `LuxRecurrentLayers.jl`: github.com/MartinuzziFrancesco/LuxRecurrentLayers.jl

Each library undergoes rigorous testing through continuous integration using GitHub Actions. Tests are executed for every new package release as well as for each commit in pull requests. `LuxRecurrentLayers.jl` and `RecurrentLayers.jl` are tested against the long-term support, latest, and nightly versions of Julia, while `torchrecurrent` is tested across the range of Python versions supported by `PyTorch` v2.0.

The libraries are accompanied by documentation hosted on GitHub. Currently, the documentation provides a detailed application programming interface (API) reference for each model in the library. Each model entry includes the forward equation implemented by that model, along with a detailed description of its arguments and keyword arguments. The documentation also lists all learnable parameters associated with the model.

4 Conclusion and Outlook

In this paper, we introduce `torchrecurrent`, `RecurrentLayers.jl`, and `LuxRecurrentLayers.jl`, three libraries that provide implementations of a large number of RNNs across multiple DL frameworks in different programming languages. These libraries offer a unified approach to building RNNs, providing APIs that remain fully compatible with the underlying frameworks while extending their available functionality. These features make the presented libraries particularly useful for academic research.

Future development will focus on improving documentation and usability. Although all models are currently documented, their close alignment with the base deep learning libraries minimizes the need for extensive duplication. However, additional examples and use cases would help new users get started more easily. Another planned direction is establishing a centralized repository for RNN benchmarks built on top of these libraries. Finally, while the current focus is on model architectures, future extensions could include alternative training strategies and optimization schemes.

Acknowledgments and Disclosure of Funding

The author thanks Frank Loebe for supporting this work, and David Montero for his help in the visualization. The author acknowledges the financial support by the Federal Ministry of Research, Technology and Space of Germany and by Sächsische Staatsministerium für

3. github.com

Wissenschaft, Kultur und Tourismus in the programme Center of Excellence for AI-research "Center for Scalable Data Analytics and Artificial Intelligence Dresden/Leipzig", project identification number: ScaDS.AI

References

Troy Arcomano, Istvan Szunyogh, Jaideep Pathak, Alexander Wikner, Brian R. Hunt, and Edward Ott. A machine learning-based global atmospheric forecast model. *Geophysical Research Letters*, 47(9), May 2020. ISSN 1944-8007. doi: 10.1029/2020gl087776. URL http://dx.doi.org/10.1029/2020gl087776.

Martin Arjovsky, Amar Shah, and Yoshua Bengio. Unitary evolution recurrent neural networks. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1120–1128, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/arjovsky16.html.

Bo Chang, Minmin Chen, Eldad Haber, and Ed H. Chi. AntisymmetricRNN: A dynamical system view on recurrent neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryxepo0cFX.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2014. doi: 10.3115/v1/d14-1179. URL http://dx.doi.org/10.3115/v1/D14-1179.

Edward De Brouwer, Jaak Simm, Adam Arany, and Yves Moreau. Gru-ode-bayes: Continuous modeling of sporadically-observed time series. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/455cb2657aaa59e32fad80cb0b65b9dc-Paper.pdf.

Daniel Durstewitz, Georgia Koppe, and Max Ingo Thurm. Reconstructing computational system dynamics from neural data with recurrent neural networks. *Nature Reviews Neuroscience*, 24(11):693–710, October 2023. ISSN 1471-0048. doi: 10.1038/s41583-023-00740-7. URL http://dx.doi.org/10.1038/s41583-023-00740-7.

Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, March 1990. ISSN 1551-6709. doi: 10.1207/s15516709cog1402_1. URL http://dx.doi.org/10.1207/s15516709cog1402_1.

Ary L. Goldberger, Luis A. N. Amaral, Leon Glass, Jeffrey M. Hausdorff, Plamen Ch. Ivanov, Roger G. Mark, Joseph E. Mietus, George B. Moody, Chung-Kang Peng, and H. Eugene Stanley. Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals. *Circulation*, 101(23), June 2000. ISSN

1524-4539. doi: 10.1161/01.cir.101.23.e215. URL http://dx.doi.org/10.1161/01.CIR.101.23.e215.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, page 6645–6649. IEEE, May 2013. doi: 10.1109/icassp.2013.6638947. URL http://dx.doi.org/10.1109/ICASSP.2013.6638947.

Albert Gu, Karan Goel, Ankit Gupta, and Christopher Ré. On the parameterization and initialization of diagonal state space models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022a. URL https://openreview.net/forum?id=yJE7iQSAep.

Albert Gu, Karan Goel, and Christopher Re. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022b. URL https://openreview.net/forum?id=uYLFoz1vlAC.

Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. Deep speech: Scaling up end-to-end speech recognition, 2014. URL https://arxiv.org/abs/1412.5567.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, November 1997. ISSN 1530-888X. doi: 10.1162/neco.1997.9.8.1735. URL http://dx.doi.org/10.1162/neco.1997.9.8.1735.

Michael Innes, Elliot Saba, Keno Fischer, Dhairya Gandhi, Marco Concetto Rudilosso, Neethu Mariya Joy, Tejan Karmali, Avik Pal, and Viral Shah. Fashionable modelling with flux. *CoRR*, abs/1811.01457, 2018. URL https://arxiv.org/abs/1811.01457.

Michael I Jordan. Attractor dynamics and parallelism in a connectionist sequential machine. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 8, 1986.

Holger Kantz and Thomas Schreiber. *Nonlinear Time Series Analysis*. Cambridge University Press, November 2003. ISBN 9780511755798. doi: 10.1017/cbo9780511755798. URL http://dx.doi.org/10.1017/CBO9780511755798.

Giancarlo Kerg, Kyle Goyette, Maximilian Puelma Touzel, Gauthier Gidel, Eugene Vorontsov, Yoshua Bengio, and Guillaume Lajoie. Non-normal recurrent neural network (nnrnn): learning long time dependencies while improving expressivity with transient dynamics. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/9d7099d87947faa8d07a272dd6954b80-Paper.pdf.

Frederik Kratzert, Daniel Klotz, Claire Brenner, Karsten Schulz, and Mathew Herrnegger. Rainfall–runoff modelling using long short-term memory (lstm) networks. *Hydrology and Earth System Sciences*, 22(11):6005–6022, November 2018. ISSN 1607-7938. doi: 10.5194/hess-22-6005-2018. URL http://dx.doi.org/10.5194/hess-22-6005-2018.

Ben Krause, Iain Murray, Steve Renals, and Liang Lu. Multiplicative LSTM for sequence modelling, 2017. URL https://openreview.net/forum?id=SJCS5rXFl.

Aditya Kusupati, Manish Singh, Kush Bhatia, Ashish Kumar, Prateek Jain, and Manik Varma. Fastgrnn: A fast, accurate, stable and tiny kilobyte sized gated recurrent neural network. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/ab013ca67cf2d50796b0c11d1b8bc95d-Paper.pdf.

Federico Landi, Lorenzo Baraldi, Marcella Cornia, and Rita Cucchiara. Working memory connections for lstm. *Neural Networks*, 144:334–341, December 2021. ISSN 0893-6080. doi: 10.1016/j.neunet.2021.08.030. URL http://dx.doi.org/10.1016/j.neunet.2021.08.030.

Martin Längkvist, Lars Karlsson, and Amy Loutfi. A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern Recognition Letters*, 42:11–24, June 2014. ISSN 0167-8655. doi: 10.1016/j.patrec.2014.01.008. URL http://dx.doi.org/10.1016/j.patrec.2014.01.008.

Mathias Lechner and Ramin Hasani. Learning long-term dependencies in irregularly-sampled time series, 2020. URL https://arxiv.org/abs/2006.04418.

Shuai Li, Wanqing Li, Chris Cook, Ce Zhu, and Yanbo Gao. Independently recurrent neural network (indrnn): Building a longer and deeper rnn. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, page 5457–5466. IEEE, June 2018. doi: 10.1109/cvpr.2018.00572. URL http://dx.doi.org/10.1109/CVPR.2018.00572.

Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379(2194):20200209, February 2021. ISSN 1471-2962. doi: 10.1098/rsta.2020.0209. URL http://dx.doi.org/10.1098/rsta.2020.0209.

Christopher D. Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA, USA, 1999. ISBN 0262133601.

Francesco Martinuzzi, Chris Rackauckas, Anas Abdelrehim, Miguel D. Mahecha, and Karin Mora. Reservoircomputing.jl: An efficient and modular library for reservoir computing models. *Journal of Machine Learning Research*, 23(288):1–8, 2022. URL http://jmlr.org/papers/v23/22-0611.html.

Francesco Martinuzzi, Miguel D. Mahecha, Gustau Camps-Valls, David Montero, Tristan Williams, and Karin Mora. Learning extreme vegetation response to climate drivers with recurrent neural networks. *Nonlinear Processes in Geophysics*, 31(4):535–557, November 2024. ISSN 1607-7946. doi: 10.5194/npg-31-535-2024. URL http://dx.doi.org/10.5194/npg-31-535-2024.

Avik Pal. Lux: Explicit Parameterization of Deep Neural Networks in Julia, April 2023. URL https://doi.org/10.5281/zenodo.7808903. If you use this software, please cite it as below.

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. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf.

Jaideep Pathak, Zhixin Lu, Brian R. Hunt, Michelle Girvan, and Edward Ott. Using machine learning to replicate chaotic attractors and calculate lyapunov exponents from data. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(12), December 2017. ISSN 1089-7682. doi: 10.1063/1.5010300. URL http://dx.doi.org/10.1063/1.5010300.

Chris Rackauckas, Mike Innes, Yingbo Ma, Jesse Bettencourt, Lyndon White, and Vaibhav Dixit. Diffeqflux.jl - a julia library for neural differential equations, 2019. URL https://arxiv.org/abs/1902.02376.

Mirco Ravanelli, Philemon Brakel, Maurizio Omologo, and Yoshua Bengio. Light gated recurrent units for speech recognition. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(2):92–102, April 2018. ISSN 2471-285X. doi: 10.1109/tetci.2017.2762739. URL http://dx.doi.org/10.1109/TETCI.2017.2762739.

T. Konstantin Rusch and Siddhartha Mishra. Coupled oscillatory recurrent neural network (co{rnn}): An accurate and (gradient) stable architecture for learning long time dependencies. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=F3s69XzWOia.

T. Konstantin Rusch, Siddhartha Mishra, N. Benjamin Erichson, and Michael W. Mahoney. Long expressive memory for sequence modeling. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=vwj6aUeocyf.

P.R. Vlachas, J. Pathak, B.R. Hunt, T.P. Sapsis, M. Girvan, E. Ott, and P. Koumoutsakos. Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics. *Neural Networks*, 126:191–217, June 2020. ISSN 0893-6080. doi: 10.1016/j.neunet.2020.02.016. URL http://dx.doi.org/10.1016/j.neunet.2020.02.016.

Guangyu Robert Yang, Madhura R. Joglekar, H. Francis Song, William T. Newsome, and Xiao-Jing Wang. Task representations in neural networks trained to perform many cognitive tasks. *Nature Neuroscience*, 22(2):297–306, January 2019. ISSN 1546-1726. doi: 10.1038/s41593-018-0310-2. URL http://dx.doi.org/10.1038/s41593-018-0310-2.

Guo-Bing Zhou, Jianxin Wu, Chen-Lin Zhang, and Zhi-Hua Zhou. Minimal gated unit for recurrent neural networks. *International Journal of Automation and Computing*, 13 (3):226–234, June 2016. ISSN 1751-8520. doi: 10.1007/s11633-016-1006-2. URL http://dx.doi.org/10.1007/s11633-016-1006-2.

Kirill Zubov, Zoe McCarthy, Yingbo Ma, Francesco Calisto, Valerio Pagliarino, Simone Azeglio, Luca Bottero, Emmanuel Luján, Valentin Sulzer, Ashutosh Bharambe, Nand Vinchhi, Kaushik Balakrishnan, Devesh Upadhyay, and Chris Rackauckas. Neuralpde: Automating physics-informed neural networks (pinns) with error approximations, 2021. URL https://arxiv.org/abs/2107.09443.