Proceedings of 8th IOE Graduate Conference
Peer Reviewed
ISSN: 2350-8914 (Online), 2350-8906 (Print)
Year: 2020 Month: June Volume: 8

Data-Driven Discovery of Governing Partial Differential Equations

Rojesh Man Shikhrakar [a], Kamal Darlami [b], Laxman Poudel [c]

[a, b, c] *Department of Mechanical Engineering, Pulchowk Campus, IOE, TU, Nepal*
Corresponding Email: [a] 074msmde015.rojesh@pcampus.edu.np, [b] darlami.kd@pcampus.edu.np, [c] laxman@ioe.edu.np

Abstract
We propose a physics-informed neural network(PINN) based framework to discover the governing partial differential equation(PDE) of a given system from data. Given data, our method generalizes a neural network to compute a matrix of candidate terms for PDE. Minimizing the residuals from the candidate matrix allows us to find the coefficients for the equation. We present a framework to discover pde not restricted to first order time derivative equations.

Keywords
data-driven discovery, machine learning, partial differential equations, deep learning, physics-informed neural network

1. Introduction

Scientists have relied on their ability to predict complex phenomena by recording the observations and modeling them into parsimonious mathematical models. But the extraction of these mathematical equations from experimental data, often in the form of differential and partial differential equations (PDEs), is a challenging endeavor, which takes ingenious imagination and years to perfect. These PDEs are ubiquitous over diverse quantitative disciplines, from engineering to basic sciences and from physics to economics, however in many cases these equations are unknown. These equations are commonly derived using first principle approaches satisfying the data observations. Using traditional methods, one can model these observations into an equation but discovering the underlying hidden physical law is much more complex. For example, using Tycho Brahe's planetary data Kepler discovered elliptical orbits but this attractor based law did not reflect the hidden dynamics of the system. The actual governing equation describing the orbital motion was later discovered by Newton.

With the advent of micro-electromechanical systems (MEMS) sensors, it has been easier to collect experimental data in many situations. The ability to derive governing equations for simple to complex natural phenomena using sensors data will be helpful in domains that lack a well-defined quantitative equations. Analyzing such a large amount of data utilize pattern recognition and machine learning methods[1]. However, machine learning models fundamentally assume that data for training as well as testing are from the same distribution. This assumption is more pronounced in deep learning methods [2]. Recent theory-guided data science (TGDS) [3] methods attempt to integrate existing scientific theory into the data science model for inference and prediction.

We present a theory guided data science approach to recover the governing partial differential equations through function approximation using Physics Informed Neural Networks (PINNs). To summarize, the main contributions of this paper we used deep learning based method for the recovery of PDEs as linear combinations of predefined candidate terms for different types of PDE.

This paper is organized as follows. Section 2 presents related and similar research works in the field of discovery of governing equation. In Section 3, the PDE estimation problem is defined formally. Section 4 present the research methodology applied to recover the underlying PDE using neural networks. In Section 5 numerical results and graphs obtained with simulated data is presented. Section 6 discusses on future application and possible improvements on this

method and finally we conclude on section 7.

2. Related Works

There are three mainstream methods developed for discovery of governing PDEs of physical system, viz. symbolic regression, sparse optimization methods, and hybrid frameworks.

2.1 Symbolic Regression

Symbolic regression [4] is an evolutionary computation based method for searching a space of mathematical expressions constructed by pre-defined analytical functions, constant coefficients and algebraic operators, to minimize certain metrics of error. The symbolic regression methods [5, 6] has shown to extract "free-form" physical laws, of any kind in theory, such as the Hamiltonian, Lagrangian, momentum conservation and equations of motion for some simple and crucial physical systems; but, these methods can be extremely time-consuming.

2.2 Sparse Optimization Methods

The sparse optimization methods such as Sparse Identification of Non-linear Dynamics (SINDy) [7] can identify first-order differential equations of non-linear dynamical systems by expressing the first-order time derivative as linear combinations of a candidate functions and determine the unknown coefficients to minimize certain metrics. The sparse-promoting methods are more efficient than symbolic methods and have been generalized to successfully discover partial differential equations as well [8, 9]. SINDy applies finite difference method to approximate the derivatives for estimating the governing equation. Finite difference methods perform poorly in the presence of noise. Furthermore, SINDy is designed to work with first-order time derivative models only.

$$u_t = f(x, u, u_x, u_{xx}, ...; \Theta) \tag{1}$$

where, the subscripts refer to time or spatial variables, $f(\cdot)$ is unknown linear function of u, its partial derivatives and parameters Θ.

2.2.1 Neural Networks Based Methods

As a consequence of universal approximation theorem (UAT) [10, 11], a neural network with some non-linear activation function can approximate any function and its derivatives to an arbitrary degree of accuracy. Hence, many attempts were made to model the physical laws with neural network. Furthermore, to constrain the neural network within the governing equations, the physics-informed neural network (PINNS) [12] can be trained with general non-linear partial differential equations as regularizing priors.

Other methods such as PDE-Net [13] applies convolutional neural networks with filters constrained to finite difference approximations, to learn the form of a PDE without sparsity constraints. Auto-encoders [9] are used to simultaneous discover co-ordinates and governing equations, also linking necessary transformation to the input data to coordinates transformation [14].

2.3 Hybrid Methods

PDE-Net 2.0 [15] builds upon PDE-Net with symbolic regression method combining both sparse optimization and symbolic regression method.

Our method focuses on sparse optimization using Neural networks to discover governing equations not restricted to temporal first-order PDEs.

3. Problem Definition

A PDE relates a function $u : \Omega \to \mathbb{R}$, where $\Omega \in \mathbb{R}^n$ with variables $\mathbf{x} = (x_1, ..., x_n)$ and partial derivatives $D^k u(\mathbf{x}) := \frac{\partial u}{\partial x_1}, ..., \frac{\partial^2 u}{\partial x_1 \partial x_n}, ..., \frac{\partial^k u}{\partial x_n^k}$ where k is the order of PDE. Here, x represent any variable, spatial such as 'x','y','z' or temporal variable 't'. Hence, the PDE has a general form

$$f(\mathbf{x}; u(\mathbf{x}); D^k u(\mathbf{x})) = 0 \tag{2}$$

where, $f(\cdot)$ is a function of $\mathbf{x}$, u, and the partial derivatives.

An example of PDE is wave equation which is a second order hyperbolic PDE represented as

$$\frac{\partial^2 u}{\partial t^2} - c^2 \frac{\partial^2 u}{\partial x^2} = 0$$

where, c is the velocity of wave propagation. Let, c = 1. The equation can be written as $u_{tt} - u_{xx} = 0$.

To search PDE for a system, the space of all possible terms are infinitely large, hence we select a list of L candidate terms $C = (c_1, ..., c_L)$, where candidate terms are functions of variables x_i, dependent variable

$u(x_i)$, and partial differential terms $D^k u(\mathbf{x})$. Then, we construct PDE as the weighted linear combination of these candidate terms, which is of the following form.

$$\sum_{i=1}^{L} \theta_i^* c_i(\mathbf{x}; u(\mathbf{x}); D^k u(\mathbf{x})) = 0 \tag{3}$$

where, $\Theta^* = (\theta_1^*, .., \theta_L^*)$ is unknown vector of coefficients for the linear combination of candidate terms. The central problem being solved by this paper is to determined this coefficient vector Θ.

For wave equation, we can select a list of candidates such as $[u_{tt}, u_{xx}, u_t, u_x, u, u*u_x]$ and so on. The solution should give us Θ^* that is close to $(1, -1, 0, 0, 0, 0)$

4. Research Methodology

4.1 Data Generation

Given data in the form $(\mathbf{x}, u) = ((\mathbf{x_1}, u_1), ..., (\mathbf{x}_N, u_N))$ where u is function of $\mathbf{x}$.

$$u_i = u^*(\mathbf{x}_i) + \varepsilon_i, \qquad i = 1, ..., N \tag{4}$$

Here, noise ε_i is assumed to be zero mean Gaussian with σ^2 variance.

Datasets were generated from symbolic function which are known to be solutions of the PDE. 10,000 sample points are drawn from these datasets using latin-hypercube sampling method to get near-random sample of data in multiple dimension. A validation set was split from the sample with test split of 0.2.

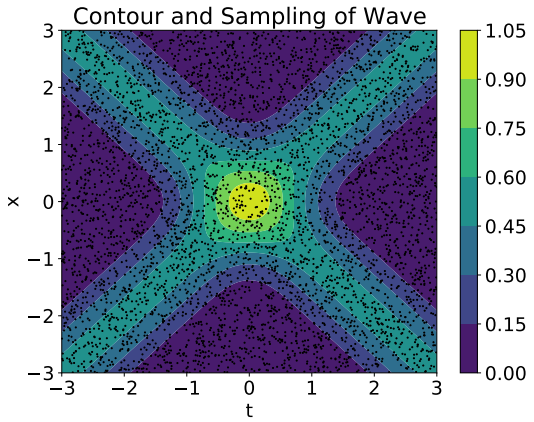

Figure 1: Sampled points (black dots) from contour of Wave Equation, $u_{tt} - c^2 u_{xx} = 0$, where c=1

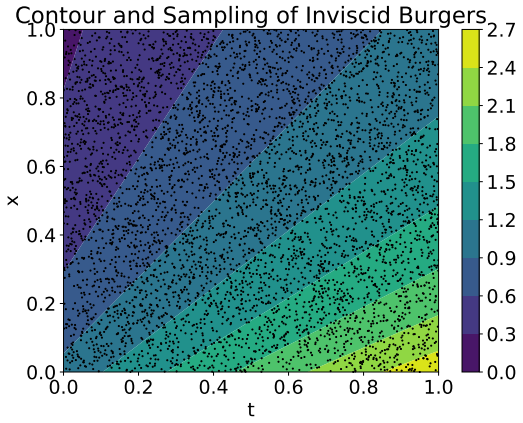

Figure 2: Sampled Points (black dots) from contour of Inviscid Burgers Equation, $u_t + u * u_x = 0$

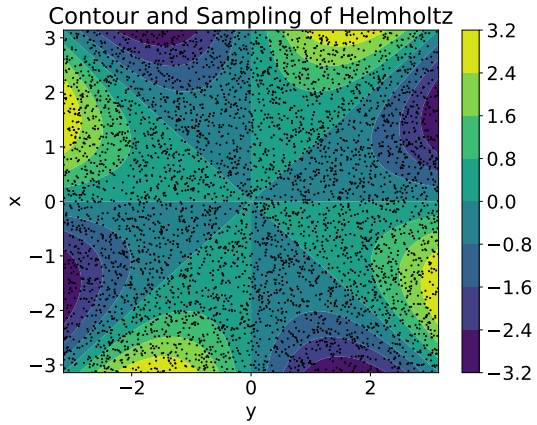

Figure 3: Sampled Points (black dots) from contour of Helmholtz Equation, $u_{xx} + u_{yy} + k^2 u = 0$, where k=1

4.2 Model Architecture

A multi-layer perceptron (MLP) with 4 hidden-layers each consisting of 50 hidden units. Softplus is used as activation function in all layers except the output layer which is just a linear unit.

4.3 Model Fitting

Root mean square error (RMSE) was used for training the neural network, since it has the same units as the output variable and can give a quick indication of how the model is performing.

$$\mathscr{L}_u(\mathbf{x}; W) = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (u_i(x) - \hat{u}_i(\mathbf{x}; W))^2} \tag{5}$$

where W are neural networks parameters and $\hat{u}$ is the prediction of the neural network. This is later regularized by additional regularization terms.

4.4 Regularization

Physics-based Regularization and Model Discovery The correct PDE solution, $\hat{u}^*(\mathbf{x};W)$ should satisfy PDE Eq. (3) for every point $x \in \Omega$, a technique used as regularization prior in PINNs [12]. Since, neural networks can also approximates gradients with arbitrary accuracy under mild assumptions on the activation function [11]. With better generalized model we can estimate differentials through auto-differentiation (auto-diff) with less truncation errors compared to numerical differentiation for higher order differentials.

Each term in the candidate list is evaluated over M collocation point through auto-diff to obtain matrix $H \in \mathbb{R}^{M\times L}$ and PDE in Eq. (3) transforms into linear system of equations. In the experiments 1000 collocation points were sampled from within the same domain.

$$\begin{bmatrix} | & & | \\ C_1(\mathbf{x},\hat{u}_i,D^k\hat{u}_i) & \dots & C_L(\mathbf{x},\hat{u}_i,D^k\hat{u}_i) \\ | & & | \end{bmatrix} \begin{bmatrix} \theta_1^* \\ \vdots \\ \theta_L^* \end{bmatrix} \quad (6)$$

For brevity $\hat{u}_i(\mathbf{x};W)$ has been written as $\hat{u}_i$. Where i = 1..M collocation point. For each collocation point we calculate the derivatives using auto-diff and estimate values for each candidate term in the list and form the matrix H.

$$H(x,\hat{u})\Theta^* = 0 \quad (7)$$

To determine Θ^*, we solve the above linear system of equation of the form $Ax = 0$. Since, the null space of M consists of all solutions Θ.

$$Null(H) = \{\Theta^* \in \mathbb{R}^L \mid H\Theta^* = \vec{0}\}$$

Equivalently, Θ^* is singular vector of H associated with the singular value 0. According to min-max theorem for singular values [16], the minimum of $\|H\Theta\|$, subjected to $\|\Theta\|_2 = 1$, is the smallest singular value of H. This constraint achieved through normalization also prevents the obvious solution $\Theta = 0$. Hence, the function residual loss is

$$\mathscr{L}_f(\mathbf{x};\hat{u};\Theta) = \|H\Theta\|_2^2, \; with \; \|\Theta\|_2 = 1 \quad (8)$$

Determining Θ having minimum residual $\|H\Theta\|$ also have the physical interpretation that we want the discovered equation to satisfies the data very well.

Parsimony Laws of nature are inherently simpler and parsimonious i.e. the governing equation consist of only few terms. Hence, to choose sparse form from relatively large space of potential candidate terms, we additionally impose L_1 sparsity on the Theta.

$$\mathscr{L}_{norm} = \|\Theta\|_1 \quad (9)$$

4.5 Combined Regularization and loss

$\mathscr{L}_u$ can be further used as an additional regularizer when multiplied with the other regularization term [17]. Here, λ_u, λ_f and λ_{norm} are coefficients for each loss function, used as additional hyper-parameters to the network. We estimated W and Θ with the minimization of $\mathscr{L}_{total}$.

$$\mathscr{L}_{total} = \lambda_u\mathscr{L}_u(1 + \lambda_f\mathscr{L}_f + \lambda_{norm}\mathscr{L}_{norm}) \quad (10)$$

5. Results and Discussion

5.1 Training Results

For training the model Adam optimizer with learning rate of 0.02 for the coefficient Θ and 0.01 for Neural Networks weights W with exponentially decay at rate of 0.9998 was used. The models were trained for 50,000 epochs as model prediction kept improving even after convergence.

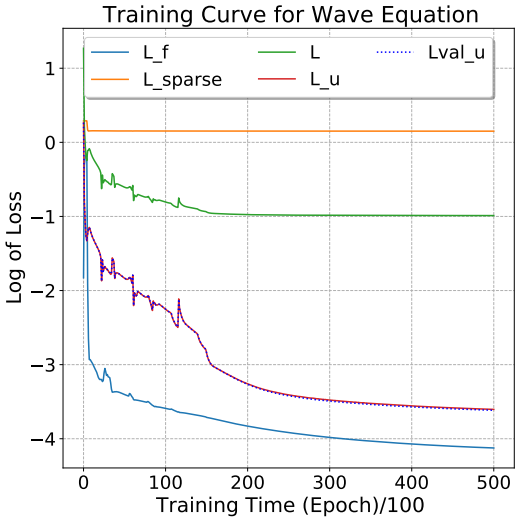

Figure 4: Training Curve for Wave Equation

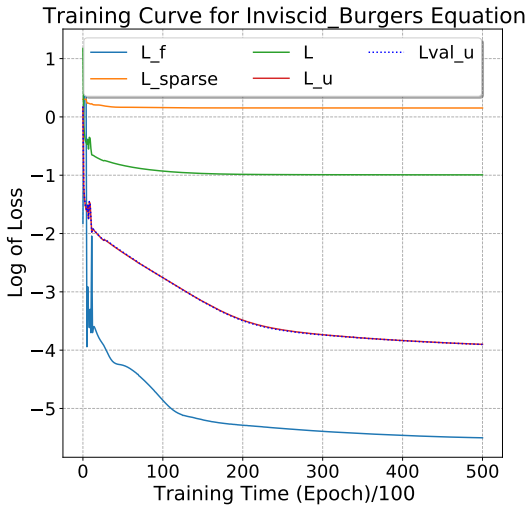

Figure 5: Training Curve for Inviscid Burgers Equation

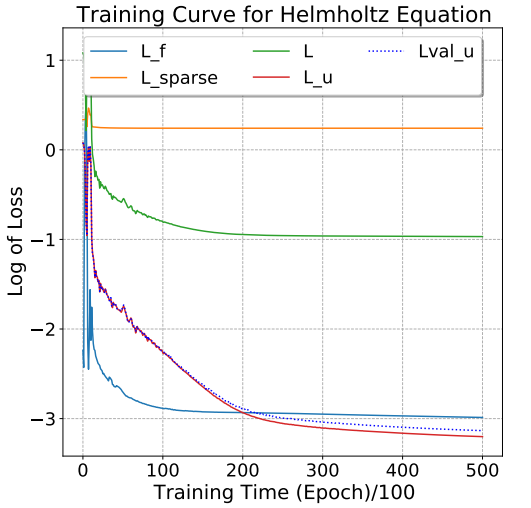

Figure 6: Training Curve for Helmholtz Equation

Fig. 4, 5 and 6, all shows similar trend in training. The $\mathscr{L}_u$ written as L_u decreases below 10^{-3} and converges. The random zigzag pattern indicates that the learning rate, and other hyper-parameters might not be optimal. Hence, different equations are likely to have dissimilar hyper-parameters but for consistency of the presentation we have used same hyper parameters for all experiments. A slight difference in residual $\mathscr{L}_f$ written as L_f can be seen, for wave and Inviscid Burgers equation, the residual is far less compared to Helmholtz. Low residual might indicate correct identification of the coefficient vector or a local minima with low residual from a different set of coefficient vector. $\mathscr{L}_sparse$ and $\mathscr{L}_total$ written as L_sparse and L attain a nearly constant value.

The coefficient vector is also taken as parameter which are stepped during the loss minimization.

5.2 Evaluation

To compare accuracy of equation discovered from this method with the correct form, we used Sqrt-Cosine Error as to determine the equation recovery error, sometimes also referred as error in law. We used cosine similarity as the measure of compare between the two coefficient vectors rather than using L_2 norm distance which are more commonly used.

$$\text{Error in Law}(\Theta^*,\Theta) = \sqrt{1-cosine(\Theta^*,\Theta)} \quad (11)$$

We have the correct coefficient vector Θ^* as we know the partial differential equation and we estimate Θ using above methods.

PDE	Candidate List	Recovery Error
Wave Equation $u_{tt} - u_{xx} = 0$ $x,t \in [-3,3]$	u_{tt}, u_{xx}, u_t, u_x u, u^2, uu_x	$(1.42 \pm 0.024) \times 10^{-3}$
Invisicd Burgers $u_t + u * u_x = 0$ $x,t \in [0,1]$	$u_{tt}, u_{xx}, u_t, u_x,$ u, u^2, uu_x, u_x^2	$(6.43 \pm 0.13) \times 10^{-3}$
Helmholtz $u_{xx} + u_{yy} + u = 0$ $x,y \in [-\pi,\pi]$	$u_{xx}, u_{yy}, u_{yx},$ $u_y, u_x, u, u^2,$ uu_x, uu_y	$(3.51 \pm 0.12) \times 10^{-3}$

Table 1: Table shows average and standard deviation of error in estimating the governing equation

The table (1) and fig. (7) shows the error in prediction (recovery) of governing equation calculated using eq. (11). All the values are in order of 10^{-3} or less which indicates that the vectors coefficients are very close to the correct vector. This values are similar to [17] as it uses similar metrics and loss functions. This model performs also performs well in comparison to [8] for discovery of PDE which uses euclidean distance for measuring error in recovery of the governing equation.

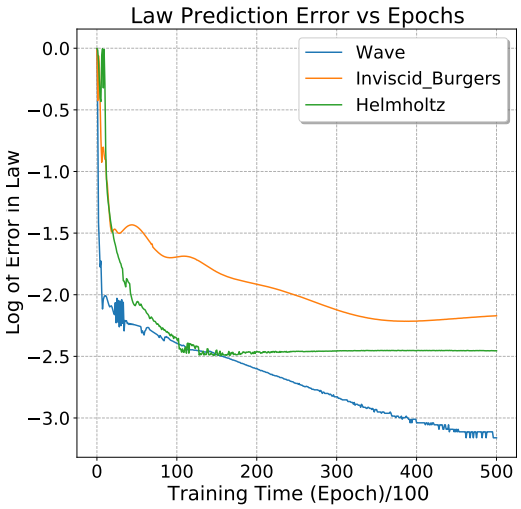

Figure 7: Plot of Log of Error in law throughout one particular training session

The fig. (7) also shows that the Helmholtz equation being an elliptic form of PDE converges early compared to other equations which have temporal components.

5.3 Validation

A validation dataset was split after data sampling with train test split ratio of 0.2. Root mean squared loss similar to Eq. (5), is calculated and plotted in training curve. The model performed very well on the data within the domain. Model validation was done only within the domain from which the data was sampled but not outside the domain. In fig. 4, 5 and 6, the line for Lval_u which is RMSE value calculated using same equation in Eq. (5). The Lval_u curve follows closely with L_u, curve.

Furthermore, to validate our method with No Free Lunch (NFL) Theorem [18], we tested the method with three different forms of partial differential equations. Wave equation as a form of hyperbolic PDE, Helmholtz equation as a form of elliptic PDE, and Inviscid Burgers equation which is a form of first order quasi-linear hyperbolic equation was used. Models performed well in all three forms of equations.

In addition, error in estimating the gradient using automatic differentiation was calculate using the actual gradient estimated from symbolic gradient estimation. The error was within 2% for first order gradients and within 5% for second order gradients.

5.4 Discussion

There are still many unanswered questions that needs further research. Firstly, some affect of candidate function choices was visible during experimentation which was not thoroughly investigated. Certain candidate function might lead to erroneous results. Previous domain knowledge in the field could help select appropriate candidate terms list. There needs further investigation on use of functions and constants in the candidate list.

Furthermore, higher dimension equations and other forms of partial differential equations can be considered. We can also make use of Sobolev training to fit the target value as well as higher order derivatives of the target value [19], which would improve the quality of the regressor.

6. Conclusion

We present a framework to recover governing partial differential equations in general form for the case where discovery of PDE from first principles is intractable. We present a framework to discover pde not restricted to first order time derivative equations. There are still number of open issues and missing gaps that could be further studied and improved.

Acknowledgments

The authors are grateful to Department of Mechanical Engineering, Pulchowk Campus, IOE, TU for all the support and guidance in this research work.

References

[1] Christopher M. Bishop. *Pattern Recognition and Machine Learning*, volume 1. Springer, 2006.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[3] Anuj Karpatne, Gowtham Atluri, James H. Faghmous, Michael Steinbach, Arindam Banerjee, Auroop Ganguly, Shashi Shekhar, Nagiza Samatova, and Vipin Kumar. Theory-guided data science: A new paradigm for scientific discovery from data. *IEEE Transactions on Knowledge and Data Engineering*, 29(10):2318–2331, 2017.

[4] John R. Koza. *Genetic programming: On the programming of computers by means of natural selection,*, volume 1. MIT Press, Cambridge, MA, 1992.

[5] Josh Bongard and Hod Lipson. Automated reverse engineering of nonlinear dynamical systems.

Proceedings of the National Academy of Sciences of the United States of America, 104(24):9943–9948, 2007.

[6] Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009.

[7] Steven L. Brunton, Joshua L. Proctor, J. Nathan Kutz, and William Bialek. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences of the United States of America*, 113(15):3932–3937, 2016.

[8] Samuel H. Rudy, Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Data-driven discovery of partial differential equations. *Science Advances*, 3(4), 2017.

[9] Kathleen Champion, Bethany Lusch, J. Nathan Kutz, and Steven L. Brunton. Data-driven discovery of coordinates and governing equations. *Proceedings of the National Academy of Sciences of the United States of America*, 116(45):22445–22451, 2019.

[10] Kurt Hornik. Multilayer Feedforward Networks are Universal Approximators. *Neural Networks*, 2:359–366, 1989.

[11] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4(2):251–257, 1991.

[12] 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.

[13] Zichao Long, Yiping Lu, Xianzhong Ma, and Bin Dong. PDE-Net: Learning PDEs from Data. In *Proceedings of the 35th International Conference on Machine Learning, PMLR*, pages 3208–3216, 2018.

[14] Jens Berg and Kaj Nyström. Data-driven discovery of PDEs in complex datasets. *Journal of Computational Physics*, 384:239–252, 2019.

[15] Zichao Long, Yiping Lu, and Bin Dong. PDE-Net 2.0: Learning PDEs from data with a numeric-symbolic hybrid deep network. *Journal of Computational Physics*, 399, 2019.

[16] Charles F Van Loan and Gene H Golub. *Matrix computations*. Johns Hopkins University Press, 3 edition.

[17] Ali Hasan, João M. Pereira, Robert Ravier, Sina Farsiu, and Vahid Tarokh. Learning Partial Differential Equations from Data Using Neural Networks. *arXiv*, 2019.

[18] David H Wolpert and William G Macready. No Free Lunch Theorems for Optimization. Technical Report 1, 1997.

[19] Wojciech Marian Czarnecki, Simon Osindero, Max Jaderberg, Grzegorz Swirszcz, and Razvan Pascanu. Sobolev Training for Neural Networks. In *31st Conference on Neural Information Processing Systems (NIPS)*, 2017.