The Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21)

Differential Spectral Normalization (DSN) for PDE Discovery

Chi Chiu So, Tsz On Li, Chufang Wu, Siu Pang Yung
The University of Hong Kong
kelccso@connect.hku.hk, toli2@cs.hku.hk, wucf@connect.hku.hk, spyung@hku.hk

Abstract

Partial differential equations (PDEs) play a prominent role in many disciplines for describing the governing systems of interest. Traditionally, PDEs are derived based on first principles. In the era of big data, the needs of uncovering PDEs from massive data-set are emerging and become essential. One of the latest advance in PDE discovery models is PDE-Net, which has shown promising predictive power with its moment-constrained convolutional filters, but may suffer from noisy data and numerical instability intrinsic in numerical differentiation. We propose a novel and robust regularization method tailored for moment-constrained convolutional filters, namely, Differential Spectral Normalization (DSN), to allow accurate estimation of coefficient functions and stable prediction of dynamics in a long time horizon. We investigated the effectiveness of DSN against batch normalization, dropout, spectral normalization, weight decay, weight normalization, jacobian regularization and orthonormal regularization and supported with empirical evidence that DSN owns the highest effectiveness by learning the convolutional filters in a robust manner. Numerical experiments further reveal that with DSN there is a substantial potential to uncover the hidden PDEs in a scarce data setting and predict the dynamical behavior for a long time horizon, even in a noisy environment where all data samples are contaminated with noise.

1 Introduction

Partial differential equations (PDEs) play a significant role in many scientific disciplines for describing the dynamics in the systems of interest. Traditionally, PDEs are derived based on first principles. For example, the Navier-Stokes equations in fluid dynamics are derived from the conservation laws of mass, momentum and energy; the Black-Scholes equations in option pricing theory are derived from the geometric brownian motion assumption of stock price dynamics. However, these first principles are not always available as the governing physical laws may be unknown or the dynamical systems are too complicated. Given the recent advances of machine learning theories, the rapid development of computational powers and the availability of vast quantities of data, a data-driven approach to uncover the hidden PDEs in dynamical systems was emerging and popular in the past decade.

Given measurements of $u\left(x_1,\ldots,x_S\right)$ over certain temporal and spatial domains, $\{u\left(t,x_1,\ldots,x_S\right) : t \in \{0,\Delta t,2\Delta t,\ldots,T\} \subset \mathbb{R}, (x_1,\ldots,x_S) \in \Omega \subset \mathbb{R}^S\}$, the objective of PDE discovery problem is to uncover equation (1) governing the observed system. Equation (1) expresses a generic form of PDE with S spatial dimensions and 1 temporal dimension of order K.

$$\frac{\partial u}{\partial t} = f\Bigg(x_1,x_2,\ldots,x_S,u,\frac{\partial u}{\partial x_1},\frac{\partial u}{\partial x_2},\ldots,\frac{\partial u}{\partial x_S}, \frac{\partial^2 u}{\partial x_1^2},\frac{\partial^2 u}{\partial x_2^2},\ldots,\frac{\partial^2 u}{\partial x_S^2},\frac{\partial^2 u}{\partial x_1 \partial x_2},\ldots,\frac{\partial^K u}{\partial x_S^K}\Bigg) \tag{1}$$

It was recently shown that residual network (ResNet) (He et al. 2016a,b) has close relation with PDEs and may be suitable for PDE modeling and discovery (Haber and Ruthotto 2017; Sonoda and Murata 2017; Qin, Wu, and Xiu 2019; Wu and Xiu 2020). Exploiting the architecture of convolutional ResNet, Long et al. (2017); Long, Lu, and Dong (2019) introduced PDE-Net which trains convolutional filters (abbr. filters) with moment constraints and weights matrices to approximate the derivatives terms and variable coefficients of PDE without assumptions on the explicit form of the PDE being known. PDE-Net possesses high interpretability and requires zero prior knowledge on the PDE to be discovered, but it lacks regularization on its filters and suffers from numerical instability intrinsic in numerical differentiation (Burden and Faires 2004; Baydin et al. 2017).

In the PDE-Net approach, (1) is discretized to

$$\frac{\partial u}{\partial t} \approx \sum_{i_1=0,i_2=0,\ldots,i_S=0}^{i_1+i_2+\ldots,+i_S=K} A_{i_1,i_2,\ldots,i_S}\left(D_{i_1,i_2,\ldots,i_S} \circledast \vec{u}\right) + \text{Non-linear terms} \tag{2}$$

where

- $D_{i_1,i_2,\ldots,i_S}$ are filters. Under moment constraints, $D_{i_1,i_2,\ldots,i_S} \circledast \vec{u}$ acts as a discretization of $\frac{\partial^{i_1+i_2\cdots+i_S} u}{\partial x_1^{i_1} \partial x_2^{i_2} \ldots \partial x_S^{i_S}}$.
- $A_{i_1,i_2,\ldots,i_S}$ are weight matrices discretizing the coefficient functions.
- Non-linear terms include $\left(D_{2,0,\ldots,0} \odot D_{0,1,\ldots,0}\right) \circledast \vec{u}$ which models $\frac{\partial u^2}{\partial x_1^2} \cdot \frac{\partial u}{\partial x_2}$ with $\odot$ being hadamard product.

- $\vec{u}$ is the discretized u.

To simplify notations, we use $\vec{u}_t$ to denote the discretized u at time t and drop the non-linear terms in equation WLOG. Equation (3) is derived from equation (2) with time step Δt under forward Euler method.

$$\vec{u}_{t+\Delta t} = \vec{u}_t + \Delta t \sum_{i_1,i_2,\dots,i_S} A_{i_1,i_2,\dots,i_S} \left(D_{i_1,i_2,\dots,i_S} \circledast \vec{u}_t\right) \tag{3}$$

Equation (3) expresses a prediction from time t (input) to $t+\Delta t$ (output), modeled as a convolutional layer ($D_{i_1,i_2,\dots,i_S}$) stacked with a fully connected layer ($A_{i_1,i_2,\dots,i_S}$) under a residual connection. Likewise, a prediction from time t to $t+n\Delta t$ proceeds via iterating equation (3), forming a ResNet with $2n$ layers and n residual connections, which is called the PDE-Net.

In this paper, we propose a novel regularization method tailored for moment-constrained filters, Differential Spectral Normalization (DSN), which enhances the robustness of filters to enable more accurate estimation of coefficients and more stable prediction of the dynamical behavior. DSN regularizes the spectral norm of the filters and can be streamlined in a fast and parallel algorithm proposed in (Sedghi, Gupta, and Long 2018), and possesses several unique properties. Firstly, DSN is carefully designed to be compatible with moment constraints which characterize filters to function as differential operators. This is warranted by our theorem which proves the invertibility of the mapping between moment-constrained filters and moment tensors which contain moments as its entries.Moreover, DSN carefully considers the physical meanings of filters and weight matrices when designing their Lipschitz constant hyper-parameter and when judging whether they are eligible for normalization. Our main contribution is DSN, the first normalization technique tailored for moment-constrained filters.

The rest of the paper is structured as follows. Section 2 introduces the background of PDE discovery problem. Section 3 presents an error analysis by considering the Lipschitz constants in PDE-Net, which motivates our proposal of DSN. The definitions and physical meanings of DSN will be discussed. Section 4 introduces common regularization techniques which will be compared. Section 5 covers our empirical studies on a linear PDE (convection-diffusion equation) and a nonlinear PDE (KdV equation), followed by a discussion on the effectiveness of DSN against other regularization methods. Section 6 is our conclusion.

2 Related Work

Different approaches have been attempted in data-driven discovery of PDEs. One of the major approaches is sparse regression, which constructs a predefined large candidate library of simple functions and their partial derivatives and leverages sparsity promoting techniques, to identify a small number of those terms in the library to best represent the data (Brunton, Proctor, and Kutz 2016; Rudy et al. 2017; Schaeffer 2017; Schaeffer et al. 2013). Such approach suffers from numerical instability intrinsic in numerical differentiation and faces challenges in front of noisy data (Burden and Faires 2004; Baydin et al. 2017). The use of finite difference filters common in this approach further limits the expressive and predictive power of the dictionary (Long et al. 2017; Long, Lu, and Dong 2019).

To alleviate the problem of numerical instability, (Raissi, Perdikaris, and Karniadakis 2017b,a) leveraged automatic differentiation which enables accurate evaluation of derivatives at machine precision to introduce the physics-informed neural network (PINN) which is capable to discover the scalar coefficients of PDE but requires the assumption that the explicit form of the PDE is known. To bypass this strong assumption, several attempts have been made. Raissi (2018) constructed a second neural network to approximate the unknown solution besides PINN. Berg and Nyström (2019) constructed a neural network for data modeling and feed a second neural network with the automatic derivatives from the first network to approximate the PDE. Both attempts fall short in interpretability of the learned models. Xu, Chang, and Zhang (2019) made another attempt by first constructing a neural network for generation of "meta-data" and apply sparse regression on the automatic derivatives of the "meta-data". Although this approach allows higher interpretability, parsimony of the fictitious "meta-data" may not be guaranteed.

Among popular regularization techniques, a recent advance was spectral normalization (SN) (Yoshida and Miyato 2017; Miyato et al. 2018), which regularizes the spectral norm of filters and weight matrices and was suggested to be an effective regularization for GAN models. Sedghi, Gupta, and Long (2018) leveraged the properties of circulant matrices to introduce an algorithm for fast computation of spectral norm of filters, based on which Singla and Feizi (2019) further developed numerical methods for estimating upper bounds on spectral norm. Saito, Matsumoto, and Saito (2017); Jia et al. (2017), on the other hand, developed singular value clipping.

Other common regularization techniques include weight decay (Krogh and Hertz 1992), weight normalization (Salimans and Kingma 2016), Jacobian regularization (Gu and Rigazio 2014), orthonormal regularization (Brock et al. 2016), batch normalization (Ioffe and Szegedy 2015) and dropout (Srivastava et al. 2014). While weight decay (aka. Frobenius norm regularization) regularizes the sum of squared singular values, weight normalization normalizes the l_2 norm of each row vector in the weight matrices. It can be shown that weight normalization is equivalent to weight decay up to a constant. Jacobian regularization instead regularizes the l_2 norm of layerwise Jacobians. For PDE-Net, which is entirely linear, Jacobian regularization degenerates to weight decay. Orthonormal regularization regularizes $\|W^\intercal W - I\|_F^2$ which sets all singular values to one. All these approaches attempt to regularize all or sum of the singular values instead of just the largest one, destroying much of the information about the spectrum of the weight matrices. On the other hand, batch normalization and dropout are well established regularization methods but do not suit our model well. We will discuss more details in section 4.

3 Our Approach

We start with an error analysis on PDE-Net for insight in regularization, motivated by which, we then propose a novel regularization technique, Differential Spectral Normalization (DSN).

Error Analysis

Our analysis focuses on error and noise propagation in PDE-Net, i.e. iterations of equation (3).There are two main sources of errors and noises. Firstly, all our data are collected from noisy environment, which is an inevitable assumption for real-world problems, and we assume no prior knowledge on their characteristic. Secondly, our model is full of round-off and truncation errors, which are induced by the limited precision of computation intrinsic in computation machines and numerical differentiation (Burden and Faires 2004; Baydin et al. 2017). These two types of errors and noises magnify in each layer and PDE-Net may turn ill-conditioned if no proper regularization is applied.

Based on equation (3), we expressed each iteration as an application of F with F defined as

$$F = I + \Delta t \sum_{i_1, i_2, \ldots, i_S} A_{i_1, i_2, \ldots, i_S} \left(D_{i_1, i_2, \ldots, i_S} \circledast \right) \tag{4}$$

To predict u at time $n\Delta t$ from time 0, we perform F on u_0 for n times. A robust F should reduce the noise and errors generated in each time step and calm down errors accumulated from previous time steps. To simplify notations, we use $\vec{u}_t$, $\hat{\vec{u}}_t$ and $\tilde{\vec{u}}_t$ to denote the true value, noisy observations and predictions at time t respectively. We start by looking at the loss function L, which compares the noisy observation with prediction,

$$L = \left\| \hat{\vec{u}}_T - \tilde{\vec{u}}_T \right\|_2 \tag{5}$$

A careful observation on the loss function (5) reveals

$$\begin{aligned} \left\| \hat{\vec{u}}_T - \tilde{\vec{u}}_T \right\|_2 &= \left\| \hat{\vec{u}}_T - F\left(\vec{u}_{T-\Delta t} + \epsilon_{T-\Delta t}\right) \right\|_2 \\ &= \Big\| \hat{\vec{u}}_T - F\left(\vec{u}_{T-\Delta t}\right) + F\left(\vec{u}_{T-\Delta t}\right) - \\ &\quad F\left(\vec{u}_{T-\Delta t} + \epsilon_{T-\Delta t}\right) \Big\|_2 \\ &\le \left\| \hat{\vec{u}}_T - F\left(\vec{u}_{T-\Delta t}\right) \right\|_2 + \\ &\quad \left\| F\left(\vec{u}_{T-\Delta t}\right) - F\left(\vec{u}_{T-\Delta t} + \epsilon_{T-\Delta t}\right) \right\|_2 \end{aligned} \tag{6}$$

where ϵ_t is the noises and errors accumulated till time t. It shows that L is bounded by two separate loss functions in equation (6), the former of which minimizes the round-off and truncation errors generated in the last time step, the latter mitigates the noise and errors accumulated from previous time steps.

To minimize the first loss term in equation (6), $\left\| \hat{\vec{u}}_T - F\left(\vec{u}_{T-\Delta t}\right) \right\|_2$, our best effort is to apply F to the observed value $\hat{\vec{u}}_{T-\Delta t}$ and compare with the observed value at time T for training, as we have no knowledge about the true value $\vec{u}$. For the second loss term $\left\| F\left(\vec{u}_{T-\Delta t}\right) - F\left(\vec{u}_{T-\Delta t} + \epsilon_{T-\Delta t}\right) \right\|_2$, we examine it in further depth as follows,

$$\begin{aligned} & \left\| F\left(\vec{u}_{T-\Delta t}\right) - F\left(\vec{u}_{T-\Delta t} + \epsilon_{T-\Delta t}\right) \right\|_2 \\ = & \left\| F\left(\vec{u}_{T-\Delta t}\right) - F\left(\vec{u}_{T-\Delta t}\right) - F\left(\epsilon_{T-\Delta t}\right) \right\|_2 \\ = & \left\| F\left(\epsilon_{T-\Delta t}\right) \right\|_2 \\ \le & \left\| F \right\|_2 \left\| \epsilon_{T-\Delta t} \right\|_2 \end{aligned} \tag{7}$$

The first equality is warranted by the linearlity of F, because unlike common neural network with non-linearity, PDE-Net is entirely linear with activation functions being identity.

As we assumed no prior knowledge on the characteristics of noises and errors, equation (7) tells us our focus should be on regularizing the spectral norm (2-norm) of F. Further analyzing equation (7) gives

$$\begin{aligned} & \left\| F \right\|_2 \\ = & \left\| I + \Delta t \sum_{i_1, i_2, \ldots, i_S} A_{i_1, i_2, \ldots, i_S} \left(D_{i_1, i_2, \ldots, i_S} \circledast \right) \right\|_2 \\ \le & 1 + \left\| \Delta t \sum_{i_1, i_2, \ldots, i_S} A_{i_1, i_2, \ldots, i_S} \left(D_{i_1, i_2, \ldots, i_S} \circledast \right) \right\|_2 \\ = & 1 + \left\| \Delta t \sum_{i_1, i_2, \ldots, i_S} A_{i_1, i_2, \ldots, i_S} W_{i_1, i_2, \ldots, i_S} \right\|_2 \\ \le & 1 + \Delta t \sum_{i_1, \ldots, i_S} \left\| A_{i_1, \ldots, i_S} \right\|_2 \left\| W_{i_1, \ldots, i_S} \right\|_2 \end{aligned} \tag{8}$$

where $W_{i_1, i_2, \ldots, i_S}$ is the matrix representation of $D_{i_1, i_2, \ldots, i_S} \circledast$.

A direct examination of equation (8) reveals that we should regularize the spectral norms of $W_{i_1, i_2, \ldots, i_S}$ and $A_{i_1, \ldots, i_S}$. However, $A_{i_1, i_2, \ldots, i_S}$ and $D_{i_1, i_2, \ldots, i_D}$ carry certain physical meanings. We have to handle two of them independently.

- $A_{i_1, i_2, \ldots, i_S}$ carries its physical meaning as discretized coefficient functions. As we assume no prior knowledge on the characteristics of the coefficient functions, it is unfeasible to apply a regularization on its norm.
- $D_{i_1, i_2, \ldots, i_S}$ carries its physical meaning as discretized differential operators. As PDE-Net applies moment constraints to characterize the filters to act as differential operators, a regularization method which is compatible with the moment constraints must be proposed.

Differential Spectral Normalization

Motivated by the above analysis, we propose Differential Spectral Normalzation (DSN) dedicated to training of filters.

Definition 1 (Differential Spectral Normalization (DSN))**.**

$$W_{DSN\, i_1, \ldots, i_S} \triangleq \left\| \hat{W}_{i_1, \ldots, i_S} \right\|_2 \frac{W_{i_1, \ldots, i_S}}{\left\| W_{i_1, \ldots, i_S} \right\|_2} \tag{9}$$

where $\hat{W}_{i_1, \ldots, i_S}$ is the symmetric finite difference filter of order $i_1, \ldots, i_S$. For example, in the case of a single spatial dimension, $\hat{W}_1$ is the matrix representation of filter $[1/12, -2/3, 0, 2/3, -1/12]$.

There are several important points to remark in the definition of DSN.

1. As there is no theoretical guarantee that the optimal spectral norm size of moment-constrained filter is 1, we take $\left\|\hat{W}_{i_1,\ldots,i_S}\right\|$ to be the size of the normalize filter, to ensure that training of filters proceed in a non-empty filter space. In other words, we are worried that the space of filters with unit norm size may has empty intersection with the space of moment-constrained filters. Therefore, we have the equality
$$\|W_{\text{DSN}\,i_1,\ldots,i_S}\|_2 = \left\|\hat{W}_{i_1,\ldots,i_S}\right\|_2 \tag{10}$$

2. Here, we drop the index $i_1,\ldots,i_S$ for simplicity. The gradient of W_{DSN} over the i,j entry of W is
$$\frac{\partial W_{\text{DSN}}}{\partial (W)_{ij}} = \frac{1}{\|W\|_2}\left[\left\|\hat{W}\right\|_2 E_{ij} - W_{\text{DSN}}\left(\eta\xi^{\intercal}\right)_{ij}\right] \tag{11}$$
where E_{ij} is a matrix with 1 in its i,j entry and 0 everywhere else, and η and ξ are the first left and right singular vectors of W.
The gradient of loss function L with respect to W is
$$\frac{\partial L}{\partial W} = \frac{1}{\|W\|_2}\left\{\left\|\hat{W}\right\|_2 \mathbb{E}\left[\delta\hat{\tilde{u}}^{\intercal}\right] - \mathbb{E}\left[\delta^{\intercal} W_{\text{DSN}}\hat{\tilde{u}}\right]\eta\xi^{\intercal}\right\} \tag{12}$$
where $\delta = \frac{\partial L}{\partial W_{\text{DSN}}\hat{\tilde{u}}}$, and $\mathbb{E}\left[\cdot\right]$ is the sample mean over a batch. $\mathbb{E}\left[\delta^{\intercal} W_{\text{DSN}}\hat{\tilde{u}}\right]$ in equation (12) acts as an adaptive hyper-parameter controlling the regularization term.

3. Another important feature of DSN lies on its implementation. As moment constraints fix some of the moments of the filters to be 0 or 1 based on the order of differential operators being approximated, a direct application of equation (12) may alter the fixed moments and is unfeasible. To bypass this challenge, we utilize moment tensor which stores the moments of a filter as their entries. Dong, Jiang, and Shen (2017); Long et al. (2017); Long, Lu, and Dong (2019) have introduced the concept of vanishing moment and moment matrix for 2D filters and a mapping from 2D filters to moment matrix, but little was said about its higher-dimensional generalization (moment tensor) and the existence of its inverse mapping. We supply a definition of its high-dimensional generalization and a theorem which proves the linearity and invertibility of the associated mapping. Such invertibility permits us to convert the parameter space from moment-constrained filter to moment tensor as leaf variables in arbitrary high spatial dimensions. Training proceeds on the space of moment tensor instead of the space of filters to overcome the challenge. Proof of Theorem 1 is appended after the references.

Definition 2 (Moment tensor)**.** *Let d be a N-dimensional filter with size L^N and $d[i_1,\ldots,i_N]$ as its $i_1,\ldots,i_N$ entry. Let M be a N-dimensional moment tensor with size L^N. The mapping from d to M is defined as*
$$d \mapsto M = (m_{j_1,\ldots,j_N})_{L^N} \tag{13}$$
where
$$m_{j_1,\ldots,j_N} = M_1 \cdot M_2$$
with
$$M_1 = \prod_{i=j_1,\ldots,j_N} \frac{1}{(i-1)!}$$
$$M_2 = \sum_{(i_1,\ldots,i_N)\in\left(\mathbb{Z}_+^L\right)^N} \left(\prod_{k=1,\ldots,N} i_k^{j_k-1}\right) d[i_1,\ldots,i_N] \tag{14}$$
for $j_1,\ldots,j_N = 1,2,\ldots,L$.

Theorem 1. *The mapping defined in Definition 2, $d \mapsto M$, is linear and invertible.*

In each batch the gradient backpropagates to the moment tensor M according to
$$\frac{\partial L}{\partial M} = \frac{\partial L}{\partial W}\cdot\frac{\partial W}{\partial M} \tag{15}$$
where $\frac{\partial L}{\partial W}$ is expressed in equation (12) and $\frac{\partial W}{\partial M}$ is the gradient of the inverse mapping proved in Theorem 1. Only some entries $m_{j_1,\ldots,j_N}$ of the moment tensor M will be updated according to the order of the differential operators being approximated. Lastly, to compute the $\|\cdot\|_2$ for filters, we have leveraged the algorithm for efficient computation of spectral norm of filters in (Sedghi, Gupta, and Long 2018).

4 Other Regularization Techniques

In this section, we discuss some major regularization techniques and their limitations.

Weight decay. Weight decay (Krogh and Hertz 1992) adds the Frobenius norm (F-norm) regularization term $\|W\|_F^2$ to reduce the model's sensitivity. Note that
$$\max_i \sigma_i(W)^2 = \|W\|_2^2 \le \|W\|_F^2 = \sum_i \sigma_i(W)^2 \tag{16}$$
which shows F-norm is a more stringent constraint than spectral norm (2-norm). Regularizing the F-norm can mitigate the noise and error propagation, but it also shrinks the weight in all directions including those orthogonal to the first singular vector, with a tendency to inadvertently reduce W to a small rank matrix.

Weight normalization. Weight normalization (Salimans and Kingma 2016) normalizes the l_2 norm of each row vector in the weight matrices, i.e. $\sum\sigma_i(W)^2 = d$ where d is a pre-defined hyper-parameter. It differs from weight decay with just this hyper-parameter which in weight decay is taken to be 1. It suffers from same pitfall as weight decay.

Jacobian regularization. Jacobian regularization (Gu and Rigazio 2014) regularizes layerwise Jacobians, i.e. $\left\|\frac{\partial\tilde{u}_t}{\partial\tilde{u}_{t+1}}\right\|_F^2$, which is same as the F-norm of the weight matrices for PDE-Net as PDE-Net is linear. Jacobian regularization degenerates to weight decay.

Orthonormal regularization. Orthonormal regularization (Brock et al. 2016) adds the regularization term $\|W^{\intercal}W - I\|_F^2$. It targets to constrain all the singular values to one, thus risk destroying the information about the spectrum, unlike in DSN where we only constrain the largest singular values.

Batch normalization. Batch normalization (Ioffe and Szegedy 2015) normalizes the input of each layer to fixed means and variances in each batch, to mitigate the problem of internal covariate shift. Since PDE-Net is linear, batch normalization degenerates to data normalization (a coordinate transform of all the observed data), which can be done as a data pre-processing step but does not help regularize the filters directly.

Dropout. Dropout (Srivastava et al. 2014) randomly drops entries of weights and their connections from the neural network during training to prevent over-fitting. It is unfeasible for PDE-Net because both the filters or weight matrices carry physical meanings which cannot be dropped.

5 Experiment

In this section, we compare DSN against 5 other settings: (1) symmetric finite difference filters (**F**), (2) moment-constrained filters without regularization (**M**), (3) spectral normalization (**SN**), (4) Weight decay (**WD**) as weight normalization and jacobian regularization both degenerate to WD, and (5) Orthonormal regularization (**OR**) on a 1D non-linear PDE, namely, Korteweg-de Vries (KdV) equation, and a 2D linear PDE, namely, confection-diffusion equation respectively. To allow comparability, the implementation of SN, WD and OR are all adjusted with reference to theorem 1 to allow their trainings to proceed on the space of moment tensor.

Evaluation metric. We define two evaluation metrics, namely, relative error and maximum derivation. Relative error ϵ measures accuracy of non-zero coefficient functions and prediction of dynamical behavior.

$$\epsilon = \frac{\left\|\tilde{\vec{u}} - \vec{u}\right\|_2}{\|\vec{u} - \vec{u}_{\text{average}}\|_2} \tag{17}$$

where $\vec{u}$ is the true value, $\tilde{\vec{u}}$ is the estimated/ predicted value, $\vec{u}_{\text{average}}$ is the spatial average of $\vec{u}$. u here can be the non-zero coefficient functions or the PDE solution.

Maximum derivation (MD), on the other hand, evaluates the largest derivation of zero coefficient functions (i.e. the derivative terms do not exist in the PDE) by

$$\text{MD} = \max_A \{\|A\|_{\max}\} \tag{18}$$

where A's are the weight matrices approximating the zero coefficients.

Korteweg-de Vries (KdV) Equation

Our 1D non-linear PDE, KdV equation reads as

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} - a(x)\frac{\partial^3 u}{\partial x^3} \tag{19}$$

where $a(x) = 0.0025\,(1 + 0.0005\sin(x))$.

We take $\Omega = [0, 2\pi]$. The grid size of Ω is 100. The time step Δt is 0.01. Filter size is 5.

Suppose we know a priori that the PDE we have been looking for is of sum of order no more than 3 in each derivative term and that the 0-th order term appears in each derivative term at most once only. Equation (4) then takes the form

$$\begin{aligned} F =& I + \Delta t\big[A_1 W_1 + B_1 (W_0 \odot W_1) + A_{1,1} (W_1 \odot W_1) \\ & + B_{1,1} (W_0 \odot W_1 \odot W_1) + A_{1,1,1} (W_1 \odot W_1 \odot W_1) \\ & + B_{1,1,1} (W_0 \odot W_1 \odot W_1 \odot W_1) \\ & + A_{1,1,1,1} (W_1 \odot W_1 \odot W_1 \odot W_1) \\ & + B_{1,1,1,1} (W_0 \odot W_1 \odot W_1 \odot W_1 \odot W_1) \\ & + A_2 (W_2) + B_2 (W_0 \odot W_2) + A_{2,1} (W_2 \odot W_1) \\ & + B_{2,1} (W_0 \odot W_2 \odot W_1) + A_{2,1,1} (W_2 \odot W_1 \odot W_1) \\ & + B_{2,1,1} (W_0 \odot W_2 \odot W_1 \odot W_1) + A_{2,2} (W_2 \odot W_2) \\ & + B_{2,2} ((W_0 \odot W_2 \odot W_2) + A_3 (W_3) \\ & + B_3 (W_0 \odot W_3) + A_{3,1} (W_3 \odot W_1) \\ & + B_{3,1} (W_0 \odot W_3 \odot W_1) + A_4 (W_4) \\ & + B_4 ((W_0 \odot W_4) + A_0 (W_0)\big] \end{aligned} \tag{20}$$

where W_i is the matrix representation of the filter approximating differential operator of order i, and A and B are the weight matrices for the corresponding coefficient functions.

Convection-diffusion Equation

Our 2D linear PDE, convection-diffusion equation reads as

$$\frac{\partial u}{\partial t} = \sum_{0 \le i,j \le 2} a_{i,j}(x,y)\frac{\partial^{i+j} u}{\partial x^i \partial y^j} \tag{21}$$

where

$$\begin{aligned} a_{0,0}(x,y) &= 0 \\ a_{1,0}(x,y) &= \cos(x) + y\,(2\pi - y)\sin(y) + 0.6 \\ a_{0,1}(x,y) &= 2\,(\cos(x) + \sin(y)) + 0.8 \\ a_{2,0}(x,y) &= 0.3\cos(x) + 0.2\cos(x) \\ a_{0,2}(x,y) &= 0.2\sin(y) + 0.1\cos(y) \\ a_{1,1}(x,y) &= 0.1\sin(x) + 0.3\cos(y) \end{aligned} \tag{22}$$

We take $\Omega = [0, 2\pi] \times [0, 2\pi]$. The grid size of Ω is 100×100. Differentiation filter size is 5×5. The time step Δt is 0.01.

Suppose we know a priori that the PDE we have been looking for is linear and of order no more than 4. Equation (4) then takes the form

$$F = I + \Delta t\left[\sum_{0 \le i+j \le 4} A_{i,j} D_{i,j} \circledast\right] \tag{23}$$

where $D_{i,j}$ is the filter approximating differential operator of order i, j and $A_{i,j}$ is the weight matrix for the corresponding coefficient functions.

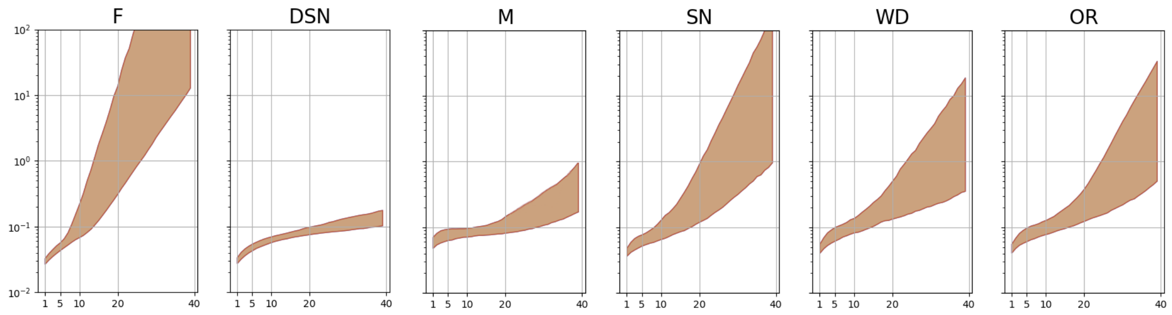

Figure 1: Relative errors of F, DSN, M, SN, WD, OR against time step Δt for scenario 2 with the KdV equation, with the banded curves showing the 25% & 75% percentile of the relative errors among 30 repeated evaluations

	F	DSN	M	SN	WD	OR
B_1	0.628	0.316	0.390	0.566	0.523	0.540
A_3	0.720	0.307	0.474	0.625	0.508	0.498

Table 1: Relative errors of non-zero coefficients for the KdV equation (Scenario 1)

	F	DSN	M	SN	WD	OR
B_1	0.823	0.352	0.499	0.693	0.642	0.649
A_3	0.783	0.389	0.542	0.710	0.631	0.602

Table 2: Relative errors of non-zero coefficients for the KdV equation (Scenario 2)

Train-set. We generate the train-set as follows.

- The initial condition for the KdV equation is $u_0 = \lambda\cos(x)$ with $\lambda \sim N(0.2, 0.05)$. The initial condition for the convection-diffusion equation is $u_0 = \lambda\sin(x) + \gamma\cos(y)$ with $\lambda, \gamma \sim N(0.1, 0.05)$. The batch size is 12. Adam (Kingma and Ba 2014) is used as the optimizer.
- For both PDEs, we set two separate scenarios with different noise magnitudes. Noise is added to train-set by

$$\hat{\vec{u}}_t = \vec{u}_t + [\max(\vec{u}_t) - \min(\vec{u}_t)] \cdot \mu \qquad (24)$$

  where max and min is the spatial maximum and minimum respectively. We set $\mu \sim N(0, 0.01)$ and $\mu \sim N(0, 0.05)$ in scenario 1 and 2 respectively.
- For both PDEs, to mimic real-world scenarios with scarce data setting, the train-set only consists of data in time $\{0, \Delta t, 5\Delta t\}$. Training consists of two steps. First step is to compare the prediction at time Δt from time 0 with the observed value. Second step is with the prediction at time $5\Delta t$.

Test-set. To evaluate the predictive powers and robustness of the models, we generate the test-set as follows.

- The time is from 0 to $40\Delta t$, $\{0, \Delta t, 2\Delta t, \ldots, 40\Delta t\}$. We will observe the prediction at time Δt to $40\Delta t$ based on input at time 0. Noise is added to the test-set in the same way as train-set scenario 2.
- In the test-set, we aim to test the trained models given unseen initial conditions. The initial condition for the KdV equation is $u_0 = \lambda\sin(x)$ with $\lambda \sim N(0.2, 0.01)$. The initial condition for convection-diffusion equation is $u_0 = \sum_{0\le i+j\le 4} \lambda_{i,j}\sin(ix+jy) + \gamma_{i,j}\cos(ix+jy)$ with $\lambda_{i,j}, \gamma_{i,j} \sim N(0.1, 0.1)$

	F	DSN	M	SN	WD	OR
MD1	2.213	0.250	0.262	1.626	1.493	1.670
MD2	3.978	0.374	0.358	2.591	1.977	2.113

Table 3: MD of zero coefficients for the KdV equation with MD1 and MD2 for scenarios 1 and 2 resp.

Results and Discussion

Here we present our experimental results and discuss the effectiveness of different regularization techniques in different settings.

KdV equation. Tables 1, 2 and 3 show that given increased noise magnitude in the train-set, both the relative errors and the maximum derivation for coefficients surge in all the six settings. This suggests the presence and increase of noise hinders the training of filters. A careful examination of tables 1 and 2 also reveals that given the same amount of noise increase, the relative errors of coefficients for DSN grow by the smallest amount, suggesting that DSN is more robust towards noise.

Tables 1 and 2 further show that DSN estimates the non-zero coefficients most accurately among all six settings, followed by M. We see that SN, WD or OR have not improved the training of PDE-Net. One possible reason is that we take the hyper-parameter for the regularization terms of all these three methods as 1. A more careful tuning of hyper-parameter may be needed, but this shows the deficiency of these three methods compared to DSN which doesn't require any hyper-parameter tuning. Also note that WD and OR perform slightly better than SN. A possible explanation is that

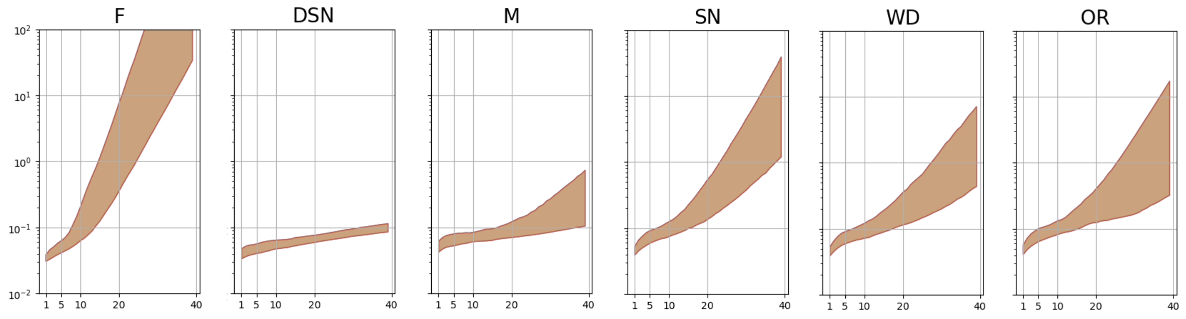

Figure 2: Relative errors of F, DSN, M, SN, WD, OR against time step Δt for scenario 2 with the convection-diffusion equation, with the banded curves showing the 25% & 75% percentile of the relative errors among 30 repeated evaluations

	F	DSN	M	SN	WD	OR
$A_{1,0}$	0.523	0.246	0.333	0.503	0.492	0.481
$A_{0,1}$	0.643	0.329	0.317	0.526	0.413	0.460
$A_{2,0}$	0.738	0.311	0.412	0.579	0.543	0.521
$A_{0,2}$	0.712	0.290	0.390	0.502	0.498	0.452
$A_{1,1}$	0.620	0.279	0.351	0.499	0.442	0.470

Table 4: Relative errors of non-zero coefficients for the convection-diffusion equation (Scenario 1)

	F	DSN	M	SN	WD	OR
$A_{1,0}$	0.742	0.366	0.486	0.648	0.599	0.584
$A_{0,1}$	0.781	0.415	0.432	0.623	0.561	0.602
$A_{2,0}$	0.803	0.353	0.590	0.688	0.703	0.610
$A_{0,2}$	0.890	0.395	0.505	0.641	0.537	0.591
$A_{1,1}$	0.784	0.312	0.421	0.712	0.606	0.572

Table 5: Relative errors of non-zero coefficients for the convection-diffusion equation (Scenario 2)

SN tremendously shrinks the filter norm size to 1 while WD and OR add their regularization terms to the loss function without directly fixing the spectral norm of the filter to 1.

Figure 1 indicates that when faced with unseen data, DSN remain to be stable and its relative error of prediction stay relatively flat till time 40 while that of M start blowing up since time 20. We also see that the relative errors of SN, WD, and OR explode at a quite early time compared to DSN and M. This result matches with our observation on the relative errors and maximum derivations of the coefficients.

Convection-diffusion equation Agreeing with the observations for the KdV equation, tables 4, 5 and 6 show that DSN estimate coefficients most accurately and are most robust to noise among the tested methods.

Figure 2 indicates that the relative errors of prediction for DSN stay very flat and stable from the beginning to time 40, showing a strong robustness towards disturbance like noises and errors. The relative errors of SN, WD and OR are observed to explode at around time 10 and that of M climb slowly.

	F	DSN	M	SN	WD	RO
MD1	1.493	0.312	0.562	1.044	1.272	1.673
MD2	4.773	0.503	0.714	3.439	2.001	2.395

Table 6: MD of zero coefficients for the convection-diffusion equation with MD1 and MD2 for scenarios 1 and 2 resp.

In summary, we observe that DSN is an effective and powerful means of regularization for moment-constrainted filters in PDE-Net. Compared to other common regularization techniques, DSN is more robust towards noise and errors, offers a more accurate estimation of coefficient functions and a more stable prediction of long term dynamics. Other common regularization techniques are either not applicable (dropout and batch normalization) or are too stringent as a constraint to have a tendency to distort the spectrum of the weight matrices (SN, WD, OR).

6 Conclusion

In this paper, we have studied the background of PDE discovery problem and the limitations of existing data-driven approaches. Motivated by one of the latest advance, PDE-Net, which lacks a robust regularization tool, we propose a novel regularization technique, Differential Spectral Normalization (DSN), tailored for the moment-constrained filters widely used in PDE-Net. We have conducted a detailed investigation on the effectiveness of DSN against other common regularization methods, including batch normalization, dropout, weight decay, weight normalization, jacobian regularization, orthonormal regularization, and examined them with data from two different PDEs (1D KdV equation and 2D convection diffusion equation) under noisy and unseen conditions. Substantial empirical evidence suggests that DSN outperforms all of them in terms of (1) predictive power over a long time horizon, (2) accurate estimation of coefficient functions, and (3) the robustness to noises and errors in both train-set and test-set. We expect moment-constrained filters with DSN to have a great potential to be widely applied in convolution neural network for diverse disciplines.

Acknowledgments

We thank anonymous reviewers for their helpful comments. This work is supported in part by the "TCL Innovative Research Fund for Science 80th Anniversary" for the HKU-TCL Ph.D. fellowship of Chi Chiu So.

Proof of Theorem 1

Here we supply the proof of Theorem 1.

Proof. If we express the moment tensor M and filter d as two vectors $\vec{M}$ and $\vec{d}$ with certain indexing, it suffices to show that the mapping defined in Definition 2 can be expressed as an invertible linear transformation. In other words, if we can find a linear transformation $\mathbf{T}$ such that

$$\vec{M} = \mathbf{T}\vec{d}$$

and show that $\mathbf{T}$ is invertible, then we are done.

We will proceed with mathematical induction on N. For $N = 1$, we have

$$m_j = \frac{1}{(j-1)!} \sum_{i \in \mathcal{Z}_L} \left(i^{j-1} d[i] \right)$$

for $j = 1, 2, \dots, L$. Expressing the transformation $\mathbf{T}$ in matrix form, we get

$$\vec{M} = \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_L \end{bmatrix} = \mathbf{T} \begin{bmatrix} d[-\frac{(L-1)}{2}] \\ d[-\frac{(L-1)}{2}+1] \\ \vdots \\ d[\frac{(L-1)}{2}] \end{bmatrix} = \mathbf{T}\vec{d}$$

where

$$\mathbf{T} = \mathbf{T_1 T_2}$$

,

$$\mathbf{T_1} = \begin{bmatrix} \frac{1}{(1-1)!} & & & \mathbf{0} \\ & \frac{1}{(2-1)!} & & \\ & & \ddots & \\ \mathbf{0} & & & \frac{1}{(L-1)!} \end{bmatrix}$$

and

$$\mathbf{T_2} = \begin{bmatrix} \left(\frac{1-L}{2}\right)^{1-1} & \left(1-\frac{(L-1)}{2}\right)^{1-1} & \cdots & \left(\frac{(L-1)}{2}\right)^{1-1} \\ \left(\frac{1-L}{2}\right)^{2-1} & \left(1-\frac{(L-1)}{2}\right)^{2-1} & \cdots & \left(\frac{(L-1)}{2}\right)^{2-1} \\ \vdots & \vdots & \ddots & \vdots \\ \left(\frac{1-L}{2}\right)^{L-1} & \left(1-\frac{(L-1)}{2}\right)^{L-1} & \cdots & \left(\frac{(L-1)}{2}\right)^{L-1} \end{bmatrix}.$$

We observe that $\mathbf{T_1}$ is a diagonal matrix, and is obviously invertible. $\mathbf{T_2}$ is in fact a square Vandermonde matrix, whose determinant is proven to be non-zero, thus is also invertible. As product of invertible matrices are invertible, $\mathbf{T} = \mathbf{T_1 T_2}$ is invertible. We have finished for the case $N = 1$.

Now assume Theorem 1 holds for the case $N = k$, we will prove for $N = k+1$. As the theorem holds for $N = k$, if d_k is a k-dimensional filter with size L^k and M_k is the moment tensor mapped by d_k according to the mapping defined in Definition 2, we have an invertible matrix $\mathbf{T_k}$ such that

$$\vec{M}_k = \mathbf{T_k}\vec{d}_k$$

where $\vec{M}_k$ and $\vec{d}_k$ are the vector expressions for M_k and d_k respectively.

Now, looking at M_{k+1} and d_{k+1}, we construct their vector forms $\vec{M}_{k+1}$ and $\vec{d}_{k+1}$ by "stacking" L copies of $\vec{M}_k$ and $\vec{d}_k$ together, such that the m-th copy contains entries of M_{k+1} and d_{k+1} indexed with m in the $k+1$ dimension. We write as

$$\vec{M}_{k+1} = \begin{bmatrix} \vec{M}_{k,j_{k+1}=1} \\ \vec{M}_{k,j_{k+1}=2} \\ \vdots \\ \vec{M}_{k,j_{k+1}=L} \end{bmatrix}_{L^{k+1}}$$

and

$$\vec{d}_{k+1} = \begin{bmatrix} \vec{d}_k[i_{k+1} = -\frac{(L-1)}{2}] \\ \vec{d}_k[i_{k+1} = -\frac{(L-1)}{2}+1] \\ \vdots \\ \vec{d}_k[i_{k+1} = \frac{(L-1)}{2}] \end{bmatrix}_{L^{k+1}}$$

where $\vec{M}_{k,j_{k+1}=x}$ refer to the copy of $\vec{M}_k$ indexed with m in the $k+1$ dimension, and $\vec{d}_k[i_{k+1} = x]$ refer to the copy of $vecd_k$ indexed with m in the $k+1$ dimension.

Now, we construct $\mathbf{T_{k+1}}$ by a product of three square matrices all of size L^{k+1} as follows:

$$\mathbf{T_{k+1}} = \mathbf{T_1 T_2 T_3}$$

where

$$\mathbf{T_1} = \begin{bmatrix} \mathbf{T_k} & & & \mathbf{0} \\ & \mathbf{T_k} & & \\ & & \ddots & \\ \mathbf{0} & & & \mathbf{T_k} \end{bmatrix}$$

,

$$\mathbf{T_2} = \begin{bmatrix} \mathbf{\frac{1}{(1-1)!}} & & & \mathbf{0} \\ & \mathbf{\frac{1}{(2-1)!}} & & \\ & & \ddots & \\ \mathbf{0} & & & \mathbf{\frac{1}{(L-1)!}} \end{bmatrix}$$

and

$$\mathbf{T_3} = \begin{bmatrix} \left(\mathbf{\frac{1-L}{2}}\right)^{\mathbf{1-1}} & \left(\mathbf{1}-\mathbf{\frac{(L-1)}{2}}\right)^{\mathbf{1-1}} & \cdots & \left(\mathbf{\frac{(L-1)}{2}}\right)^{\mathbf{1-1}} \\ \left(\mathbf{\frac{1-L}{2}}\right)^{\mathbf{2-1}} & \left(\mathbf{1}-\mathbf{\frac{(L-1)}{2}}\right)^{\mathbf{2-1}} & \cdots & \left(\mathbf{\frac{(L-1)}{2}}\right)^{\mathbf{2-1}} \\ \vdots & \vdots & \ddots & \vdots \\ \left(\mathbf{\frac{1-L}{2}}\right)^{\mathbf{L-1}} & \left(\mathbf{1}-\mathbf{\frac{(L-1)}{2}}\right)^{\mathbf{L-1}} & \cdots & \left(\mathbf{\frac{(L-1)}{2}}\right)^{\mathbf{L-1}} \end{bmatrix}$$

where $\mathbf{T_1}$ is a diagonal block matrix with each diagonal block being $\mathbf{T_k}$, $\mathbf{\frac{1}{(i-1)!}}$ in T_2 refers to a diagonal matrix of size L^k with diagonal entries taking $\frac{1}{(i-1)!}$ and $\mathbf{i^{j-1}}$ in T_3 refers to a diagonal matrix of size L^k with diagonal entries taking i^{j-1}.

In details, $\mathbf{T_1}$ is composed of L's $\mathbf{T_k}$ on its diagonal, $\mathbf{T_2}$ is composed of L's $\frac{\mathbf{1}}{\mathbf{(i-1)!}}$ on its diagonal and $\mathbf{T_3}$ is composed of $L \times L$'s diagonal matrices of size L^k. Direct checking entry by entry shows that $\vec{M}_{k+1} = \mathbf{T_{k+1}}\vec{d}_{k+1} = \mathbf{T_1T_2T_3}\vec{d}_{k+1}$. Intuitively speaking, $\mathbf{T_1}$ is the transformation in k dimensions and $\mathbf{T_2} \times \mathbf{T_3}$ is the transformation in the newly added $k+1$-th dimension. It is obvious that $\mathbf{T_1}$ and $\mathbf{T_2}$ are both diagonal and invertible. For $\mathbf{T_3}$, though it may not be obvious in a first glance, its inverse can be constructed by a combination of the inverses of each submatrices in $\mathbf{T_3}$, which are diagonal and invertible. As $\mathbf{T_1}$, $\mathbf{T_2}$ and $\mathbf{T_3}$ are all invertible, $\mathbf{T_{k+1}}$ is invertible and our proof is finished. □

References

Baydin, A. G.; Pearlmutter, B. A.; Radul, A. A.; and Siskind, J. M. 2017. Automatic differentiation in machine learning: a survey. *The Journal of Machine Learning Research* 18(1): 5595–5637.

Berg, J.; and Nyström, K. 2019. Data-driven discovery of PDEs in complex datasets. *Journal of Computational Physics* 384: 239–252.

Brock, A.; Lim, T.; Ritchie, J. M.; and Weston, N. 2016. Neural photo editing with introspective adversarial networks. *arXiv preprint arXiv:1609.07093* .

Brunton, S. L.; Proctor, J. L.; and Kutz, J. N. 2016. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences* 113(15): 3932–3937.

Burden, R.; and Faires, J. 2004. *Numerical analysis*. Cengage Learning.

Dong, B.; Jiang, Q.; and Shen, Z. 2017. Image restoration: Wavelet frame shrinkage, nonlinear evolution pdes, and beyond. *Multiscale Modeling & Simulation* 15(1): 606–660.

Gu, S.; and Rigazio, L. 2014. Towards deep neural network architectures robust to adversarial examples. *arXiv preprint arXiv:1412.5068* .

Haber, E.; and Ruthotto, L. 2017. Stable architectures for deep neural networks. *Inverse Problems* 34(1): 014004.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016a. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016b. Identity mappings in deep residual networks. In *European conference on computer vision*, 630–645. Springer.

Ioffe, S.; and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167* .

Jia, K.; Tao, D.; Gao, S.; and Xu, X. 2017. Improving training of deep neural networks via singular value bounding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4344–4352.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Krogh, A.; and Hertz, J. A. 1992. A simple weight decay can improve generalization. In *Advances in neural information processing systems*, 950–957.

Long, Z.; Lu, Y.; and Dong, B. 2019. PDE-Net 2.0: Learning PDEs from data with a numeric-symbolic hybrid deep network. *Journal of Computational Physics* 399: 108925.

Long, Z.; Lu, Y.; Ma, X.; and Dong, B. 2017. PDE-net: Learning PDEs from data. *arXiv preprint arXiv:1710.09668* .

Miyato, T.; Kataoka, T.; Koyama, M.; and Yoshida, Y. 2018. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957* .

Qin, T.; Wu, K.; and Xiu, D. 2019. Data driven governing equations approximation using deep neural networks. *Journal of Computational Physics* 395: 620–635.

Raissi, M. 2018. Deep hidden physics models: Deep learning of nonlinear partial differential equations. *The Journal of Machine Learning Research* 19(1): 932–955.

Raissi, M.; Perdikaris, P.; and Karniadakis, G. 2017a. Physics informed deep learning (part II): Data-driven discovery of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10566* .

Raissi, M.; Perdikaris, P.; and Karniadakis, G. E. 2017b. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations. *arXiv preprint arXiv:1711.10561* .

Rudy, S. H.; Brunton, S. L.; Proctor, J. L.; and Kutz, J. N. 2017. Data-driven discovery of partial differential equations. *Science Advances* 3(4): e1602614.

Saito, M.; Matsumoto, E.; and Saito, S. 2017. Temporal generative adversarial nets with singular value clipping. In *Proceedings of the IEEE international conference on computer vision*, 2830–2839.

Salimans, T.; and Kingma, D. P. 2016. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in neural information processing systems*, 901–909.

Schaeffer, H. 2017. Learning partial differential equations via data discovery and sparse optimization. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 473(2197): 20160446.

Schaeffer, H.; Caflisch, R.; Hauck, C. D.; and Osher, S. 2013. Sparse dynamics for partial differential equations. *Proceedings of the National Academy of Sciences* 110(17): 6634–6639.

Sedghi, H.; Gupta, V.; and Long, P. M. 2018. The singular values of convolutional layers. *arXiv preprint arXiv:1805.10408* .

Singla, S.; and Feizi, S. 2019. Bounding Singular Values of Convolution Layers. *arXiv preprint arXiv:1911.10258* .

Sonoda, S.; and Murata, N. 2017. Double continuum limit of deep neural networks. In *ICML Workshop Principled Approaches to Deep Learning*, volume 1740.

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15(1): 1929–1958.

Wu, K.; and Xiu, D. 2020. Data-driven deep learning of partial differential equations in modal space. *Journal of Computational Physics* 408: 109307.

Xu, H.; Chang, H.; and Zhang, D. 2019. Dl-pde: Deep-learning based data-driven discovery of partial differential equations from discrete and noisy data. *arXiv preprint arXiv:1908.04463* .

Yoshida, Y.; and Miyato, T. 2017. Spectral norm regularization for improving the generalizability of deep learning. *arXiv preprint arXiv:1705.10941* .