SANDIA REPORT

SAND2018-11042
Unlimited Release
Printed 9,2018

Neural Networks as Surrogates of Nonlinear High-Dimensional Parameter-to-Prediction Maps

John D. Jakeman, Mauro Perego, William M. Severa

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Approved for public release; further dissemination unlimited.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by National Technology and Engineering Solutions of Sandia, LLC.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government, nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof, or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.fedworld.gov
Online ordering: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2018-11042
LDRD Project Number: 212725
Unlimited Release
Printed 9,2018

Neural Networks as Surrogates of Nonlinear High-Dimensional Parameter-to-Prediction Maps

John D. Jakeman and Mauro Perego and William M. Severa

Abstract

We present a preliminary investigation of the use of Multi-Layer Perceptrons (MLP) and Recurrent Neural Networks (RNNs) as surrogates of parameter-to-prediction maps of computational expensive dynamical models. In particular, we target the approximation of Quantities of Interest (QoIs) derived from the solution of a Partial Differential Equations (PDEs) at different time instants. In order to limit the scope of our study while targeting a relevant application, we focus on the problem of computing variations in the ice sheets mass (our QoI), which is a proxy for global mean sea-level changes. We present a number of neural network formulations and compare their performance with that of Polynomial Chaos Expansions (PCE) constructed on the same data.

Contents

List of Figures

Introduction

Uncertainty quantification (UQ) of high-fidelity models typically requires large numbers of simulations. Building an approximation or surrogate of the model is an effective and popular approach to reduce the computational burden of UQ. Numerous techniques have been developed for approximating models parameterized by random variables. Some of the most widely adopted methods for approximating models parameterized by random variables are those based on generalized polynomial chaos expansions [15, 38], sparse grid approximation [25, 26, 37], Gaussian process models [29, 4, 33] and low-rank tensor decompositions [13, 24, 16, 27].

Neural networks (NN) have also been successfully used to approximate nonlinear functions. Indeed the universal approximation property states that in principle a sufficiently large three layer neural network can in principle model any continuous function on a compact domain [20]. In this paper we investigate the numerical performance of NN used extensively in the machine learning community and and the popular PCE used by the UQ community for building approximations of quantities of interested extracted from dynamic ice sheet models.

The mass loss of Greenland and Antarctic ice sheets significantly contributes the rise of the sea level (Figure 1). Global mean sea level is rising at a rate of 3.2 mm/yr, but the rate is increasing [9] and recent studies [36] predict that the sea level could rise from 0.3 to 2.7 meters by the end of the century. These studies highlight the need of more accurate and probabilistic predictions of sea-level rise, which is also a goal of the DOE Earth System Modeling program [1, 2]. There are several uncertain fields in an ice sheet model, including the basal friction field (i.e. the friction between the ice and the bedrock), the bed topography and the basal geothermal heat flux. In this paper we limit to consider the uncertainty on the basal friction field. In the following we will consider a simplified ice sheet model based upon the Shallow Ice Approximation [3]. The model approximates ice sheet dynamics well in regions where the ice is grounded and where the bed is mostly frozen. We will build surrogates of the parameter-to-prediction map that maps the basal friction parameters into our QoI, the total change of mass of the ice sheet.

The accurate prediction of the evolution of the mass of ice sheets, requires the solution of very expensive physics-based computational models that feature $O(10^8)$ unknowns and a large number of uncertain parameters. The high dimensionality of the parameter space and the large computational cost of the forward model make UQ extremely challenging for the aforementioned UQ methods.

In this report we will explore the benefits of using RNN surrogates. Specifically we will consider, accuracy and training and evaluation costs. The structure of RNN differs substantially from more established surrogate models like the PCE. RNNs can use temporal and state information from the training data to train the network. This can increase the amount of training data significantly. This relationship has already been used used to effectively model deterministic dynamical systems [28]

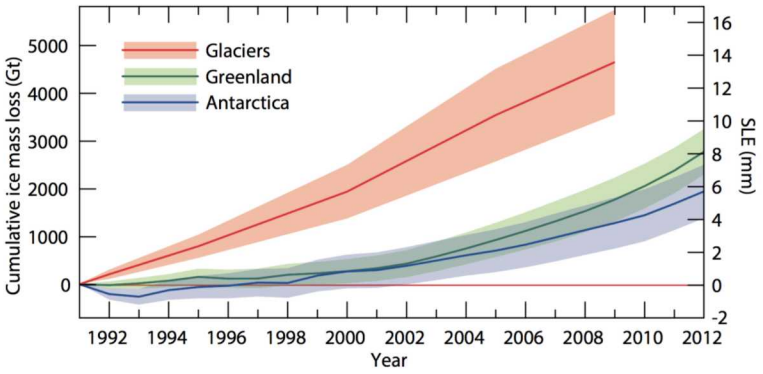

Figure 1. Cumulative ice sheet mass loss (left axis) and resulting sea-level rise (right axis) over the last two decades (after Fig. TS.3 from [34]).

Detailed description of the methods

In this section we briefly describe the methods and introduce the notation we will use in the rest of the paper. We first define a general stochastic problem governed by PDEs, then we introduce the PCE approach for generating a surrogate for the QoI of our computational model. Finally we describe neural networks and how we plan to use them to generate surrogates. The NN surrogates will be compared to the PCE surrogates, considered as a baseline for comparison.

Stochastic problem setup

Let $D \subset \mathbb{R}^\ell$, $\ell = 1, 2, 3$, be a physical domain and let $T > 0$ be a real number. We consider the following general stochastic partial differential equation

$$\begin{cases} u_t(\boldsymbol{x}, \boldsymbol{z}) = \mathcal{L}(u), & D \times (0, T] \times \Gamma, \\ \mathcal{B}(u) = 0, & \partial D \times [0, T] \times \Gamma, \\ u = u_0, & D \times \{t = 0\} \times \Gamma, \end{cases} \tag{1}$$

where $\boldsymbol{x} = (x_1, \ldots, x_\ell, t)$ represents both the physical and temporal coordinates, $\mathcal{L}$ is a (non-linear) differential operator, $\mathcal{B}$ is the boundary condition operator, u_0 is the initial condition, and $\boldsymbol{z} = (z_1, \ldots, z_{n_z}) \in \Gamma \subset \mathbb{R}^{n_z}, n_z \geq 1$, are a set of random variables characterizing the random inputs to the governing equation. The solution is therefore a stochastic quantity,

$$u(\boldsymbol{x}, \boldsymbol{z}) : \bar{D} \times [0, T] \times \Gamma \to \mathbb{R}^{n_u}. \tag{2}$$

Often one is not interested in approximating the entire solution but rather some Quantities of Interest (QoIs) which are functionals of the solution. That is we want to focus on

the approximation of

$$Q(u(\boldsymbol{x}, \boldsymbol{z})) : \mathbb{R}^{n_u} \to \mathbb{R}^{n_q} \tag{3}$$

and

$$f(\boldsymbol{z}) : \Gamma \to \mathbb{R}^{n_q}$$

is a function assigning the value of the quantity of interest $Q(u)$ to each realization of $\boldsymbol{z}$. Without loss of generality, hereafter we assume Q and f are a scalar valued functions with $n_q = 1$.

Polynomial chaos expansions

Polynomial chaos expansions (PCE) [15, 22, 35, 38] represent the model output $f(\boldsymbol{z})$ as an expansion in orthonormal polynomials

$$f(\boldsymbol{z}) \approx f_N(\boldsymbol{z}) = \sum_{\lambda \in \Lambda} \alpha_\lambda \phi_\lambda(\boldsymbol{z}), \qquad |\Lambda| = N. \tag{4}$$

The basis functions ϕ_λ are constructed to be orthonormal with respect to the density ω, that is

$$(\phi_{\lambda^{(i)}}(\boldsymbol{z}), \phi_{\lambda^{(j)}}(\boldsymbol{z}))_{L^2_\omega(\Omega)} \coloneqq \int_\Gamma \phi_{\lambda^{(i)}}(\boldsymbol{z}) \phi_{\lambda^{(j)}}(\boldsymbol{z}) \, \mathrm{d}\omega(\boldsymbol{z}) = \delta_{i,j},$$

where Γ is the support of the density ω, $\delta_{i,j}$ is the Kronecker delta function and $\lambda^{(i)}, \lambda^{(j)}$ are two different multivariate indices where $\lambda = (\lambda_1 \ldots, \lambda_{n_z}) \in \mathbb{N}_0^{n_z}$. Under mild conditions on the distribution $\omega(\boldsymbol{z})$, any function $f(\boldsymbol{z})$ with finite variance, i.e. $f \in L^2_\omega(\Omega)$ can be represented by a PCE that converges in L^2_ω to the true function asymptotically [14]. Polynomial chaos expansions are typically constructed when the components of $\boldsymbol{z}$ are independent. Under the assumption of independence, we have

$$\Gamma = \times_{i=1}^{n_z} \Gamma_i, \qquad \Gamma_i \subset \mathbb{R}, \qquad \omega(\boldsymbol{z}) = \prod_{i=1}^{n_z} \omega_i(z_i),$$

where ω_i are the marginal densities of the variables $\boldsymbol{z}_i$, which completely characterizes the distribution of $\boldsymbol{z}$. This allows us to express the basis functions ϕ as tensor products of univariate orthonormal polynomials. That is

$$\phi_\lambda(\boldsymbol{z}) = \prod_{i=1}^{n_z} \phi^i_{\lambda_i}(z_i), \tag{5}$$

where the univariate basis functions ϕ^i_j are defined uniquely (up to a sign) for each $i = 1, \ldots, n_z$, as

$$\int_{\Omega_i} \phi^i_j(z_i) \phi^i_k(z_i) \omega_i(z_i) \, \mathrm{d}z_i = \delta_{j,k}, \qquad j, k \geq 0, \qquad \deg \phi^i_j = j.$$

With this notation, the total degree of the polynomial ϕ_λ is $|\lambda| \coloneqq \sum_{j=1}^{n_z} \lambda_j$. In practice the PCE (4) must be truncated to some finite number of terms, say N, defined by a multi-index set $\Lambda \subset \mathbb{N}_0^{n_z}$. Frequently the PCE is truncated to retain only the multivariate polynomials whose associated multi-indices have norm at most p, i.e.,

$$\Lambda = \Lambda_{p,q}^{n_z} = \{\phi_\lambda : \|\lambda\|_q \leq p\}, \qquad \|\lambda\|_q \coloneqq \left(\sum_{i=1}^{n_z} \lambda_i^q\right)^{1/q}. \tag{6}$$

Taking $q = 1$ results in a total-degree space having dimension card $\Lambda_{p,1}^{n_z} \equiv N = \binom{n_z+p}{n_z}$.

In this paper we will compute the coefficients of PCE using compressed sensing. Recently compressed sensing techniques [6, 7, 10, 11] have been shown to be an effective means of approximating PCE coefficients from small number of function samples [5, 12, 21, 39]. These methods are most effective when the number of non-zero expansion terms in the PCE approximation of the model output is small (i.e. expansion coefficient sparsity), or the magnitude of the PCE coefficients decays rapidly (i.e. compressibility).

Given a set of M realizations $\bar{\boldsymbol{z}} = \{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(M)}\}$, with corresponding model outputs $\boldsymbol{f} = (f(\boldsymbol{z}^{(1)}), \ldots, f(\boldsymbol{z}^{(M)}))^T$, we would like to find a solution that satisfies $\Phi\boldsymbol{\alpha} \approx \boldsymbol{f}$, where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_N)^T$ denotes the vector of PCE coefficients and $\Phi \in \mathbb{R}^{M\times N}$ denotes the Vandermonde matrix with entries $\Phi_{kj} = \phi_k(\boldsymbol{z}^{(i)}), \quad i = 1, \ldots, M,\ k = 1, \ldots, N$. ℓ^1-minimization attempts to find the dominant PCE coefficients by solving the optimization problem

$$\boldsymbol{\alpha}^\star = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \ \|\boldsymbol{\alpha}\|_1 \quad \text{such that} \quad \|\Phi\boldsymbol{\alpha} - \boldsymbol{f}\|_2 \leq \epsilon, \tag{7}$$

where ϵ is a noise/tolerance that allows the data to slightly deviate from the PCE. This ℓ^1-minimization problem is often referred to as Basis Pursuit Denoising (BPDN) [8].

Neural Network Methods

In recent years, advances in computational power and available data has led to a boom in deep learning neural network methods [23]. These methods are fundamentally similar to early, underwhelming neural networks but iterative improvements have led to systems capable of surpassing human performance at certain tasks such as image classification [18]. The traditional formulation of a neural network is a directed *acyclic* graph of neuron layers.

In this report we investigate the utility of using recurrent neural network (RNN) which has a structure well suited to the sequential formulation of a QoI evolving over time. A RNN differs from a standard neural network (or feed-forward neural network) in that the layer graph may have *cyclic* sub-graphs. This means that a network computes on both the propagated values of the input as well as the recurrent state. The particular type of RNN we decided to use is called Long Short Term Memory (LSTM) [19]. LSTM nodes are distinguished by having auxiliary activation functions responsible for maintaining the state. In the original formulation, there is both a 'forget' gate and a 'remember' gate, though later formulations remove one or merge these concepts.

We are interested in using LSTMs to predict the QoI $f(\boldsymbol{z})$ and/or PDE states $u(\boldsymbol{x}, \boldsymbol{z})$ at timestep t using values of those quantities at previous time steps $t-1, \ldots, t-k$, for some window size k, and the value of the random variables $\boldsymbol{z}$. It is unlikely for many models that the entire time history is needed to predict the future state of the model. Consequently we can use LSTM to train only on data in a small time window. The usage of temporal data in this fashion increases the amount of data available for training a network. This data can be split up into overlapping or non-overlapping time windows. Each window now represents training data which can be used to train the network. If the model data from M samples is divided into N_w windows then LSTMs are trained using MN_w pieces of information which are considered independent. To reduce correlation between neighboring time-windows we randomly order the windowed data during training.

We designed two types of LSTM networks that predict (1) the QoI at the next time step or (2) the QoI over a vector of values, representing time steps, i.e. 'sequence-to-sequence' learning. With these two network types, we structured the input data in two separate ways. For type-1 networks (Figure 2), we split the data into vectors of predetermined window sizes. This effectively increases the number of sample point, but each sample now carries less information. For type-2 networks (Figure 3), we used the entire time history of the QoI as a single sample.

In our study we also consider Multi-Layer Perceptron (MLP), which is the most simplistic form of an artificial neural network. In an MLP, neurons are arranged in layers and, in its basic formulation, the neurons in each layer compute $\sigma(W\vec{x})$ where $\vec{x}$ is the input vector (or the activation in the previous layer), W is a weight matrix determined usually determined by stochastic gradient descent, and σ is a monotonically increasing, differentiable activation function. For our experiments, we implement two key differences from a standard MLP:

1. We utilize *dropout* between the layers. Dropout is a often-used method where, during training, a random percentage of the activations are set to 0. This method allows neurons to learn distributed representations rather than track specific upstream neurons. This method has been used regularly to prevent over-fitting to the training data.

2. After the multiplication $W\vec{x}$ we apply *Batch Normalization* before the result is passed to the activation function. Batch Normalization attempts to keep the mean and standard deviation of the intermediate states near 0 and 1 respectively. Batch Normalization is computationally efficient and improves general performance, particularly with deep networks.

In all our MLP experiments, we use a single network with multiple outputs (one for each prediction time). This allows the remainder of the network to have shared representations, thus improving performance across a small data set.

One of the key challenges in designing and training neural networks is selection of *hyperparameters*. The term hyperparameters is commonly used in the deep learning community and refers to high-level decisions necessary to training a neural network. Hyperparameters

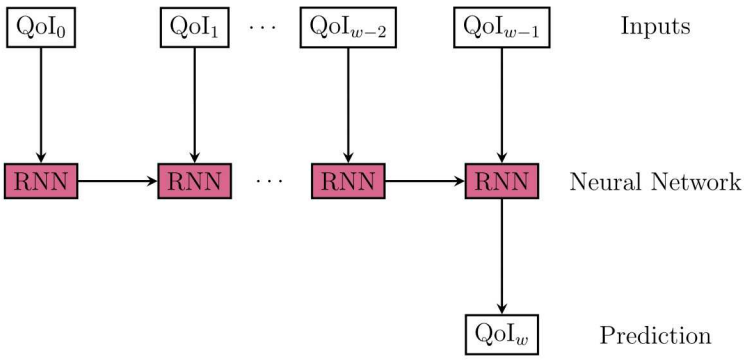

Figure 2. Type 1 Network

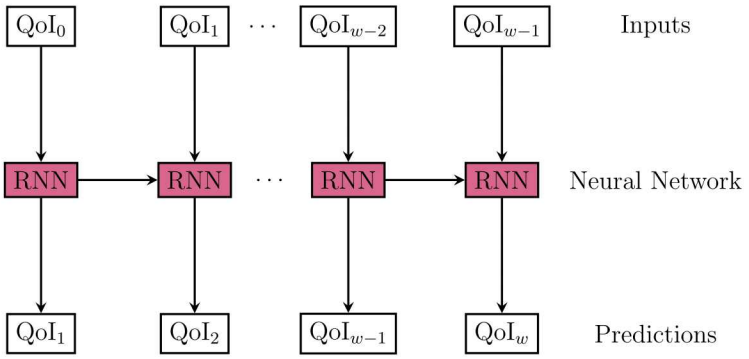

Figure 3. Type 2 Network

affect the network but are not part of the direct optimization problem (e.g. the number of layers, the width of the layers, rate of weight updates (called the *learning rate*)). These hyperparameters can have a huge effect on the performance of a network as well as its training requirements, both in terms of required training data and training wall time. Given the short duration of this project, we focused on the hyperparameters most likely to impact the network performance and started with well-accepted default values. In lieu of a full search space, we used a simple grid search through reasonable values.

Results

Shallow ice approximation

In the following we use the shallow ice approximation (SIA) in a one-dimensional spatial domain. The ice is confined between the ice bed $b(x) \geq 0$ and the ice surface $h(x, t, \boldsymbol{z}) \geq b(x)$, see Figure 4. The model specifies the evolution of the ice sheet surface h in terms of the ice

thickness $H = h - b$, the surface gradient ∇h, and the mass balance M:

$$\begin{aligned}
\tilde{h}_t(x,t,\boldsymbol{z}) &= \nabla \cdot \left(\alpha \left(\frac{h^2(x,t)}{\beta(x,\boldsymbol{z})} + \Gamma H^{n+2}(x,t) |\nabla h(x,t)|^{n-1} \right) \nabla h(x,t) \right) + M(x,t) \\
h_t(x,t,\boldsymbol{z}) &= \tilde{h}_t, \quad \text{if } h(x,t,\boldsymbol{z}) > b(x) \text{ or } \tilde{h}_t(x,t,\boldsymbol{z}) > 0, \\
h(x,t,\boldsymbol{z}) &= 0, \quad \text{otherwise}, \\
h(x,0) &= \sqrt{\max\left(\frac{1}{2} - \frac{x^2}{900}, b^2(x) \right)}, \quad h(\pm L, t) = b(\pm L)
\end{aligned} \tag{8}$$

where $x \in D = (-L, L)$, $t \in (0, T]$, $\alpha = \frac{\rho g}{2}$ and $\Gamma = 2A(\rho g)^n/(n+2)$. In this paper we set the bed topography $b(x) = 0.01(1 + \cos(\frac{2\pi x}{20}))$[km], the forcing term $M(x,t) = -0.4 + \cos\left(\frac{\pi x}{L}\right)\left(1 + 0.5\sin\left(\frac{2\pi t}{T}\right)\right)$ [km yr^{-1}], $L = 40$ km, the Glen exponent $n = 3$, the flow rate factor $A = 1e-15$ Pa^n yr^{-1}, the ice density $\rho = 910$ kg m^{-3}, and the gravity acceleration $g = 9.81$ m s^{-2}.

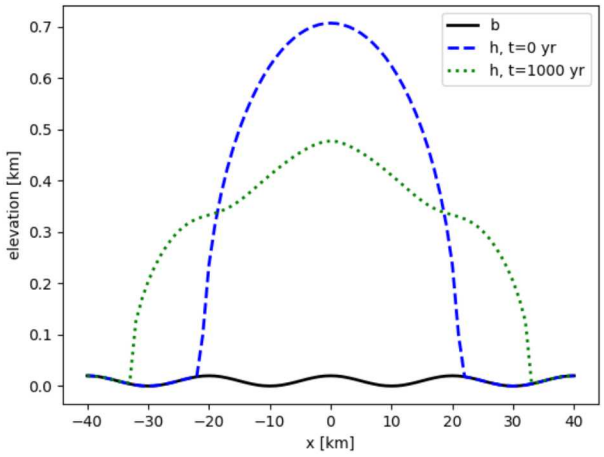

Figure 4. Ice dome problem at initial and final time.

The governing equations of the shallow ice approximation model are nonlinear but do have a similar structure to the well known linear diffusion equation $h_t(x,t,\boldsymbol{z}) = \nabla \cdot (D \nabla h(x,t)) + M(x,t)$. When the ice is both thick and has a large slope the nonlinear "diffusivity" D is large and ice flows downhill at a faster rate than when D is smaller. As the ice thins the solution changes more slowly. When an ice sheet is monotonically thinning the change in the ice sheet profile in the first 100 years is larger than that between 100 and 200 years.

We also note that as the ice sheet spreads and thins, ice covers regions that were previously dry. The resulting solution h at the ice-extent boundaries is not smooth, which can lead to larger numerical errors at the boundary. This can limit the accuracy of any approximation of QoI from the SIA model.

Discretization

In this paper we use central finite difference in space and a second-order explicit time-stepping scheme to solve the SIA equation. Unless stated otherwise, we use 129 spatial mesh points and adaptive time-stepping, which makes the physical discretization error significantly smaller than the errors in the surrogate models induces by finite sampling. The adaptive time-stepping chooses the largest time-step that satisfies the CFL condition at the current time. Our neural network approximations required the solution data for each random sample to have to be on the same time mesh. To satisfy this requirement we set N_t =1000 checkpoints $t_i, i = 1, \ldots, N_t$ equidistantly distributed in $[0, T = 1000yr]$. At a given time t, let t_i denote the previous checkpoint and $\Delta t^\star$ the adaptive time step, then the next timestep is satisfies $\Delta t = \min(\Delta t^\star, t_i - t)$.

Random Field

Predictions obtained from ice sheet models are subject to a wide range of uncertainties. One large source of uncertainty is the friction between the landmass and the ice sheet. Here we assume that this basal friction β is a random field represented by the Karhunen-Loéve expansion expansion (KLE)

$$\log(\beta(\hat{x}, \boldsymbol{z})) = \bar{k}(\hat{x}) + \sigma \sum_{k=1}^{n_z} \sqrt{\lambda_k}\phi_k(\hat{x}) z_k \qquad C(\hat{x}_1, \hat{x}_2) = \exp\left[-\frac{|\hat{x}_1 - \hat{x}_2|}{l}\right] \tag{9}$$

where $\hat{x} = \frac{(x-L)}{2L} \in [0, 1]$, $\{\lambda_k\}_{k=1}^{n_z}$ and $\{\phi_k(\hat{x})\}_{k=1}^{n_z}$ are, respectively, the eigenvalues and eigenfunctions of the covariance kernel C. In this paper we construct a KLE with $n_z = 20, 100$ or 500 independent and uniformly distributed random variables $z_k \in [-1, 1]$, $k \in [n_z]$. We also set $\sigma = 20$, $\bar{k}(\hat{x}) = 10^5(1 + \frac{9}{10}\cos(\frac{2\pi\hat{x}}{40}))$, and set the correlation length of the covariance kernel to $l = \frac{1}{100}$.

Quantity of interest and error computation

The computational expense of quantifying uncertainty in a model is dependent on the quantities of interest. In this paper our quantity of interest is the change of mass of the ice at time t with respect to the initial mass at time t_0. Since the bed is fixed in time, we can compute the QoI as

$$f(t) = \rho \int_D h(t) - h(t_0)\, dx$$

We compute the integral using a trapezoidal quadrature rule.

To measure the performance of an approximation, we will compute the error of the difference between the QoI f computed using the ice flow model and the surrogate prediction

$\hat{f}$. For a set of Q random samples $\{\boldsymbol{z}^{(j)}\}_{j=1}^{Q} \subset \Gamma$ drawn from the density ω of the uncertain variables $\boldsymbol{z}$, we define the ℓ^2 norm as

$$\|f\|_{\ell^2} = \sqrt{\sum_{j\in[Q]} |f(\boldsymbol{z}^{(j)})|^2}$$

We then compute the relative error at time t_i as

$$\mathcal{E}_{t_i}^f = \frac{\|f_{t_i} - \hat{f}_{t_i}\|_{\ell^2}}{\max\limits_{i=1\ldots,N_t} \|f_{t_i} - \hat{f}_{t_i}\|_{\ell^2}} \tag{10}$$

where f_{t_i} is the exact QoI and $\hat{f}_{t_i}$ is the approximation at time t_i.

Numerical Experiments

In this section we present a number of numerical experiments that highlight the efficacy of using different NN surrogates to approximate our QoI. We compare accuracy of the NN surrogates with the accuracy of polynomial chaos expansions trained on the same data.

In the following experiments, we generated 1000 uniformly distributed random samples, 900 of which are used to train the NN or PCE surrogate and the remaining 100 for testing the surrogate. In all the figures in this section we will plot the error $E_{t_i}^f$, defined in eq. (10) where Q represents the testing set of 100 samples, as a function of time.

Long Short Term Memory

In this section we investigate the accuracy of LSTM networks for varying number of random parameters. We first consider type-1 LSTM networks (Figure 2). The network input consists of the stochastic basal friction parameters, and our QoI (total mass change of the ice sheet f) at times $t_{n-1}, \ldots, t_{n-w}$ where w is the size of the windows. The output is the QoI at time t_n. The results, for $w = 10$ and for parameter spaces of dimension 20 and 100 are reported in Figure 5. The performance of LSTM is dependent on the network hyperparameters. The various curves in Figure 5 represent the error in LSTM surrogates with different values for important hyperparameters. Specifically, we consider the number of layers (1 to 4), how many weights per layer (5 to 500), the dropout (0.5) and the learning rate (0.0005 to 0.01).

In order to improve the accuracy of the prediction, we can also include the forcing term (that depends on time) as input of the LSTM. Results, for a window of size 50, are reported in Figure 6. The Figure also illustrates how the addition of dense preprocessing (left plot) significantly improves the accuracy of the network.

In Figure 3 we plot the error in type-2 LSTM surrogates. The type-2 LSTM represents the limit case of the type-1 LSTM as the window size is reduced to 1. The results are

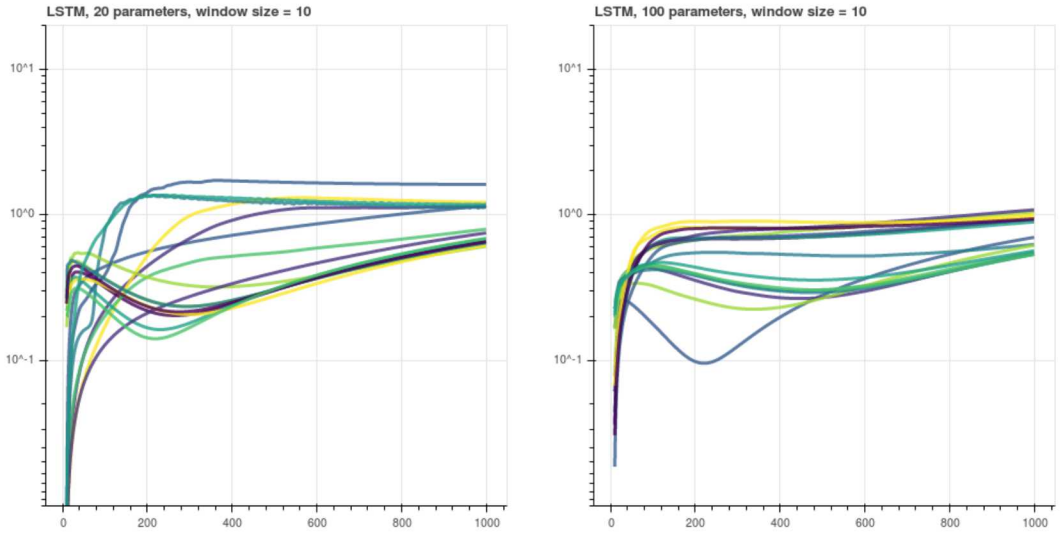

Figure 5. Errors for Type-1 LSTM networks with a window of size 10 [yr], for different hyperparameters and for 20 stochastic parameters (left) and 100 stochastic parameters (right).

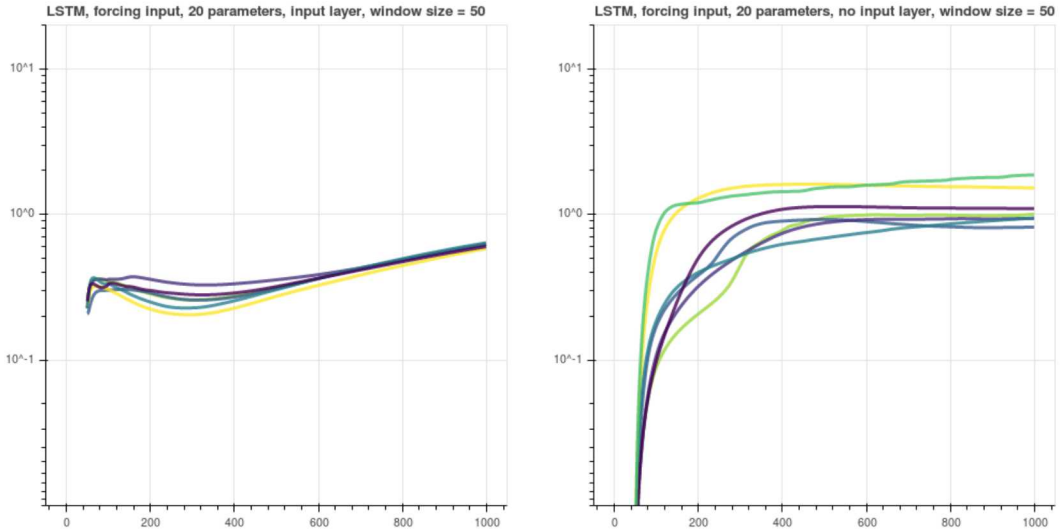

Figure 6. Errors for Type-1 LSTM networks with a window of size 50 [yr], using the forcing term as input, for different hyperparameters and for 20 stochastic parameters. The results on the left have been computed using networks with dense preprocessing, whereas the ones on the right without it.

shown in Figure 7. In this case the training data where not enough for properly training the network, so we used a data-augmentation technique, consisting in expanding the training set

by adding Gaussian noise ($\sigma = 0.05$) to the primary input of the LSTM (the QoI value).

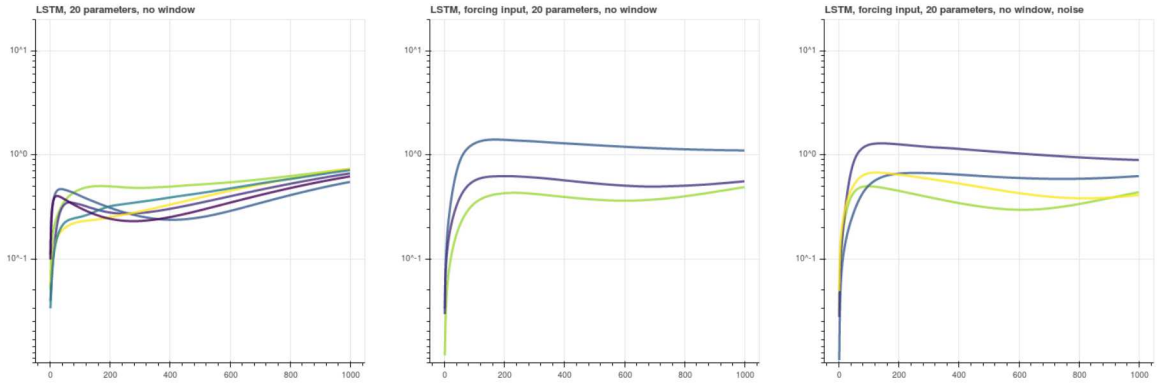

Figure 7. Errors for Type-2 LSTM networks, for different hyperparameters and for 20 stochastic parameters, with and without input forcing. For the plot on the right we expanded the training set by adding noise.

LSTMs can be used to predict scalar or vector-valued quantities. Our previous experiments have considered the prediction of a scalar quantity, specifically the mass change at time t. Here we use LSTM networks to predict the surface elevation h (solution of our PDE model) at each grid point and time instant t_i as input o the neural network instead of the QoI. In this case the neural network will predict the solution $\hat{h}_{t_n}(x_i)$ and the QoI $\hat{f}_{t_n}$ is computed integrating $\hat{h}_{t_n} - h_{t_0}$. The results are shown in Figure 8. Note that in this case one of the networks (purple line) is not stable.

Multi-layer Perceptron

In this section we investigate the performance of more standard MLP networks. Unlike LSTMs which predict the QoI at every time point MLP are used to predict the QoI at a subset of set of time instants. Here we are interested in predicting the QoI every 50 years. The plot shown in Figure 9 depicts the error in MLP surrogates for for 20, 100 and 500 stochastic parameters[1]. Upon visual inspection, one can easily classify, especially in the first two plots, the oscillating and low accurate results from more accurate and smooth ones. The latter have been obtained with a learning rate of 0.01, whereas the former have been obtained with learning rates equal or smaller than 0.001.

Comparing Neural networks and sparse polynomial chaos expansions

In this section we compare MLP surrogates with polynomial chaos expansion surrogates constructed using the approach described in Section. We only compare against MLP because

[1]For the problems with 500 parameters we discretize the spatial domain using 512 mesh points.

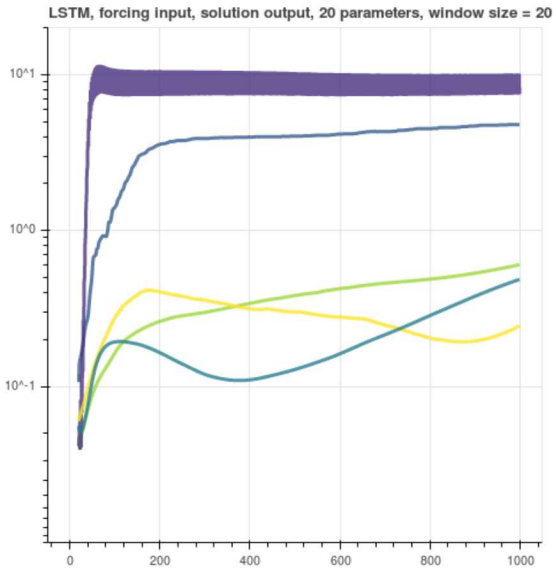

Figure 8. Errors for Type-2 LSTM networks, trained on the ice surface elevation for different hyperparameters and for 20 stochastic parameters. For one of the networks, the predicted QoI is oscillating (purple line).

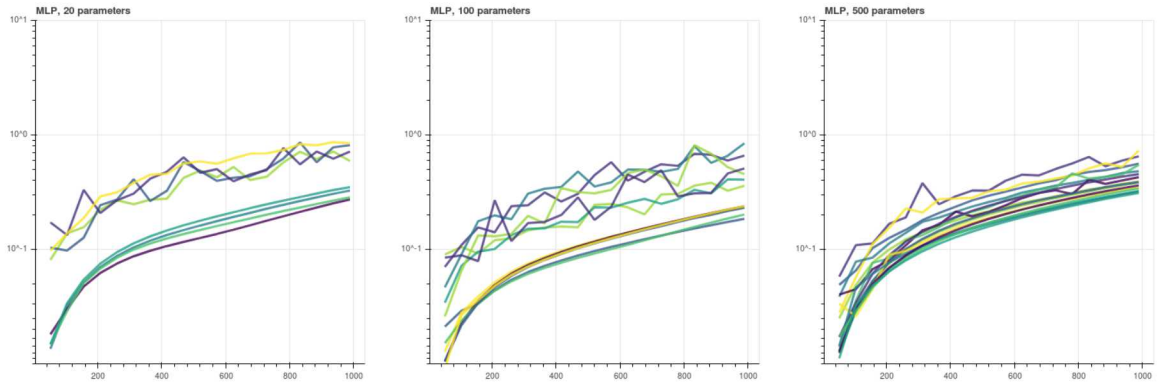

Figure 9. Errors for MLP networks, for 20 (left), 100 (center) and 500 (right) stochastic parameters.

our previous experiments suggest that they always outperform LSTM surrogates.

The error in the PCE and MLP surrogates, as a function of time, for 20, 100 and 500 stochastic parameters is depicted in Figure 10. Here we use a total-degree basis consisting of the tensor-product of univariate Legendre polynomials. We use quadratic approximations when we have 20 and 100 parameters and linear for 500 parameters. As before we choose to predict the QoI every 50 years and build as many PCEs surrogates as the number of time instants we want to predict.

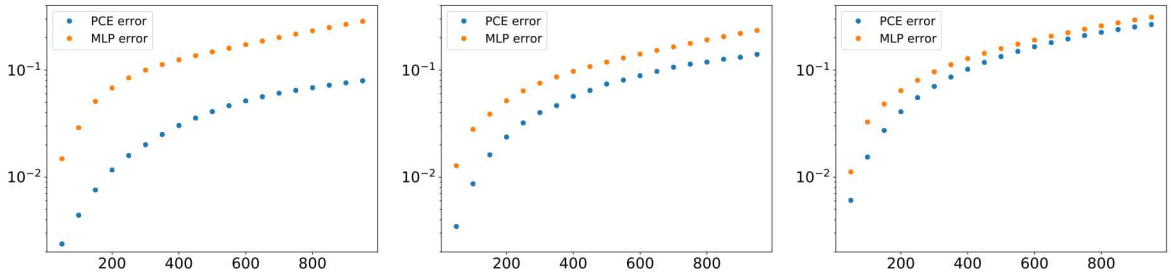

Figure 10. Error comparison between PCE surrogates an the best MLP surrogates, for 20 (left), 100 (center) and 500 (right) stochastic parameters.

In all cases the error of both surrogate types increases with time. This is due to the fact that the variance, and thus complexity of the quantity of interest as a function of the random variables, increases with time. At early times the ice sheet is insensitive to the random parameters because the dynamics have not had long enough to effect the initial profile of the ice sheet. Although error increases with time, the rate of increase in the error decreases with time. This is because as the ice sheet diffuses it slows down thus reducing the impact of the ice sheet dynamics.

Discussion

We investigated the use of NNs and in particular of RNNs for building surrogates of a rather simple ice sheet computational model. In our preliminary study, the LSTM networks performed poorly both in terms of accuracy and of computational cost when compared to MLP networks or to PCE. Our working hypothesis that recurrent neural networks could naturally approximate a dynamical system is not supported by our experiments. In fact, most of the LSTM networks feature predictions with relative errors larger than 50%, which makes them hardly usable even in a multi-fidelity uncertainty quantification framework [30]. Providing the forcing term as input to the NNs does not significantly improve the accuracy (Figures 5, 6), and in some cases it makes it worse, which is surprising given that the variation in the QoI in this model is mainly due to the integral of the forcing term.

The hyperparameters of neural networks have a significant impact on the accuracy of the resulting surrogate. One important hyperparameter is the learning rate. Setting learning rate to about 0.01 produces the most accurate NN, while training NNs with smaller or lager rates is less effective (Figure 9). Dense preprocessing seems to help as well in improving the network predictive skills (Figure 6). In terms of number of layers, networks with 2-3 layers perform better than those with only one layer, although deeper networks do not seem to significantly improve the accuracy. One-layer networks performs well when tested on predicting the QoI $\hat{f}_{t_n}$ given exact inputs $f_{t_{n-1}}, f_{t_{n-2}}, \ldots, f_{t_{n-w}}$ (which is the setting used in the training phase, called *teacher forcing*), but they perform poorly when the input values

are generated from the network outputs at previous time steps (which is the setting used in practice to emulate the computational model). In fact, our numerical experiments show that the *score*, the mean squared error over the testing set, of the NN after the training phase is poorly correlated with the actual relative errors $\mathcal{E}^f_{t_i}$ we are interested in. This means that the network is performing well locally but it cannot compensate for accumulation errors. This suggests that the way we trained the LSTM networks, which is standard in NN applications, does not work well in our case.

MLP networks outperform the more complex LSTM in terms of accuracy and training and evaluation costs, despite the fact that the LSTM training data is bigger due to the temporal dimension. Unlike RNNs that can in principle be used to predict any time instant, MLP (and PCE), can only be trained on a set of time instants which must be specified a priori. In most application this is not a practical limitation and in our application we choose to predict the QoI every time step, for a total of 20 time instants.

Finally we compared the best MLP networks with the more established PCE surrogates, and the latter turned out to be significantly more accurate the MLP networks (Figure 10). However, an interesting trend appearing in such figure is that the MLP error is less sensitive to the number of parameters than the PCE ones, suggesting that for realistic 2D/3D applications, for which the number of random variable used to represent the friction will increase, the MLP could perform better than PCE.

Given the different and not optimized implementations of the PCE, MLP and LSTM surrogates, it is hard to precisely compare the approaches in terms of computational cost. Approximately, PCE and MLP have similar computational costs, while LSTM cost is 2-3 orders of magnitude higher.

Anticipated outcomes and impacts

This study represents one of the few works in the literature where Recurrent Neural Networks (LSTM) have been used to emulate a computational dynamical model. A strength of this work is the rigorous quantification of the NN surrogate accuracy and its comparison with PCE, a technique well established and well understood in the field of uncertainty quantification.

In our study the LSTM surrogate performed poorly in terms of accuracy and training costs. In fact, none of the LSTM networks we trained could be used in an effective way for our applications. Given the preliminary nature of this work, we shall not draw any hasty conclusion about the possibility of generating competitive RNN surrogates of computational dynamical models. In fact the complexity of neural networks (large number of hyperparameters, architectures and of training algorithms) makes it extremely hard to identify a setting that would work well with new target applications like the one considered here. As a matter of fact, only a decade ago neural networks would perform rather poorly when targeting the same class of problems that in recent years determined their undisputed success.

The outcome of this work could be used to identify promising research areas. Our experiments indicate that LSTM networks with small *score* do not necessarily perform well as surrogates. This might indicate a structural limit of the network architecture that hinders its capability of capturing the global behavior of a dynamical system, or that the *teacher forcing* training, based on a local loss function, is not suited for our problem. We plan to investigate different training techniques in a future work. An idea is to have a separate network to predict long-term errors and use this as an auxiliary loss for the RNN. Along the same lines, we plan to significantly increase (e.g. to 50 years) the network time step for type-2 LSTM networks. This should help the network in capturing the global behavior of the dynamical system. In addition we think it would be worth gaining more mathematical understanding of RNNs by, e.g., deriving continuous models for RNNs by extending the recent works for residual neural networks in [17]. An alternative approach worth pursuing, is to include time and space coordinates as networks inputs in additions to the basal parameters, and build two networks that directly approximate the solutions $h(x, t, \boldsymbol{z})$ and the residual of the ice flow model, as done in [31].

We obtained more promising results when using the standard MLP networks both in terms of accuracy and computational costs. Despite the fact that the PCE surrogate performed better than all the MLP networks considered here, MLP network showed to be less sensitive to the dimension of the parameter space than the PCE approach. This property could be of paramount importance when considering more realistic 2D or 3D problem. We plan to investigate the behavior of MLP for 2D problems in a future work. Another research direction we think it is worth exploring, is to use the weights of the input layer to determine what parameters (or combination of parameters) are not affecting the prediction of the QoI. This could allow a significant reduction of the parameter space, which is greatly beneficial when performing UQ analysis.

The use of neural networks is gaining popularity in scientific computing. In this context neural networks are being used, e.g., to discover equations by estimating coefficients of PDE equations [32], to directly discover the solutions of nonlinear PDEs [31] and for building model surrogates. These are outside the set of applications that are typically targeted by NNs and there is not enough understanding to confidently predict in what cases a particular NN architecture would be advantageous. Our study is part of this initial exploration of possible applications that can benefit from the use of NNs. While preliminary in nature, our work shows that a rather straightforward application of RNNs (LSTM in particular) can produce underwhelming results when emulating a nonlinear computational model. It also points out the aforementioned possible research directions, that could help shaping the research of newly funded projects such as DOE MMCCs PhILMs (Collaboratory on Mathematics and Physics-informend Learning Machines for Multiscale and Multiphysics Problems) and the upcoming ASCR FY20 scientific machine learning Funding Opportunity Announcements. The information gained in this work will certainly be useful for directing the research on surrogates for the UQ analysis in ProSPect [2].

Conclusion

In this paper we compare the error in two types of neural networks approximations of the change in mass of an ice sheet model as a function of random variables which parametrize the friction between the ice and the bedrock. Specifically, we use Long Short Term Memory (LSTM) networks in an attempt to leverage the temporal output of the transient model and Multi-Layer Perceptron (MLP) networks which are simpler and easier to train. Despite the LSTM being able to separate the temporal data to create more training data than available to the MLP, in our experiments the accuracy of the LSTM network is worse than the accuracy of the MLP network and insufficient to be used for uncertainty quantification. Compared on the same problems, simple linear and quadratic Polynomial Chaos Expansion (PCE) approximations are more accurate than the LSTM and MLP ones, for a number of parameters varying from 20 to 100 to 500 variables. The accuracy of the PCE approximation deteriorates as the number of parameters increase, while the MLP accuracy is not significantly affected by the dimensionality of the parameter space.

References

[1] DOE earth system modeling (ESM) program. https://science.energy.gov/ber/research/cesd/earth-system-modeling-program/.

[2] SciDAC BER ProSPect (probabilistic sea-level projections from ice sheets and earth system models) project. https://climatemodeling.science.energy.gov/projects/probabilistic-sea-level-projections-ice-sheet-and-earth-system-models.

[3] D. R. Baral, K. Hutter, and R. Greve. Asymptotic Theories of Large-Scale Motion, Temperature, and Moisture Distribution in Land-Based Polythermal Ice Sheets: A Critical Review and New Developments. *Applied Mechanics Reviews*, 54:215, 2001.

[4] Ilias Bilionis, Nicholas Zabaras, Bledar A. Konomi, and Guang Lin. Multi-output separable gaussian process: Towards an efficient, fully bayesian paradigm for uncertainty quantification. *Journal of Computational Physics*, 241:212 – 239, 2013.

[5] Géraud Blatman and Bruno Sudret. Adaptive sparse polynomial chaos expansion based on least angle regression. *Journal of Computational Physics*, 230(6):2345 – 2367, 2011.

[6] E.J. Candes and T. Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *Information Theory, IEEE Transactions on*, 52(12):5406–5425, Dec 2006.

[7] Emmanuel J. Candés, Justin K. Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on Pure and Applied Mathematics*, 59(8):1207–1223, 2006.

[8] Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. Atomic Decomposition by Basis Pursuit. *SIAM Review*, 43(1):129–159, January 2001.

[9] J.A. Church, P.U. Clark, A. Cazenave, J.M. Gregory, S. Jevrejeva, A. Levermann, M.A. Merrifield, G.A. Milne, R.S. Nerem, P.D. Nunn, A.J. Payne, W.T. Pfeffer, D. Stammer, and A.S. Unnikrishnan. *Sea Level Change*, book section 13, pages 1137–1216. Cambridge University Press, Cambridge, United Kingdom and New York, NY, USA, 2013.

[10] D.L. Donoho. Compressed sensing. *Information Theory, IEEE Transactions on*, 52(4):1289–1306, April 2006.

[11] D.L. Donoho, M. Elad, and V.N. Temlyakov. Stable recovery of sparse overcomplete representations in the presence of noise. *Information Theory, IEEE Transactions on*, 52(1):6–18, Jan 2006.

[12] A. Doostan and H. Owhadi. A non-adapted sparse approximation of PDEs with stochastic inputs. *Journal of Computational Physics*, 230(8):3015 – 3034, 2011.

[13] A. Doostan, A. Validi, and G. Iaccarino. Non-intrusive low-rank separated approximation of high-dimensional stochastic models. *Computer Methods in Applied Mechanics and Engineering*, 263:42–55, 2013.

[14] Oliver G. Ernst, Antje Mugler, Hans-Jrg Starkloff, and Elisabeth Ullmann. On the Convergence of Generalized Polynomial Chaos Expansions. *ESAIM: Mathematical Modelling and Numerical Analysis*, 46(02):317–339, 2012.

[15] R. G. Ghanem and P. D. Spanos. *Stochastic finite elements: a spectral approach*. Springer-Verlag New York, Inc., 1991.

[16] A. Gorodetsky and J.D. Jakeman. Gradient-based optimization for regression in the functional tensor-train format. *Journal of Computational Physics*, 2018.

[17] Eldad Haber and Lars Ruthotto. Stable architectures for deep neural networks. *Inverse Problems*, 34(1):014004, 2018.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[19] Sepp Hochreiter and Jrgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[20] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359 – 366, 1989.

[21] J.D. Jakeman, M.S. Eldred, and K. Sargsyan. Enhancing ℓ_1-minimization estimates of polynomial chaos expansions using basis selection. *Journal of Computational Physics*, 289(0):18 – 34, 2015.

[22] J.D. Jakeman, F. Franzelin, A. Narayan, M. Eldred, and D. Pflüger. Polynomial chaos expansions for dependent random variables. 2018. submitted.

[23] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[24] L. Mathelin. Quantification of uncertainty from high-dimensional scattered data via polynomial approximation. *International Journal for Uncertainty Quantification*, 4(3), 2014.

[25] A. Narayan and J.D. Jakeman. Adaptive Leja sparse grid constructions for stochastic collocation and high-dimensional approximation. *SIAM Journal on Scientific Computing*, 36(6):A2952–A2983, 2014.

[26] F. Nobile, R. Tempone, and C.G. Webster. A sparse grid stochastic collocation method for partial differential equations with random input data. *SIAM Journal on Numerical Analysis*, 46(5):2309–2345, 2008.

[27] I. V. Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.

[28] Jaideep Pathak, Brian Hunt, Michelle Girvan, Zhixin Lu, and Edward Ott. Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Phys. Rev. Lett.*, 120:024102, Jan 2018.

[29] Arendt PD, Apley DW, and Chen W. Quantification of model uncertainty: Calibration, model discrepancy, and identifiability. *ASME. J. Mech. Des.*, 134(10):100908–100908–12, 2012.

[30] B. Peherstorfer, K. Willcox, and M. Gunzburger. Survey of multifidelity methods in uncertainty propagation, inference, and optimization. *SIAM Review*, 60(3):550–591, 2018.

[31] Mazier Raissi, Paris Perdikaris, and George E. Karniadakis. Physics informed deep learning (part I): Data-driven solutions of nonlinear partial differential equations. 2017.

[32] Mazier Raissi, Paris Perdikaris, and George E. Karniadakis. Physics informed deep learning (part II): Data-driven discovery of nonlinear partial differential equations. 2017.

[33] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[34] T.F. Stocker, D. Qin, G.-K. Plattner, L.V. Alexander, S.K. Allen, N.L. Bindoff, F.-M. Bre??on, J.A. Church, U. Cubasch, S. Emori, P. Forster, P. Friedlingstein, N. Gillett, J.M. Gregory, D.L. Hartmann, E. Jansen, B. Kirtman, R. Knutti, K. Krishna Kumar, P. Lemke, J. Marotzke, V. Masson-Delmotte, G.A. Meehl, I.I. Mokhov, S. Piao, V. Ramaswamy, D. Randall, M. Rhein, M. Rojas, C. Sabine, D. Shindell, L.D. Talley, D.G. Vaughan, and S.-P. Xie. *Technical Summary*, book section TS, pages 33–115. Cambridge University Press, Cambridge, United Kingdom and New York, NY, USA, 2013.

[35] B. Sudret. Global sensitivity analysis using polynomial chaos expansions. *Reliability Engineering & System Safety*, 93(7):964–979, JUL 2008.

[36] W. V. Sweet, R. E. Kopp, C. P. Weaver, J. Obeysekera, R. M. Horton, E. R. Thieler, and C. Zervas. Global and regional sea level rise scenarios for the united states, 2017.

[37] D. Xiu and J. S. Hesthaven. High-order collocation methods for differential equations with random inputs. *SIAM Journal on Scientific Computing*, 27(3):1118–1139, January 2005.

[38] D. Xiu and G. E. Karniadakis. The Wiener–Askey polynomial chaos for stochastic differential equations. *SIAM Journal on Scientific Computing*, 24(2):619–644, January 2002.

[39] L. Yan, L. Guo, and D. Xiu. Stochastic collocation algorithms using ℓ_1-minimization. *International Journal for Uncertainty Quantification*, 2(3):279–293, 2012.

DISTRIBUTION:

1	MS 0359	D. Chavez, LDRD Office, 1911
1	MS 0899	Technical Library, 9536 (electronic copy)


Sandia National Laboratories