Physics-guided Neural Networks (PGNN): An Application in Lake Temperature Modeling

Anuj Karpatne*
karpa009@umn.edu

William Watkins†
wwatkins@usgs.gov

Jordan Read†
jread@usgs.gov

Vipin Kumar*
kumar001@umn.edu

Abstract

This paper introduces a novel framework for combining scientific knowledge of physics-based models with neural networks to advance scientific discovery. This framework, termed as physics-guided neural network (PGNN), leverages the output of physics-based model simulations along with observational features to generate predictions using a neural network architecture. Further, this paper presents a novel framework for using physics-based loss functions in the learning objective of neural networks, to ensure that the model predictions not only show lower errors on the training set but are also scientifically consistent with the known physics on the unlabeled set. We illustrate the effectiveness of PGNN for the problem of lake temperature modeling, where physical relationships between the temperature, density, and depth of water are used to design a physics-based loss function. By using scientific knowledge to guide the construction and learning of neural networks, we are able to show that the proposed framework ensures better generalizability as well as scientific consistency of results.

1 Introduction

Data science has become an indispensable tool for knowledge discovery in the era of big data, as the volume of data continues to explode in practically every research domain. Recent advances in data science such as deep learning have been immensely successful in transforming the state-of-the-art in a number of commercial and industrial applications such as natural language translation and image classification, using billions or even trillions of data samples. In light of these advancements, there is a growing anticipation in the scientific community to unlock the power of data science methods for accelerating scientific discovery [1, 4, 8, 19].

However, a major limitation in using "black-box" data science models, that are agnostic to the underlying scientific principles driving real-world phenomena,

---
*Department of Computer Science, University of Minnesota.
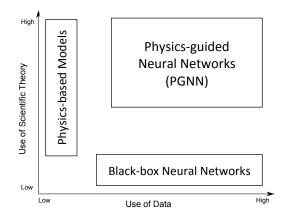†United States Geological Survey.

Figure 1: A schematic representation of physics-guided neural networks in the context of other knowledge discovery approaches that either use physics or data. The X-axis measures the use of data while the Y-axis measures the use of scientific knowledge.

is their sole dependence on the available labeled data, which is often limited in a number of scientific problems. In particular, a black-box data science model for a supervised learning problem can only be as good as the representative quality of the labeled data that it is fed with. When the size of both the training and test sets are small, it is easy to learn spurious relationships that look deceptively good on both training and test sets (even after using standard methods for model evaluation such as cross-validation), but do not generalize well outside the available labeled data. A more serious concern with black-box applications of data science models is the lack of scientific consistency of its predictions with respect to the known laws of physics. Hence, even if a black-box model achieves somewhat more accurate performance but lacks the ability to adhere to mechanistic understandings of the underlying physical processes, it cannot be used as a basis for subsequent scientific developments.

On the other end of the spectrum, physics-based models, which are founded on core scientific principles, strive to advance our understanding of the physical world by learning explainable relationships between in-

put and output variables. These models have been the cornerstone of knowledge discovery in a wide range of scientific and engineering disciplines. There are two basic forms in which physical knowledge is generally available: (a) as physics-based rules or equations that dictate relationships between physical variables, and (b) in the form of numerical models of complex physical systems, e.g., simulations of dynamical systems that are heavily used in computational chemistry, fluid dynamics, climate science, and particle physics. While these models have significantly advanced our understanding of the physical universe, they are limited in their ability to extract knowledge directly from data and are mostly reliant only on the available physics. For example, many physics-based models use parameterized forms of approximations for representing complex physical processes that are either not fully understood or cannot be solved using computationally tractable methods. Calibrating the parameters in physics-based models is a challenging task because of the combinatorial nature of the search space. In particular, this can result in the learning of over-complex models that lead to incorrect insights even if they appear interpretable at a first glance. For example, these and other challenges in modeling hydrological processes using state-of-the-art physics-based models were the subject of a series of debate papers in Water Resources Research (WRR) [5, 9, 12]. The dichotomy between physics-based models and black-box neural network models is schematically depicted in Figure 1, where they both occupy the two extreme ends of knowledge discovery, either relying only on the data (black-box neural networks) or only on scientific knowledge (physics-based models).

In this paper, we introduce a novel framework of knowledge discovery in scientific problems that combines the power of neural networks with physics-based models, termed as physics-guided neural networks (PGNN). There are two primary contributions of this work. First, we present an approach to create hybrid combinations of physics-based models and neural network architectures to make full use of both physics and data. Second, we present a novel framework for training neural network architectures using the knowledge contained in physics-based equations, to ensure the learning of physically consistent solutions. To demonstrate the framework of PGNN, we consider the illustrative problem of modeling the temperature of water in a lake at varying depths and times, using input drivers as well as physics-based model simulations. For this problem, we exploit a key physical relationship between the temperature, density, and depth of water in the form of physics-based loss function.

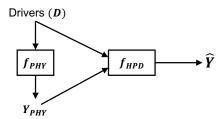The remainder of this paper is organized as follows.

Figure 2: A schematic illustration of a hybrid-physics-data (HPD) model, where the output of a physics-based model, Y_{PHY}, is used as another feature in the data science model, along with the drivers.

Section 2 presents the generic framework of physics-guided neural networks that can be applied in any domain with some availability of scientific knowledge. Section 3 presents the specific PGNN formulation for the illustrative problem of lake temperature modeling. Section 4 describes the evaluation procedure and presents experimental results, while Section 5 provides concluding remarks.

2 Physics-guided Neural Networks

In this section, we describe the generic framework of physics-guided neural networks (PGNN) that involves two key steps: (a) creating hybrid combinations of physics-based models and neural networks, termed hybrid-physics-data (HPD) models, and (b) using scientific knowledge as physics-based loss functions in the learning objective of neural networks, as described in the following.

2.1 Constructing Hybrid-Physics-Data Models

Consider a predictive learning problem where we are given a set of input drivers, $\mathbf{D}$, that are physically related to a target variable of interest, Y. A standard approach is to train a data science model, e.g., a neural network, $f_{NN} : \mathbf{D} \rightarrow Y$, over a set of training instances, which can then be used to produce estimates of the target variable, $\hat{Y}$. Alternatively, we can also use a physics-based numerical model, $f_{PHY} : \mathbf{D} \rightarrow Y$, to simulate the value of the target variable, Y_{PHY}, given its physical relationships with the input drivers. Analogous to the process of training, physics-based models often require "calibrating" their model parameters using observational data—a process that is both time-consuming and label-expensive. Furthermore, Y_{PHY} may provide an incomplete representation of the target variable due to simplified or missing physics in f_{PHY}, thus resulting in model discrepancies w.r.t. observations. Hence, the basic goal of HPD modeling is to combine f_{PHY} and f_{NN} so as to overcome their complementary deficiencies and leverage information in both physics and data.

One simple way for combining f_{PHY} and f_{NN} is to use the simulated outputs of the physics-based model, Y_{PHY}, as another input in the data science model (neural network) along with the drivers, $\mathbf{D}$. This results in the following HPD model:

$$f_{HPD} : \mathbf{X} = [\mathbf{D},\ Y_{PHY}] \rightarrow Y,$$

which is schematically illustrated in Figure 2. In this setup, notice that if the physics-based model is accurate and Y_{PHY} perfectly matches with observations of Y, then the HPD model can learn to predict $\hat{Y} = Y_{PHY}$. However, if there are systematic discrepancies (biases) in Y_{PHY}, then f_{HPD} can learn to complement them by extracting complex features from the space of input drivers and thus reducing our knowledge gaps.

2.2 Using Physics-based Loss Functions A standard approach for training the HPD model described in Figure 2 is to minimize the empirical loss of its model predictions, $\hat{Y}$, on the training set, while maintaining low model complexity as follows:

$$(2.1) \qquad \arg\min_{f}\ \ Loss(\hat{Y}, Y) + \lambda\ R(f),$$

where $R(.)$ measures the complexity of a model and λ is a trade-off hyper-parameter. However, the effectiveness of any such training procedure is limited by the size of the labeled training set, which is often small in many scientific problems. In particular, there is no guarantee that model trained by minimizing Equation 2.1 will produce results that are *consistent* with our knowledge of physics. Hence, we introduce physics-based loss functions to guide the learning of data science models to physically consistent solutions as follows.

Let us denote the physical relationships between the target variable, Y, and other physical variables, $\mathbf{Z}$ using the following equations:

$$\begin{aligned} \mathcal{G}(Y, \mathbf{Z}) &= 0, \\ (2.2) \qquad \mathcal{H}(Y, \mathbf{Z}) &\leq 0. \end{aligned}$$

Note that $\mathcal{G}$ and $\mathcal{H}$ are generic forms of physics-based equations that can either involve algebraic manipulations of Y and $\mathbf{Z}$ (e.g., in the laws of kinematics), or their partial differentials (e.g., in the Navier–Stokes equation for studying fluid dynamics or in the Schrödinger equation for studying computational chemistry). One way to measure if these physics-based equations are being violated in the model predictions, $\hat{Y}$, is to evaluate the following physics-based loss function:

$$(2.3)\ \ Loss.PHY(\hat{Y}) = ||\mathcal{G}(\hat{Y}, \mathbf{Z})||^2 + \text{ReLU}\,(\mathcal{H}(\hat{Y}, \mathbf{Z})),$$

where ReLU(.) denotes the rectified linear unit function. Since $Loss.PHY$ does not require actual observations of the target variable, Y, it can be evaluated even on unlabeled data instances, in contrast to traditional loss functions. The complete learning objective of PGNN involving $Loss.PHY$ can then be stated as:

$$(2.4) \quad \arg\min_{f}\ \ \underbrace{Loss(\hat{Y}, Y)}_{\text{Empirical Error}}\ +\ \underbrace{\lambda\ R(f)}_{\text{Structural Error}}\ + $$

$$\underbrace{\lambda_{PHY}\ Loss.PHY(\hat{Y})}_{\text{Physical Inconsistency}},$$

where λ_{PHY} is the hyper-parameter that decides the relative importance of minimizing physical inconsistency compared to the empirical loss and the model complexity. Since the known laws of physics are assumed to hold equally well for any unseen data instance, ensuring physical consistency of model outputs as a learning objective in PGNN can help in achieving better generalization performance even when the training data is small and not fully representative. Additionally, the output of a PGNN model can also be interpreted by a domain expert and ingested in scientific workflows, thus leading to scientific advancements.

There are several optimization algorithms that can be used for minimizing Equation 2.4, e.g., the stochastic gradient descent (SGD) algorithm and its variants that have found great success in training deep neural networks. In particular, the gradients of $Loss.PHY$ w.r.t model parameters can be easily computed using the automatic differentiation procedures available in standard deep learning packages. This makes neural networks a particularly suited choice for incorporating physics-based loss functions in the learning objective of data science models.

3 PGNN for Lake Temperature Modeling

In this section, we describe our PGNN formulation for the illustrative problem of modeling the temperature of water in lakes. In the following, we first provide some background information motivating the problem of lake temperature modeling, and then describe our PGNN approach.

3.1 Background: Lake Temperature Modeling The temperature of water in a lake is known to be an ecological "master factor" [10] that controls the growth, survival, and reproduction of fish (e.g., [17]). Warming water temperatures can increase the occurrence of aquatic invasive species [15, 18], which may displace fish and native aquatic organisms, and result in more harmful algal blooms (HABs) [6, 13]. Understanding tem-
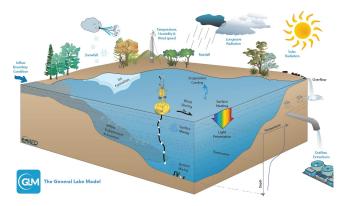
Figure 3: A pictorial description of the physical processes governing the dynamics of temperature in a lake. Figure courtesy: [7]. (Note: Figures in this paper are best viewed in color.)

perature change and the resulting biotic "winners and losers" is timely science that can also be directly applied to inform priority action for natural resources. Accurate water temperatures (observed or modeled) are critical to understanding contemporary change, and for predicting future thermal for economically valuable fish.

Since observational data of water temperature at broad spatial scales is incomplete (or non-existent in some regions) high-quality temperature modeling is necessary. Of particular interest is the problem of modeling the temperature of water at a given depth[1], d, and on a certain time, t. This problem is referred to as 1D-modeling of temperature (depth being the single dimension). A number of physics-based models have been developed for studying lake temperature, e.g., the state-of-the-art general lake model (GLM) [7]. This model captures a variety of physical processes governing the dynamics of temperature in a lake, e.g., the heating of the water surface due to incoming shortwave radiation from the sun, the attenuation of radiation beneath the surface and the mixing of layers with varying energies at different depths, and the dissipation of heat from the surface of the lake via evaporation or longwave radiation, shown pictorially in Figure 3. We use GLM as our preferred choice of physics-based model for lake temperature modeling.

The GLM has a number of parameters (e.g., parameters related to vertical mixing, wind energy inputs, and water clarity) that needs to be custom-calibrated for each lake if some training data is available. The basic idea behind these calibration steps is to run the model for each possible combination of parameter values and select the one that has maximum agreement with the

observations. Because this step of custom-calibrating is both labor- and computation-intensive, there is a trade-off between increasing the accuracy of the model and expanding the feasability of study to a large number of lakes.

3.2 Proposed PGNN Formulation We consider the physical variables governing the dynamics of lake temperature at every depth and time-step as the set of input drivers, $\mathbf{D}$. This includes meteorological recordings at the surface of water such as the amount of solar radiation at different wavelengths, wind speed, and air temperature, as well as the value of depth and the day of the year. To construct an HPD model of the type shown in Figure 2, we use simulations of lake temperature from the GLM, Y_{PHY}, along with the input drivers $\mathbf{D}$ at every depth and time-step to obtain the augmented set of features,

$$\mathbf{X} = [\mathbf{D},\ Y_{PHY}].$$

We adopt a basic multi-layer perceptron architecture to regress the temperature, Y, on any given depth and time, using $\mathbf{X}$. For a fully-connected network with L hidden layers, this amounts to the following modeling equations relating the input features, $\mathbf{x}$, to its target prediction, $\hat{y}$:

$$(3.5) \quad \mathbf{z}_1 \quad = \quad \mathbf{W}_1^T \mathbf{x} + \mathbf{b}_1$$
$$(3.6) \quad \mathbf{z}_i \quad = \quad \mathbf{W}_i^T \mathbf{a}_{i-1} + \mathbf{b}_i \quad \forall\ i = 2 \text{ to } L$$
$$(3.7) \quad \mathbf{a}_i \quad = \quad f(\mathbf{z}_i) \quad \forall\ i = 1 \text{ to } L$$
$$(3.8) \quad \hat{y} \quad = \quad \mathbf{w}_{L+1}^T \mathbf{a}_L + b_{L+1}$$

where $(\mathbf{W}, \mathbf{b}) = \{(\mathbf{W}_i, \mathbf{b}_i)\}_1^{L+1}$ represents the set of weight and bias parameters across all hidden and output layers, and f is the activation function used at the hidden layers. We use the mean squared error as our choice of loss function and L_1 and L_2 norms of network weights, $\mathbf{W}$ as regularization terms in Equation 2.1 as follows:
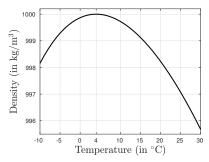
$$(3.9) \quad Loss(\hat{Y}, Y) \quad = \quad \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2,$$
$$(3.10) \quad \lambda\ R(\mathbf{W}) \quad = \quad \lambda_1 ||\mathbf{W}||_1 + + \lambda_2 ||\mathbf{W}||_2,$$
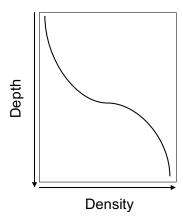
where $\{\mathbf{x}, y\}_1^n$ is the set of training instances.

To incorporate the knowledge of physics as a loss function in the training of neural networks, we employ a key physical relationship between the temperature, density, and depth of water as our physics-based equation (Equation 2.2). In the following, we introduce the two key components of this physical relationship and describe our approach for using it to ensure the learning of physically consistent results.

---

[1]Depth is measured in the direction from the surface of the water to the lake bottom.

(a) Temperature–Density Relationship

(b) Density–Depth Relationship

Figure 4: Plots of physical relationships between temperature, density, and depth of water that serve as the basis for introducing physical consistency in PGNN.

3.2.1 Temperature–Density Relationship: The temperature, Y, and density, ρ, of water are non-linearly related to each other according to the following known physical equation [11]:

(3.11)
$$\rho = 1000 \times \left(1 - \frac{(Y + 288.9414) \times (Y - 3.9863)^2}{508929.2 \times (Y + 68.12963)}\right)$$

Figure 4(a) shows a plot of this relationship between temperature and density, where we can see that water is maximally dense at 4°Celsius (due to the hydrogen bonding between water molecules)[2]. Given the temperature predictions of a model, $\hat{Y}[d,t]$, at depth, d, and time-step, t, we can use Equation 3.11 to compute the corresponding density prediction, $\hat{\rho}[d,t]$.

3.2.2 Density–Depth Relationship: The density of water monotonically increases with depth as shown

---

[2]This simple fact is responsible for the sustenance of all forms of aquatic life on our planet, as water at 4°C moves down to the bottom and stops the freezing of lakes and oceans.

in the example plot of Figure 4(b), since denser water is heavier and goes down to the bottom of the lake. Formally, the density of water at two different depths, d_1 and d_2, on the same time-step, t, are related to each other in the following manner:

(3.12) $\rho[d_1, t] - \rho[d_2, t] \leq 0$ if $d_1 < d_2$.

To ensure that this physics-based equation is upheld in the temperature predictions of a physics-based model, $\hat{Y}$, we can construct a physics-based loss function as follows. Let us consider an unlabeled data set of input features on a regular grid of n_d depth values and n_t time-steps. On any pair of consecutive depth values, d_i and d_{i+1} ($d_i < d_{i+1}$), we can compute the difference in the density estimates of a model on time-step t as

(3.13) $\Delta[i, t] = \hat{\rho}[d_i, t] - \hat{\rho}[d_{i+1}, t]$

A positive value of $\Delta[i, t]$ can be viewed as a violation of the physics-based equation 3.12 on depth d_i and time t. This can be evaluated as a non-zero occurrence of $\text{ReLU}(\Delta[d_i, t])$. Hence, we can consider the mean of all physical violations across every consecutive depth-pair and time-step as our physics-based loss function:
(3.14)
$$PHY.Loss(\hat{Y}) = \frac{1}{n_t(n_d - 1)} \sum_{t=1}^{n_t} \sum_{i=1}^{n_d - 1} \text{ReLU}(\Delta[i, t]).$$

Using this physics-based loss along with the empirical loss (Equation 3.9) and regularization terms (Equation 3.10) in the learning objective (Equation 2.4), we obtain our complete PGNN formulation. Note that in our particular problem of lake temperature modeling, even though the neural network is being trained to improve its accuracy on the task of predicting water temperatures, the use of physics-based loss function ensures that the temperature predictions also translate to consistent relationships between other physical variables, namely density and depth, thus resulting in a wholesome solution to the physical problem.

4 Evaluation

4.1 Data We consider two example lakes to demonstrate the effectiveness of our PGNN framework for lake temperature modeling, lake Mille Lacs in Minnesota, USA, and lake Mendota in Wisconsin, USA. Both these lakes are reasonably large (536 km^2 and 40 km^2 in area, respectively) and show sufficient dynamics in the temperature profiles across depth over time, making them interesting test cases for analyses. Observations of lake temperature were collated from a variety of sources including Minnesota Department of Natural Resources and a web resource that collates data from federal and

	Input Drivers
1	Day of Year (1 – 366)
2	Depth (in m)
3	Short-wave Radiation (in W/m^2)
4	Long-wave Radiation (in W/m^2)
5	Air Temperature (in $°C$)
6	Relative Humidity (0 – 100 %)
7	Wind Speed (in m/s)
8	Rain (in cm)
9	Growing Degree Days [14]
10	Is Freezing (True or False)
11	Is Snowing (True or False)

Table 1: Input drivers for lake temperature modeling.

state agencies, academic monitoring campaigns, and citizen data [16]. These temperature observations vary in their distribution across depths and time, with some years and seasons being heavily sampled, while other time periods having little to no observations.

The overall data for lake Mille Lacs comprised of 7,072 temperature observations from 17 June 1981 to 01 Jan 2016, and the overall data for Mendota comprised of 13,543 temperature observations from 30 April 1980 to 02 Nov 2015. For each observation, we used a set of 11 meteorological drivers as input variables, listed in Table 1. While many of these drivers were directly measured, we also used some domain-recommended ways of constructing derived features such as Growing Degree Days [14]. We used the General Lake Model (GLM) [7] as the physics-based approach for modeling lake temperature in our experimental studies. The GLM uses the drivers listed in Table 1 as input parameters and balances the energy and water budget of lakes or reservoirs on a daily or sub-daily timestep. It performs a 1D modeling (along depth) of a variety of lake variables (including water temperature) using a vertical Lagrangian layer scheme.

Apart from the labeled set of data instances where we have observations of temperature, we also considered a large set of unlabeled instances (where we do not have temperature observations) on a regular grid of depth values at discrete steps of 0.5m, and on a daily timescale from 02 April 1980 to 01 Jan 2016 (amounting to 13,058 dates). We ran the GLM model on the unlabeled instances to produce Y_{PHY} along with the input drivers **D** at every unlabeled instance. Ignoring instances with missing values, this amounted to a total of 299,796 unlabeled instances in lake Mille Lacs and 662,781 unlabeled instances in lake Mendota.

4.2 Experimental Design We considered contiguous windows of time to partition the labeled data set into training and test splits, to ensure that the test set is indeed independent of the training set and the two data sets are not temporally auto-correlated. In particular, we chose the center portion of the overall time duration for testing, while the remainder time periods on both ends were used for training. For example, to construct a training set of n instances, we chose the median date in the overall data and kept on adding dates on both sides of this date for testing, till the number of observations in the remainder time periods became less than or equal to n. Using this protocol, we constructed training sets of size $n = 3000$ for both lake Mille Lacs and lake Mendota, which were used for calibrating the physics-based model, PHY, on both lakes. We used the entire set of unlabeled instances for evaluating the physics-based loss function on every lake.

All neural network models used in this paper were implemented using the Keras package [2] using Tensorflow backend. We used the AdaDelta algorithm [20] for performing stochastic gradient descent on the model parameters of the neural network. We used a batch size of 1000 with maximum number of epochs equal to 10,000. To avoid over-fitting, we employed an early stopping procedure using 10% of the training data for validation, where the value of patience was kept equal to 500. We also performed gradient clipping (for gradients with L_2 norm greater than 1) to avoid the problem of exploding gradients common in regression problems (since the value of Y is unbounded). We standardized each dimension of the input attributes to have 0 mean and 1 standard deviation, and applied the same transformation on the test set. The fully-connected neural network architecture comprised of 3 hidden layers, each with 12 hidden nodes. The value of hyper-parameters λ_1 and λ_2 (corresponding to the L_1 and L_2 norms of network weights, respectively) were kept equal to 1 in all experiments conducted in the paper, to demonstrate that no special tuning of hyper-parameters was performed for any specific problem. The value of the hyper-parameter λ_{PHY} corresponding to the physics-based loss function was kept equal to $std(Y^2)/std(\rho)$, to factor in the differences in the scales of the physics-based loss function and the mean squared error loss function. We used uniformly random initialization of neural network weights from 0 to 1. Hence, in all our experiments, we report the mean and standard deviation of evaluation metrics of every neural network method over 50 runs, each run involving a different random initialization.
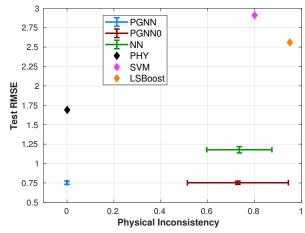
4.3 Baseline Methods and Evaluation Metrics
We compared the results of PGNN with the following baseline methods:

- **PHY**: The GLM models calibrated on the training

sets of size $n = 3000$ for both lakes were used as the physics-based models, PHY.
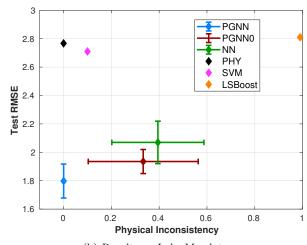
- **Black-box Models**: In order to demonstrate the value in incorporating the knowledge of physics with data science models, we consider three standard non-linear regression models: support vector machine (**SVM**) with radial basis function (RBF) kernel, least squares boosted regression trees (**LSBoost**), and the neural network (**NN**) model. All of these models were trained to predict temperature using the same set of input drivers as PGNN, but without using any knowledge of physics (either in the form of model simulations or as physics-based loss functions).

- **PGNN0**: In order to understand the contribution of the physics-based loss function in PGNN, we consider an intermediate product of our framework, PGNN0, as another baseline, which uses the hybrid-physics-data modeling setup described in Figure 2, but does not use the physics-based loss function in its learning objective (Equation 2.1). Hence, PGNN0 differs from black-box models in its use of physics-based model simulations as input attributes, and differs from PGNN in its use of a purely data-driven learning objective.

We considered the following evaluation metrics for comparing the performance of different algorithms:

- **RMSE**: We use the root mean squared error (RMSE) of a model on the test set as an estimate of its generalization performance. The units of this metric are in °C.

- **Physical Inconsistency**: Apart from ensuring generalizability, a key contribution of PGNN is to ensure the learning of physically consistent model predictions. Hence, apart from computing the RMSE of the model on the test set, we also compute the fraction of time-steps where the model makes physically inconsistent predictions (i.e., the density-depth relationship stated in Equation 3.12 is violated). We report this fraction as the physical inconsistency measure. Note that this measure does not require actual observations, and hence, we compute this measure over the plentifully large unlabeled data set.

4.4 Results Figure 5 provides a summary of the performance of different methods for modeling lake temperature on the two example lakes, Mille Lacs and Mendota. The X-axis in these plots represents the physical inconsistency of a model, while the Y-axis represents the RMSE of the model predictions

(a) Results on Lake Mille Lacs

(b) Results on Lake Mendota

Figure 5: Scatter plots showing test RMSE values (Y-axis) and physical inconsistency (X-axis) of comparative methods.

w.r.t. observations on the test set. We also show the standard deviation around the evaluation metrics of neural network-based methods (i.e., PGNN, PGNN0, and NN), since we used random initialization of network weights for every one of the 50 runs.
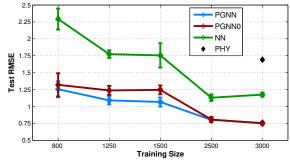
For lake Mille Lacs, we can see from Figure 5(a) that the test RMSE of the physics-based model, PHY, is 1.69. If we use black-box data science models such as SVM and LSBoost, that try to learn non-linear relationships between drivers and temperature directly without using physics, we would end up with a test RMSE that is even higher than that of PHY. Further, they also show high physical inconsistency in their model predictions (greater than 0.8). If we instead use a black-box NN model that learns non-linear compositions of features from the space of input drivers, we can achieve a test RMSE of 1.18 that is significantly

lower than that of PHY. This provides evidence of the information contained in the driver data, which if used effectively, can help in closing the knowledge gaps of PHY. However, this improvement in RMSE comes at the cost of a large value of physical inconsistency in the model predictions of NN (almost 73% of the time-steps have inconsistent density-depth relationships in its predictions). This makes NN unfit for use in the process of scientific discovery, because although it is able to somewhat improve the predictions of the target variable (i.e. temperature), it is incurring large errors in capturing the physical relationships of temperature with other variables, leading to non-meaningful results.
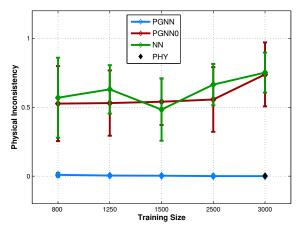
If we use the output of the physics-based model along with the drivers as inputs in the PGNN0 model, we can achieve an even lower value of test RMSE than that of NN. This is because the output of PHY (although with a high RMSE) contains vital physical information about the dynamics of lake temperature, which when coupled with powerful data science frameworks such as neural networks, can result in major improvements in RMSE. However, the results of PGNN0 are still physically inconsistent for roughly 72% of the time. In contrast, it is only by the use of physics-based loss functions in PGNN that we can not only achieve an RMSE of 0.73, but also substantially lower value of physical inconsistency (close to 0). To appreciate the significance of a drop in RMSE of $0.96°C$, note that a lake-specific calibration approach that produced a median RMSE of $1.47°C$ over 28 lakes is considered to be the state-of-the-art in the field [3]. By being accurate as well as physically consistent, PGNN provides an opportunity to produce physically meaningful analyses of lake temperature dynamics that can be used in subsequent scientific studies.

A similar summary of results can also be obtained from Figure 5(b) for lake Mendota. We can see that the test RMSE of the physics-based model in this lake is 2.77, which is considerably higher than that of Mille Lacs. This shows the complex nature of temperature dynamics in Mendota which is ineffectively being captured by PHY. The average test RMSE scores of NN and PGNN0 on this lake are 2.07 and 1.93, respectively. On the other hand, PGNN is able to achieve an average RMSE of 1.79, while being physically consistent. This is a demonstration of the added value of using physical consistency in the learning objective of data science models for improving generalization performance.

4.4.1 Effect of Varying Training Size We next demonstrate the effect of varying the size of the training set on the performance of PGNN, in comparison with

(a) Effect on Test RMSE

(b) Effect on Physical Inconsistency

Figure 6: Effect of varying training size on the performance of different methods on lake Mille Lacs.

other baseline methods. Figure 6 shows the variations in the test RMSE and physical inconsistency of different methods on lake Mille Lacs, as we vary the training size from 3000 to 800. We can see from Figure 6(a) that the test RMSE values of all data science methods increase as we reduce the training size. For example, the test RMSE of the black-box model, NN, can be seen to over-shoot the test RMSE of the physics-based model for training sizes smaller than 1500. On the other hand, both PGNN and PGNN0 show a more gradual increase in their test RMSE values on reducing training size. In fact, the PGNN can be seen to provide smaller RMSE values than all baseline methods, especially at training sizes of 1250 and 1500. This is because the use of physics-based loss function ensures that the learned PGNN model is consistent with our knowledge of physics and thus is not spurious. Such a model thus stands a better chance at capturing generalizable patterns and avoiding the phenomena of over-fitting, even after being trained with limited number of training samples. If we further reduce the training size to 800, the results

of PGNN and PGNN0 become similar because there is not much information left in the data that can provide improvements in RMSE.

While the lower RMSE values of PGNN is promising, the biggest gains in using PGNN arise from its drastically lower values of physical inconsistency as compared to other data science methods, as shown in Figure 6(b), even when the training sizes are small. Note that the results of PGNN are physically consistent across all time-steps, while PGNN0 and NN violate the density-depth relationship more than 50% of time-steps on an average. We can also see that PHY has an almost zero value of physical inconsistency, since it is inherently designed to be physically consistent.

4.4.2   Analysis of Results To provide a deeper insight into the results produced by competing methods, we analyze the predictions of lake temperature produced by a model as follows. As described previously, any estimate of temperature can be converted to its corresponding density estimate using the physical relationship between temperature and density represented in Equation 3.11. Hence, on any given time-step, we can produce a profile of density estimates at varying values of depth for every model, and match it with the density estimates of observed temperature on test instances. Visualizing such density profiles can help us understand the variations in model predictions across depth, in relationship to test observations. Some examples of density profiles on different dates in lake Mille Lacs and Mendota are provided in Figure 7, where the X-axis represents estimated density, and the Y-axis represents depth.

In the density profiles of different algorithms on lake Mille Lacs in Figure 7(a), we can see that the density estimates of PHY are removed from the actual observations by a certain amount, indicating a bias in the physics-based model. All three data science methods, NN, PGNN0, and PGNN, attempt to compensate for this bias by shifting their density profiles closer to the actual observations. On the three depth values where we have observations, we can see that both PGNN and PGNN0 show lower discrepancy with observations as compared to PHY. In fact, the density profile of PGNN matches almost perfectly with the observations, thus demonstrating the value in using physics-based loss function for better generalizability. However, the most striking insight from Figure 7(a) is that although the density estimate of PGNN0 is reasonably close to the three observations (thus indicating a low value of test RMSE), the density estimates soon start showing physically inconsistent patterns as we move lower in depth beyond the observations. In particular, the density estimates of PGNN0 start decreasing as we increase the

depth beyond 6m. This is a violation of the monotonic relationship between density and depth as illustrated in Figure 4(b). The presence of such physical inconsistencies reduces the usefulness of a model's predictions in scientific analyses, even if the model shows low test RMSE. In contrast, the predictions of PGNN, while being closer to the actual observations, are always consistent with the monotonic relationship between density and depth.

Figure 7(b) shows another example of density profiles on a different date in lake Mendota. We can see that PGNN is again able to improve upon PHY and produce density estimates that are closest to the observations. On the other hand, both PGNN0 and NN shows large discrepancies with respect to the actual observations. This is because of the complex nature of relationships between the drivers and the temperature in lake Mendota that are difficult to be captured without the use of physical relationships in the learning of neural networks. Additionally, the model predictions of PGNN0 can be seen to violate the physical relationship between density and depth (density estimates of PGNN0 decrease as we increase the depth from 10m to 12m), thus further reducing our confidence in PGNN0 representing physically meaningful results.

5   Conclusions and Future Work

This paper presented a novel framework for learning physics-guided neural networks (PGNN), by using the outputs of physics-based model simulations as well as by leveraging physics-based loss functions to guide the learning of neural networks to physically consistent solutions. By anchoring neural network methods with scientific knowledge, we are able to show that the proposed framework not only shows better generalizability, but also produces physically meaningful results in comparison to black-box data science methods.

We anticipate this paper to be a stepping stone in the broader theme of research on using physics-based learning objectives in the training of data science models. While the specific formulation of PGNN explored in this paper was developed for the example problem of modeling lake temperature, similar developments could be explored in a number of other scientific and engineering disciplines where known forms of physical relationships can be exploited as physics-based loss functions. This paper paves the way towards learning neural networks by not only improving their ability to solve a given task, but also being cognizant of the physical relationships of the model outputs with other tasks, thus producing a more holistic view of the physical problem.
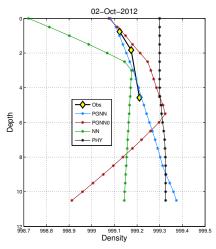
There are a number of directions of future research that can be explored as a continuation of this work.

First, for the specific problem of lake temperature modeling, given the spatial and temporal nature of the problem domain, a natural extension would be to exploit the spatial and temporal dependencies in the test instances, e.g., by using recurrent neural network based architectures. Second, the analysis of the physically consistent model predictions produced by PGNN could be used to investigate the modeling deficiencies of the baseline physics-based model in detail. Third, while this paper presented a simple way of constructing hybrid-physics-data (HPD) models where Y_{PHY} was ingested as an input in the data science model, more complex ways of constructing HPD models where the physics-based and data science components are tightly coupled need to be explored. Fourth, theoretical analyses studying the impact of introducing physics-based loss functions on the sample complexity or convergence guarantees need to be investigated. Fifth, the research direction of PGNN can be complemented with other related efforts on producing interpretable data science results. In particular, the use of physics-based equations for interpreting the results of data science methods needs to be explored. Finally, while this paper explored the use of physical relationships between temperature, density, and depth of water in the learning of multi-layer perceptrons, other forms of physical relationships in different neural network models can be explored as future work. Of particular value would be to develop generative models that are trained to not only capture the structure in the unlabeled data, but are also guided by physics-based models to discover and emulate the known laws of physics. The paradigm of PGNN, if effectively utilized, could help in combining the strengths of physics-based and data science models, and opening a novel era of scientific discovery based on both physics and data.
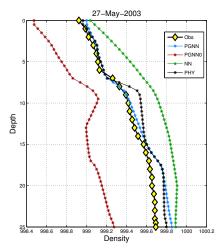
References

[1] T. Appenzeller. The scientists' apprentice. *Science*, 357(6346):16–17, 2017.

[2] F. Chollet. keras. `https://github.com/fchollet/keras`, 2015.

[3] X. Fang, S. R. Alam, H. G. Stefan, L. Jiang, P. C. Jacobson, and D. L. Pereira. Simulations of water quality and oxythermal cisco habitat in minnesota lakes under past and future climate scenarios. *Water Quality Research Journal*, 47(3-4):375–388, 2012.

[4] D. Graham-Rowe, D. Goldston, C. Doctorow, M. Waldrop, C. Lynch, F. Frankel, R. Reid, S. Nelson, D. Howe, S. Rhee, et al. Big data: science in the petabyte era. *Nature*, 455(7209):8–9, 2008.

[5] H. V. Gupta and G. S. Nearing. Debatesthe future of hydrological sciences: A (common) path forward? using models and data to learn: A systems theoretic perspective on the future of hydrological science. *Water Resources Research*, 50(6):5351–5359, 2014.

[6] T. D. Harris and J. L. Graham. Predicting cyanobacterial abundance, microcystin, and geosmin in a eutrophic drinking-water reservoir using a 14-year dataset. *Lake and Reservoir Management*, 33(1):32–48, 2017.

[7] M. Hipsey, L. Bruce, and D. Hamilton. Glmgeneral lake model: Model overview and user information. *Perth (Australia): University of Western Australia Technical Manual*, 2014.

[8] T. Jonathan, A. Gerald, et al. Special issue: dealing with data. *Science*, 331(6018):639–806, 2011.

[9] U. Lall. Debatesthe future of hydrological sciences: A (common) path forward? one water. one world. many climes. many souls. *Water Resources Research*, 50(6):5335–5341, 2014.

[10] J. J. Magnuson, L. B. Crowder, and P. A. Medvick. Temperature as an ecological resource. *American Zoologist*, 19(1):331–343, 1979.

[11] J. L. Martin and S. C. McCutcheon. *Hydrodynamics and transport for water quality modeling*. CRC Press, 1998.

[12] J. J. McDonnell and K. Beven. Debatesthe future of hydrological sciences: A (common) path forward? a call to action aimed at understanding velocities, celerities and residence time distributions of the headwater hydrograph. *Water Resources Research*, 50(6):5342–5350, 2014.

[13] H. W. Paerl and J. Huisman. Blooms like it hot. *Science*, 320(5872):57–58, 2008.

[14] I. C. Prentice, W. Cramer, S. P. Harrison, R. Leemans, R. A. Monserud, and A. M. Solomon. Special paper: a global biome model based on plant physiology and dominance, soil properties and climate. *Journal of biogeography*, pages 117–134, 1992.

[15] F. J. Rahel and J. D. Olden. Assessing the effects of climate change on aquatic invasive species. *Conservation biology*, 22(3):521–533, 2008.

[16] E. K. Read, L. Carr, L. De Cicco, H. A. Dugan, P. C. Hanson, J. A. Hart, J. Kreft, J. S. Read, and L. A. Winslow. Water quality data for national-scale aquatic research: The water quality portal. *Water Resources Research*, 53(2):1735–1745, 2017.

[17] J. J. Roberts, K. D. Fausch, D. P. Peterson, and M. B. Hooten. Fragmentation and thermal risks from climate change interact to affect persistence of native trout in the colorado river basin. *Global Change Biology*, 19(5):1383–1398, 2013.

[18] J. J. Roberts, K. D. Fausch, M. B. Hooten, and

D. P. Peterson. Nonnative trout invasions combined with climate change threaten persistence of isolated cutthroat trout populations in the southern rocky mountains. *North American Journal of Fisheries Management*, 37(2):314–325, 2017.

[19] T. J. Sejnowski, P. S. Churchland, and J. A. Movshon. Putting big data to good use in neuroscience. *Nature neuroscience*, 17(11):1440–1441, 2014.

[20] M. D. Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

(a) Lake Mille Lacs on 02-October-2012

(b) Lake Mendota on 27-May-2003

Figure 7: Density profiles of varying algorithms on different dates in lake Mille Lacs (Figure 7(a)) and Mendota (Figure 7(b)).