Department of Mathematical Sciences
TMA4500 - Industrial Mathematics, Specialization Project

Physics Guided Neural Network-assisted Corrective Source Term Approach to Hybrid Analysis and Modeling

Sondre Sørbø

Supervisors:
Professor Trond Kvamsdal
Professor Adil Rasheed

Trondheim, December 21st, 2021

NTNU
Norwegian University of
Science and Technology

Faculty of Information Technology and Electrical Engineering
DEPARTMENT OF MATHEMATICAL SCIENCES

Preface

The work presented in this report is done the fall of 2021 in the course *TMA4500 - Industrial Mathematics, Specialization Project*. The work has given interesting results, and we will try to publish an article describing these findings. Although a specialication project usually results in its own report, I have been permitted to submit my report in the form of this article, which follows after this preface.

I would like to thank my two supervisors, professor Adil Rasheed and professor Trond Kvamsdal, for their guidance throughout this project. I am looking forward to working more with them with my masters thesis, for which we will look into some of the future research directions outlined at the end of the following paper.

Physics Guided Neural Network-assisted Corrective Source Term Approach to Hybrid Analysis and Modeling

Sondre Sørbø[a], Sindre Stenen Blakseth[b], Adil Rasheed[c,*], Trond Kvamsdal[a], Omer San[d]

[a]*Department of Mathematical Sciences, Norwegian University of Science and Technology*
[b]*Department of Gas Technology, SINTEF Energy Research*
[c]*Department of Engineering Cybernetics, Norwegian University of Science and Technology*
[d]*School of Mechanical and Aerospace Engineering, Oklahoma State University*

Abstract

Hybrid Analysis and Modeling (HAM) is an emerging modeling paradigm that aims to combine physics-based models (PBM) and data-driven models (DDM) to create generalizable, trustworthy, accurate, computationally efficient and self-evolving models. In this article we put forth a modular framework by synthesizing ideas gathered from two emerging HAM methods; the Physics Guided Neural Network (PGNN), and Corrective Source Term Approach (CoSTA). This approach, which we abbreviate PGNN-CoSTA, utilizes PGNN to learn the corrective source term in the CoSTA method. Through applications to the heat diffusion problems, we show that the introduction of PGNN architecture in the CoSTA yields a significant increase in the accuracy of the CoSTA method. Another key accomplishment of the proposed method is its ability to provide lower uncertainty in most cases tested, which is crucial in scientific machine learning applications. Due to its flexible but solid theoretical foundation, PGNN-CoSTA provides a modular framework for leveraging novel developments within both PBM and DDM, and can be a potential door-opener for data-driven techniques to enter high-stakes applications previously reserved for pure PBM.

Keywords: Physics guided neural network, Corrective Source Term Approach, Hybrid analysis and modeling, Deep Neural Network

1. Introduction

With the current trend of digitalization, fuelled by cost-effective sensors, computational infrastructure, availability of data, and breakthroughs in advanced machine learning algorithms, we see technologies like digital twins (Rasheed et al., 2020), cyberphysical systems (Liu et al., 2017) and Industry 4.0 (Ghobakhloo et al., 2021) becoming increasingly mainstream. However, one of the many prerequisites for advancing these technologies is models that are generalizable, trustworthy, computationally efficient and accurate, and dynamically self-adapting (San et al., 2021). The two most popular modeling paradigms; physics based modeling (PBM) and data-driven modeling (DDM), unfortunately do not possess all these desired characteristics on their own. PBMs are based on robust foundation and understanding of the physics, and is interpretable and generalizable, while DDMs, and in particular deep neural networks (DNN), are more computationally efficient, in many cases more accurate, and automatically recognises patterns in data. The strengths and weaknesses of each of these models are briefly discussed in sections 2.3 and 2.4.

The newly emerging modelling paradigm called Hybrid Analysis and Modeling (HAM) (San et al., 2021) combines these strengths. There are numerous techniques for integrating DNNs or other DDMs into PBMs, of which an overview can be found in Willard et al. (2020). They systematically organize aspects of hybridization regarding dowscaling (Srivastava et al., 2013), reduced order modeling (Mou et al., 2021), equation discovery (Zhang and Ma, 2020), parameterization (Brenowitz and Bretherton, 2019), predictability (Grover et al., 2015), data generation (Cheng et al., 2020) and inverse modeling (Svendsen et al., 2017). (Blakseth et al., 2022).

More recently, the corrective source term approach (CoSTA) and physics-guided neural networks (PGNNs) have emerged as

*Adil Rasheed
Email addresses: `sondsorb@stud.ntnu.no` (Sondre Sørbø), `sindre.blakseth@sintef.no` (Sindre Stenen Blakseth), `adil.rasheed@ntnu.no` (Adil Rasheed), `trond.kvamsdal@ntnu.no` (Trond Kvamsdal), `osan@okstate.edu` (Omer San)

two methods within the HAM paradigm that seem to hold huge potential for acceptance in high stake applications, where reliability is vital. CoSTA is a method proposed by Blakseth et al. (2022) that explicitly addresses the problem of unaccounted / unresolved / incorrect physics in PBM. This is done by augmenting the governing equations of a PBM that describes partial physics, with a DNN-generated corrective source term that takes into account the remaining unknown / ignored physics. Blakseth et al. (2022) demonstrated the potential of the method by applying it to one dimensional heat diffusion problem. The method was shown to outperform both PBM and DDM for interpolation as well as extrapolation scenarios. The advantage of CoSTA is that the approach to a large extent exploits the known physics, and only relies on black-box modeling as a correction. Additionally, physical laws (like the law of conservation of energy) can be used to put a sanity check on the predictions of the DNN before making use of it in the COSTA, further improving the reliability of the method.

In the PGNN, proposed by Pawar et al. (2021), a neural network is guided using partial knowledge or a PBM utilizing this partial knowledge. This works better when one does not know the exact form of the governing equations, but are able to guess the form of the most important parts of it. In order to use this information, the method addresses a flaw in the usual architecture of a neural network, namely that important features, like physical parameters or predictions from simplified PBMs, may be forgotten if injected at the input layer. By injecting these features at an appropriate intermediate layer, the PGNN is able to better utilize this kind of features. It has also been used to incorporate predictions from simplified models, in Pawar et al. (2021). The advantage of PGNN is that it helps in effectively utilizing partial knowledge to improve model accuracy and robustness, and reducing uncertainty.

It is clear that both CoSTA and PGNN each have their distinct advantages, and distinct ways of incorporating knowledge about the physics of a system. In this paper we therefore attempt to combine their strengths by using a PGNN architecture for the correction term of CoSTA. Through a series of cases involving heat diffusion problems, we demonstrate that the combined PGNN-CoSTA approach is up to several orders of magnitude more accurate than both PBM and DDM, and even the CoSTA applied to similar problems in (Blakseth et al., 2022). In most cases, the PGNN-CoSTA model has the least uncertainty, and its solutions are much smoother.

This article consists of a quick introduction of the relevant theory in Section 2, followed by all the details of the set-up of the experiments in Section 3. Results and discussions are presented in Section 4 followed by conclusions and suggestions for future work in Section 5.

2. Theory

2.1. Heat diffusion

In this article, we choose the heat diffusion problem to demonstrate the effectiveness of the PGNN-CoSTA method. The reasons for choosing the heat diffusion problem are many. Firstly, temperature measurements are being increasingly used for health / condition monitoring on a large variety of applications. Mansor et al. (2013) demonstrated how body temperature can be used for remote health monitoring while Shahidi et al. (2013) used temperature for real-time bearing condition monitoring. Secondly, many quantities that can not be directly measured in a process can be indirectly inferred from the measured temperature, and thirdly, non-intrusive, cost-effective methods exist to make temperature measurements in real-time (eg. Prajapati et al. (2020)). However, it is important to stress that the method is not limited to this chosen problem, and, as have been argued by Blakseth (2021), can be applied to large variety of equations.

The equation of heat diffusion problem can be written as

$$\frac{\partial u}{\partial t} = \Delta u + f, \tag{1}$$

where $u = u(t, x)$ is the temperature and $f = f(t, x)$ is a heat source term, and $\Delta = \nabla^2$ is the Laplace operator. The equation states that heat will flow from warm to cold areas at a rate proportional to the heat gradient, while additional heat may be introduced or extracted using the source term. We will consider the equation on a bounded spatial domain $x \in \Omega$, for a time interval $t \in [0, T]$. In order for the equation to yield a unique solution, we need an initial condition and a boundary condition,

$$u(0, x) = u_0(x) \quad \forall x \in \Omega \tag{2}$$

$$L_{\partial\Omega} u(t, x) = g(t, x) \quad \forall x \in \partial\Omega, \tag{3}$$

for some appropriate[1] differential operator $L_{\partial\Omega}$. In this article, we will use Dirichlet boundary conditions, i.e. $L_{\partial\Omega} = I$, where

[1] There has to be Dirichlet condition on at least some part of $\partial\Omega$, or constant terms can be added to the solution without violating the heat diffusion equation, thus making the solution non-unique.

I is the identity operator.

2.2. Method of manufactured solutions

We wish to evaluate the performance of the models in a variety of different scenarios. To this end we use the method of manufactured solutions (Roache, 2001), which involves choosing a solution u, and calculating the source f from equation (1), before trying to reproduce the solution using f, $u|_{t=0}$ and $u|_{\Omega}$. This approach enables comparison with the known exact solution to accurately quantify the error of the model, as opposed to resorting to numerical solutions of the equation which might have their own limitations.

2.3. Physics based model

This approach involves careful observation of a physical phenomenon of interest (heat diffusion in the current work), development of its partial understanding, expression of the understanding in the form of mathematical equations (based on the law of conservation of energy) and ultimately solution of these equations. Due to the partial understanding and numerous assumptions along the steps from observation to solution of the equations, a large portion of the important governing physics gets ignored. Even the applicability of high fidelity simulators with minimal assumptions has so far been limited to the offline design phase only. Despite this major drawback, what makes these models attractive are sound foundations from first principles, interpretability, generalizability and existence of robust theories for the analysis of stability and uncertainty. However, most of these models are generally computationally expensive, do not adapt to new scenarios automatically and can be susceptible to numerical instabilities. The PBM used in this article is a discretization of the heat diffusion equation (1) using the finite element method (FEM) along the spatial dimension, and the backward Euler method along the temporal dimension. For an introduction to FEM, see Quarteroni (2017) or Brenner and Scott (2008). We will now derive the discretized equation, first for equation (1), then including the boundary conditions (3).

Let k be some constant time step length, and define the time steps $t_i = ik$ for $i \in \{0, 1, ...\}$. Let $u_i(\cdot) \approx u(t_i, \cdot)$ be the numeric approximation of the solution at time t_i. We approximate the time derivative by $\frac{\partial u}{\partial t}(t_i) \approx \frac{u_i - u_{i-1}}{k}$ and get the semi-discretized heat equation

$$u_i - k\Delta u_i = u_{i-1} + f(t_i, \cdot). \tag{4}$$

We require the functions u_i to have integrable first derivatives, and consider the solution space $X = H^1$. Integrating over the spacial domain Ω and multiplying with a test function $v \in X_0 = \{v \in X : v|_{\partial\Omega} = 0\}$, we get

$$\int_{\Omega} (u_i v - k_i v \Delta u) dx = \int_{\Omega} v(u_{i-1} + f(t_i, \cdot)) dx, \tag{5}$$

or equivalently, since $v|_{\partial\Omega} = 0$,

$$\int_{\Omega} (u_i v + k_i \nabla v \nabla u) dx = \int_{\Omega} v(u_{i-1} + f(t_i, \cdot)) dx. \tag{6}$$

We now consider the above equation with the functions in a finite dimensional subspace $X^h \subset X$ spanned by the basis of functions $\{\phi_j\}_j$. This means we have $u_i = \sum_j u_{i,j}\phi_j$, $v = \sum_j v_j \phi_j$, and $f(t_i, x) = \sum_j f_{i,j}\phi_j(x)$, for some sets of scalars $\{u_{i,j}\}_j, \{v_j\}_j, \{f_{i,j}\}_j$.

We can then rewrite the above equation into $v^T(M + kA)u_i = v^T M u_{i-1} + v^T F_i$ where M is the matrix with elements $m_{j,l} = \int_{\Omega} \phi_j \phi_l dx$, A the matrix with elements $a_{j,l} = \int_{\Omega} \nabla\phi_j \nabla\phi_l dx$ and F_i the vector with elements $\int_{\Omega} \phi_j f(t_i, \cdot) dx$. This should hold for all vectors v in the euclidean space of appropriate dimension, hence we can simplify to the equation

$$(M + kA)u_i = M u_{i-1} + F_i. \tag{7}$$

Ensuring the boundary conditions hold corresponds to requiring $\int_{\partial\Omega} \sum_j u_{i,j}\phi_j(x)\phi_k(x) dx = \int_{\partial\Omega} g(t_i, x)\phi_k(x) dx$, or $Bu = G_i$ where $B = (\int_{\partial\Omega} \phi_j(x)\phi_k(x) dx)_{j,k}$ and the j-th element of the vector G_i is $\int_{\partial\Omega} g(t_i, x)\phi_k(x) dx$. We add this to the discretized equation (7) after multiplying with a big number $1/\epsilon$ to ensure it dominates the equations describing the boundary values of u, and we end up with

$$(M + kA + B/\epsilon)u_i = M u_{i-1} + F_i + G_i/\epsilon. \tag{8}$$

This system of linear equations can be solved by any appropriate solver, preferably one for sparse matrices if the basis allows it. Starting from the known initial condition u_0 projected onto X^h, this enables us to calculate an approximation of $u(x, t)$ step by step. The results in this article are found using piecewise linear Lagrange elements (Courant, 1943) on a equidistant grid. As long as the grid is fine enough, and the equation, the boundary and initial conditions and the source term is known, this gives a very accurate solution. However, we will see that when parts of the system, like the source term, is unknown, the solution might be quite far from the correct one.

2.4. Data driven model

Data driven modeling has become very popular in recent years, following the increased availability of large amounts of data and computational resources as well as free, state of the art machine learning libraries. Contrary to PBM, DDM learns from data, and assuming the data provided is of sufficient quantity as well as quality, it will learn the full physics of the system it describes, independent of our knowledge and understanding of it. Some of the advantages of this approach is highly computationally efficient predictions, ability to adapt to changes on the fly, as well as high accuracy even for difficult problems, when sufficient data is available. Nevertheless, they are not widely used in high stake situations due to weaknesses like poor generalizability, lack of explainability and theory of error analysis and stability, and the need for large amounts of data. Although there is a large variety of DDM available we use deep neural networks in the current article as they are universal approximators and hence have the ability to model highly complex non-linear phenomena. (Blakseth et al., 2022). A DNN with n layers is a function $D : \mathbb{R}^{d_0} \to \mathbb{R}^{d_n}$ defined by

$$D(x) = T_n \circ \sigma_{n-1} \circ T_{n-1} \circ \sigma_{n-2} \cdots \circ \sigma_1 \circ T_1(x) \tag{9}$$

where $T_i : \mathbb{R}^{d_{i-1}} \to \mathbb{R}^{d_i}$ are affine transformations and $\sigma_i : \mathbb{R}^{d_i} \to \mathbb{R}^{d_i}$ are some preferably non-linear activation functions. Each transformation T_i is determined by a weight matrix and a bias vector, a total of $d_{i-1} \cdot d_i + d_i$ values. These are tuned to minimize the error on the training data.

DNNs are widely used, due to their simplicity and yet astonishing performance in many situations. They can be applied to a large variety of problems, and have delivered impressive achievements across numerous fields of research and applications. Meanwhile, they also have some inherent weaknesses. They are tuned for performing on the training data. While there are many ways of preventing the network from being too specialised (Nusrat and Jang, 2018), they will not be good at extrapolation cases where the prediction task is somehow qualitatively different from the training tasks. In addition, they are not easily explainable or predictable, meaning its hard to explain what kind of patters the network will look for, and unexpected predictions can occur. Compared to PBMs, DDMs usually make predictions much faster, but may need long training times for optimal performance.

2.5. Hybrid Analysis and Modeling

As mentioned earlier in section 1, the HAM paradigm eliminates the weaknesses of both the PBM and DDM. In the following we briefly describe the two HAM approaches used in this work:

2.5.1. Corrective source term approach

In this section we present a brief justification of the use of CoSTA. It is based on the more thorough argument that can be found in Blakseth et al. (2022).

Consider the differential equation

$$\begin{aligned} L_\Omega u &= f, \quad \forall\, x \in \Omega \\ L_{\partial\Omega} u &= g, \quad \forall\, x \in \partial\Omega, \end{aligned} \tag{10}$$

where $L_\Omega, L_{\partial\Omega}$ are differential operators, f is a source term and g a function specifying the boundary condition. Now let $\tilde{u}$ be the solution to the perturbed problem

$$\begin{aligned} \tilde{L}_\Omega \tilde{u} &= \tilde{f}, \quad \forall\, x \in \Omega \\ \tilde{L}_{\partial\Omega} \tilde{u} &= \tilde{g} \quad \forall\, x \in \partial\Omega, \end{aligned} \tag{11}$$

where the perturbations $\tilde{\cdot}$ are due to imperfections such as unknown physics, modelling errors, discretization error or inaccurate data. Assume we can calculate the residuals

$$\begin{aligned} \sigma_\Omega &= \tilde{L}_\Omega (u - \tilde{u}) \\ \sigma_{\partial\Omega} &= \tilde{L}_{\partial\Omega} (u - \tilde{u}), \end{aligned} \tag{12}$$

and let $\hat{\tilde{u}}$ be the solution of the corrected, perturbed problem

$$\begin{aligned} \tilde{L}_\Omega \hat{\tilde{u}} &= \tilde{f} + \sigma_\Omega, \quad \forall\, x \in \Omega \\ \tilde{L}_{\partial\Omega} \hat{\tilde{u}} &= \tilde{g} + \sigma_{\partial\Omega} \quad \forall\, x \in \partial\Omega. \end{aligned} \tag{13}$$

Using the definition of the residuals, as well as the perturbed equation (11), we see that

$$\begin{aligned} \tilde{L}_\Omega \hat{\tilde{u}} &= \tilde{f} + \tilde{L}_\Omega (u - \tilde{u}) = \tilde{L}_\Omega u, \quad \forall\, x \in \Omega \\ \tilde{L}_{\partial\Omega} \hat{\tilde{u}} &= \tilde{g} + \tilde{L}_{\partial\Omega} (u - \tilde{u}) = \tilde{L}_{\partial\Omega} u \quad \forall\, x \in \partial\Omega, \end{aligned}$$

which reduces to $\hat{\tilde{u}} - u$ if the corrected, perturbed problem (13) yields a unique solution. From this argument, we see that the source term corrections are able to compensate for perturbations in the differential operators as well as the source terms.

In real scenarios, we obviously cannot calculate the residual exactly, as that would require the solution we are trying to estimate. The idea of the CoSTA method is to use a DDM to estimate the residual. For the input of the DDM we use uncorrected solution $\tilde{u}$. This means the PBM is used twice, first to solve (11) for $\tilde{u}$, then (13) for $\hat{\tilde{u}}$. Since the DDM output is only a correction, the CoSTA framework exploits the knowledge in the physical model, unlike a pure DDM approach.

For the discretized version of heat equation derived in section 2.3, the corrected equation is

$$(M + kA + B/\epsilon)u_i = Mu_{i-1} + F_i + G_i/\epsilon + \sigma. \tag{14}$$

As the boundary values are known, and ensured by the ϵ^{-1}-terms, we do not need any correction for these elements. Therefore the DDM only need to output values for the inner nodes.

In combining a PBM and a DDM, CoSTA aims to combine the strengths of both of these. While this works well, it is clear that it also inherits weaknesses from both. It requires both a physical description of the system it models, as well as empirical data. It needs to be trained before it can be used, but it also uses much computational resources to make prediction, as two steps must be made with the PBM for every step with the CoSTA, in addition to the DDM prediction. Also, if a regular DNN is used, it is difficult to include extra parameters into the input in a satisfactory way. This is what we aim to address in this article.

2.5.2. *Physics guided neural network*

Long vectors with function values at grid points normally contain a lot of quite redundant information. This can contribute to overfitting a DDM. To avoid this, one can reduce the dimensionality of the vector before inputting it, e.g., by using principal component analysis (PCA), as used in principal component regression (PCR). When using a DNN, an alternative approach is to let the dimentionality reduction be a part of the neural network by including a low dimensional layer (i.e., a small d_i), henceforth referred to as a bottleneck layer. Contrary to PCA, this allows for nonlinear dimensionality reduction, that is trained together with the prediction part of the network for optimal performance.

Introducing additional parameters or terms relevant for the prediction in the input layer is often not very beneficial, as these parameters are of another nature, and often much fewer, than the function values. A better strategy can be to concatenate them with the lower dimensional vector in the bottleneck layer, as shown in Pawar et al. (2021).

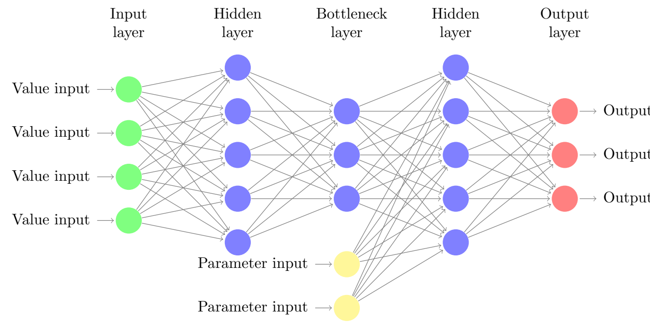


Figure 1: A visualization of an example PGNN architecture, which differs from the usual DNN architecture by having two input layers. The first input layer are typically function values, while the second one may contain parameters or predictions from simple PBMs.

Figure 1 shows a PGNN architecture. In the format of equation (9), a PGNN will take the form

$$D(x, y) = T_n \circ \sigma_{n-1} \circ \cdots T_i(C(\cdot, y)) \circ \sigma_{i-1} \circ \cdots \sigma_1 \circ T_1(x) \tag{15}$$

where T_i is the layer after the bottleneck layer, $C(x, y)$ is the concatenation function, and y is the extra input vector.

3. Set-up and experiments

In this section we present the details of the models we used and the problems we tested them on. The design of the experiments, including choice of problems, is very similar to that in Blakseth et al. (2022), for easy comparison and validation of the results. We will first describe how the data was generated, then the architectural design of the models.

3.1. *Data generation*

The manufactured solutions used in this project are presented in table 1. They cover a variety of functions, with polynomial, harmonic and hyperbolic terms. In order to simulate some unknown physics, the source term f was assumed unknown and replaced with the zero function in the PBM. This means the residual which the neural networks in the CoSTA methods would try to approximate, would be (at least approximately) the source term f.

In all tests the spatial domain was $x \in [0, 1]$, with 21 grid points, and an equal distance of $1/20$ between any pair of neighbouring points. Piecewise linear elements were used for the FEM solver. The temporal domain was $0 \le t \le T = 5$ with $k = 1/1000$, resulting in 5000 time steps.

The neural networks were only trained and tested on one solution each, but for several values of the parameter α. Table 2

Table 1: Manufactured solutions the methods were tested on. The first ones were used for hyperparameter tuning, the rest for evaluation and comparison of the methods.

Label	$u(t, x, \alpha)$	$f(t, x, \alpha) = \frac{\partial u}{\partial t} - \Delta u$
Solutions used for tuning		
A	$\sqrt{t+\alpha+1} + 7x^2(x-1)(x+2)$	$\frac{1}{2\sqrt{t+\alpha+1}} - 84x^2 - 42x + 28$
B	$\frac{-x^3(x-\alpha)}{t+0.1}$	$-x^3(x-\alpha)(t+0.1)^2 + \frac{12x^2-6\alpha x}{t+0.1}$
Solutions used for evaluation		
1	$t + \alpha\frac{x^2}{2}$	$1-\alpha$
2	$\sqrt{1+\alpha+t} + 10x^2(x-1)(x+2)$	$\frac{1}{2\sqrt{1+\alpha+t}} - 120x^2 - 60x + 40$
3	$2 + \alpha(x-1)\tanh(\frac{x}{t+0.1})$	$\alpha\frac{x(1-x) + 2((x-1)\tanh(\frac{x}{t+0.1}) - t - 0.1)}{(t+0.1)^2\cosh(\frac{x}{t+0.1})^2}$
4	$1 + \sin(2\pi t + \alpha)\cos(2\pi x)$	$2\pi\cos(2\pi t + \alpha) + 4\pi^2\sin(2pit + \alpha)\cos(2\pi x)$

presents values of α used for training, testing and validation. For each α in the training and validation set, the exact solution was calculated for each time step, along with the PBM predictions and residuals for training the CoSTA models. Each PBM prediction was calculated using the exact solution in the preceding step as initial value. This differs from the predictions and residuals on the test set, for which the prediction for the previous step was used as input. The weights of the neural networks were adjusted so that they minimised the error in all of the single step predictions in the training set. The validation data was used to assert that the results generalised to other parameters, and the training algorithm utilised early stopping based on this data. The testing set includes both the interpolated values $0.7, 1.5$ and extrapolated values $-0.5, 2.5$. This makes it possible to simulate scenarios with different degrees of how representative the available data is.

Table 2: Sets of values of the parameter α in the solutions, used to make the training, validation and testing sets.

Set usage	Notation	Values
Testing	$\mathcal{A}_{\text{test}}$	$\{-0.5, 0.7, 1.5, 2.5\}$
Validation	$\mathcal{A}_{\text{val}}$	$\{0.8, 1.1\}$
Training	$\mathcal{A}_{\text{train}}$	$\{0.1, 0.2, ...2.0\} \setminus (\mathcal{A}_{\text{test}} \cup \mathcal{A}_{\text{val}})$

3.2. Neural network architecture

The neural networks are implemented in TensorFlow (Abadi et al., 2015), using fully connected layers and leaky ReLU (Maas et al., 2013), with coefficient 0.01 for negative inputs, as activation function. Parameters used for the training are presented in Table 3. Both the input and output of all the neural networks were normalized based on the training data. The structures of the neural networks are presented in table 4. The parameter α was injected with the bottleneck layer along with the time t. The hyperparameters of the networks were chosen based on their performance on the tuning problems given in Table 1. The networks used in DDM and CoSTA have the same structure. However, it was found that merely by injecting α and t, the complexity of the network could be considerably reduced in terms of the number of trainable parameters (22739 for DDM and CoSTA, against 6605 in PGNN-CoSTA). This is expected as the networks in DDM and CoSTA would have to discover the fact that these values have a significant impact on the temperature field during the training process, while this information was explicitly injected into an intermediate layer in the PGNN-CoSTA.

Table 3: Parameters used for the training procedures for all of the neural networks

Loss function	MSE
Optimizer	Adam (Kingma and Ba, 2014)
Learning rate	$1e-5$
Patience	20

4. Results and discussions

In this section we present and discuss the results, first for the interpolation, and then for the extrapolation cases. To quantify the model uncertainty an ensemble of four neural network with slightly different initialization of the weights were trained resulting in all the networks to converge to different minima and hence slightly different predictions on the test data. In the plots that are shown in this section, the means are plotted as solid lines. For solution plots, the intervals $\pm$ one standard deviation, calculated with one reduced degree of freedom,

Table 4: Neural network structures. All the layers are fully connected. The bottleneck layer in the PGNN is a concatenation of 5 nodes from the previous layers, along with the parameter α and time t.

DNN structure

Layer type	Input	Hidden	Hidden	Hidden	Hidden	Output
Nodes	21	80	80	80	80	19

Physics guided network structure

Layer type	Input	Hidden	Hidden	Bottleneck	Hidden	Hidden	Hidden	Output
Nodes	21	125	25	5+2	7	10	14	19

are shaded around the means. In the error plots, the areas between the means, and the means plus one added standard deviation, are shaded. The errors are quantified using the (relative) discrete l_2-semi-norm. We emphasise that this only considers the error on the grid nodes, and not in between.

4.1. Interpolation

In the interpolation scenarios, solutions 1-4 were predicted with $\alpha \in \{0.7, 1.5\}$. The relative discrete l_2 errors are plotted against time in figure 2. We observe in all solutions that the CoSTA and PGNN-CoSTA models consistently outperform both the PBM and pure DDM, in most cases by several orders of magnitude. For solutions 1, 3 and 4, the PGNN-CoSTA models have considerably lower error than than all the other methods. For solutions 1 and 4, it is several orders of magnitude lower during the entire period, while for solution 3 CoSTA is best in the initial period. For solution 2, CoSTA consistently perform slightly better than PGNN-CoSTA. For solution 1 and 3 we observe one or more dips in the error of PGNN-CoSTA. This is likely the solution going between predicting too high or too low values. The CoSTA models have quite noisy error for solutions 1 and 4, while PGNN-CoSTA in comparison has a much smoother error development.

The final solutions are plotted in figure 3. For solutions 1,2 and 4, all the CoSTA predictions are qualitatively correct. For solution 3, while both CoSTA and PGNN-CoSTA methods have correct shape and quite accurate mean, PGNN-CoSTA has much less uncertainty.

For solution 3, it seems that all predictions have the correct shape of the curve, and it is only the amplitude that is a bit tricky to get right. This fits well with the claim that the prediction is either too high or too low as suggested earlier.

4.2. Extrapolation

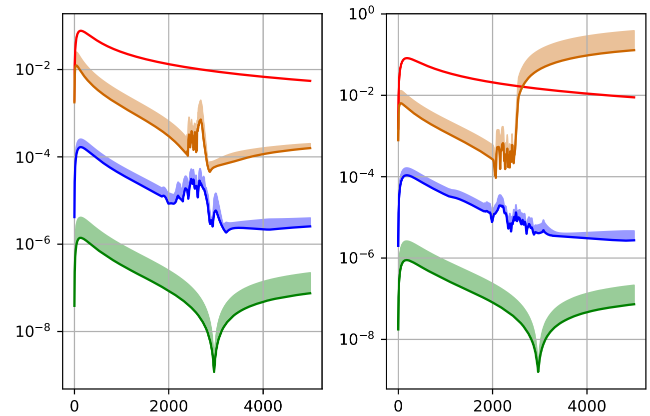
In the interpolation scenarios, solutions 1-4 were predicted with $\alpha \in \{-0.5, 2.5\}$. The relative l_2 errors are plotted against time in figure 4. While the plots seem quite similar to the interpolation ones, there are some differences. For solution 1 we observe the CoSTA error being much larger than in the interpolation scenario, while PGNN-CoSTA maintains its accuracy also for these values of α. For solution 2, PGNN-CoSTA does slightly worse than in the interpolation scenario, while the other models have quite similar error to the interpolation - except for at the very end of the $\alpha = 2.5$ case. For solution 3 we observe that both CoSTA and PGNN-CaSTA has much larger error than in the interpolation. CoSTA is worse than both FEM and DNN, while the PGNN-CoSTA is still by far the most accurate method. Solution 4 results seem unaffected by the extrapolation.

The final solutions are plotted in figure 5. For solution 3, we see that a negative α yields a qualitatively different exact solution from the training data, being positive instead of negative. This explains the increased error. We observe that only the PGNN-CoSTA predicted relatively well.

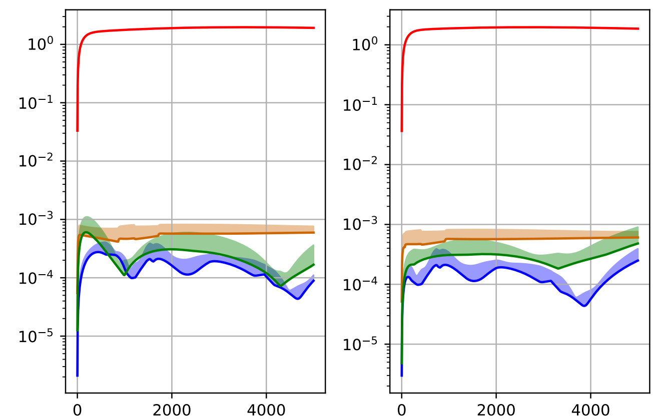
For solution 3, both in interpolation and extrapolation cases, the plots of the final solution show that the PGNN-CoSTA method has much less uncertainty than the CoSTA method. For the other solution this is not evident from the final solution plots, but looking at the error plots, and keeping in mind that they are exponentially scaled, we generally see a lower uncertainty for PGNN-CoSTA than CoSTA for solutions 1 and 4, while CoSTA seem more certain for solution 2.

For solution 1, for $\alpha = 1.5$ and $\alpha = -0.5$ we see the DNN prediction is very off, while for $\alpha = 0.7$ and $\alpha = 2.5$ it is relatively good. This shows that the quality of the DNN predictions are quite unpredictable.

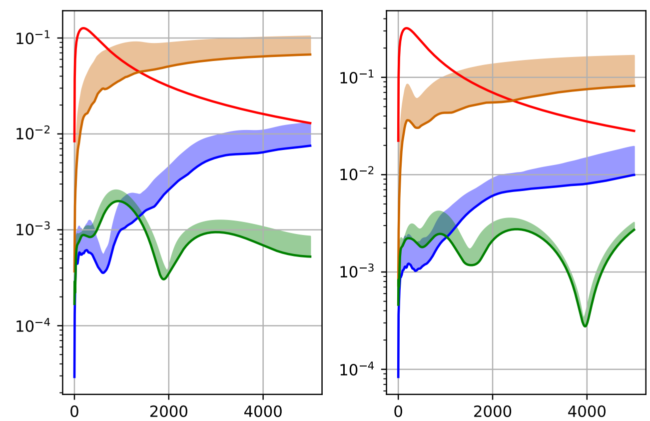
To summarize, PGNN-CoSTA is both more accurate and less uncertain than CoSTA in solution 1, 3 and 4, by a large degree, both for interpolation and extrapolation cases. For solution 2, however, CoSTA provides slightly more accurate and less uncertain results. PGNN-CoSTA also has a much smoother error in most cases, indicating that the predictions are more stable, and likely more realistic.



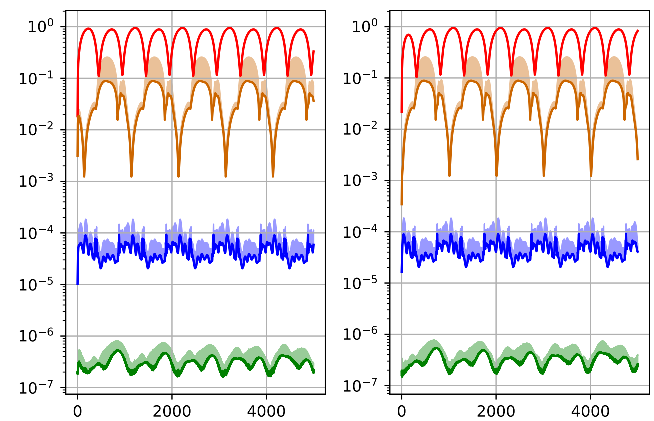
(a) Solution 1 with α = 0.7 (left) and α = 1.5 (right)



(b) Solution 2 with α = 0.7 (left) and α = 1.5 (right)



(c) Solution 3 with α = 0.7 (left) and α = 1.5 (right)



(d) Solution 4 with α = 0.7 (left) and α = 1.5 (right)

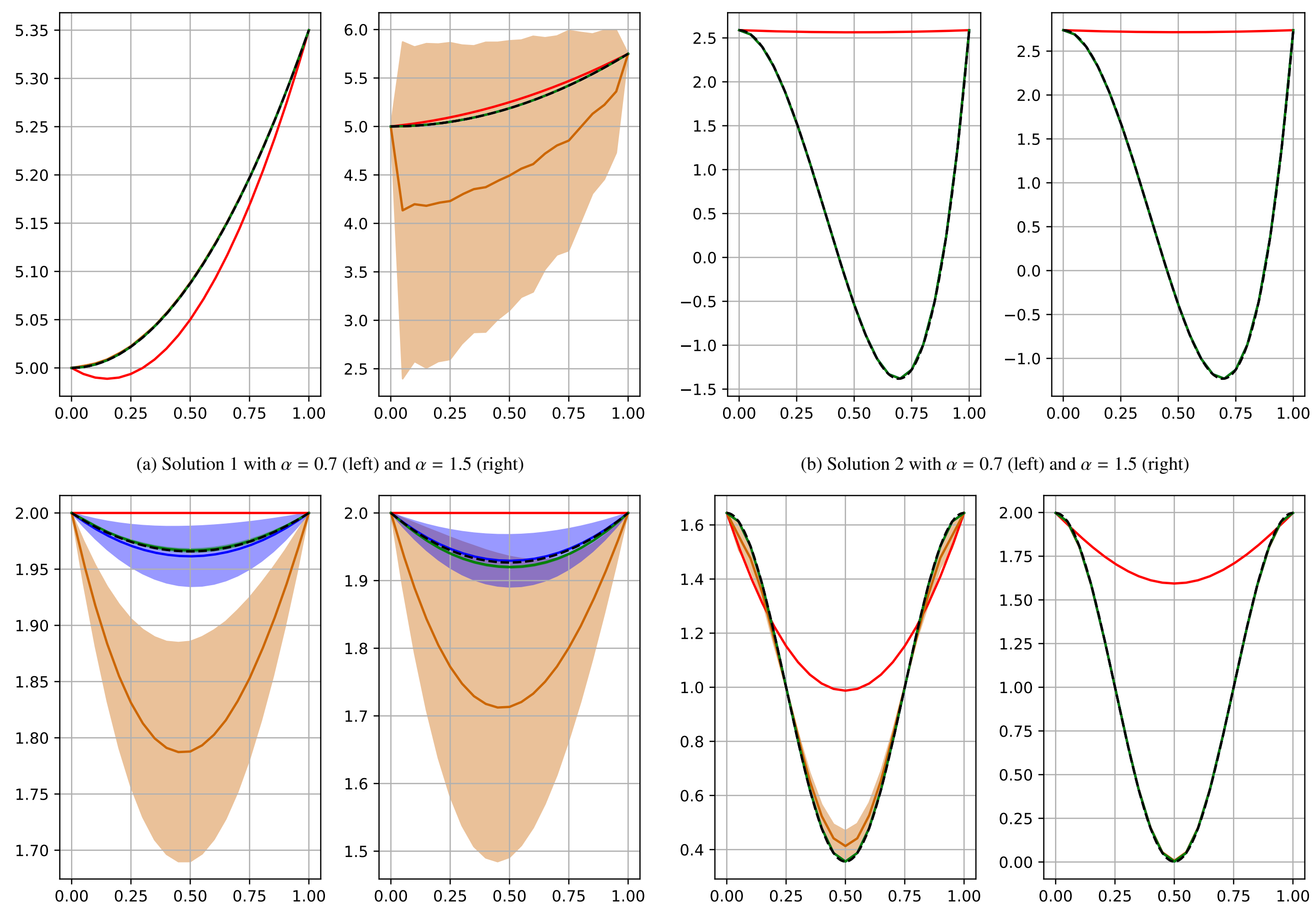
Figure 2: Temporal development of relative l_2 error for solutions 1-4 in interpolation scenarios. For solution 1,3 and 4 we observe the error from the PGNN-CoSTA method is considerably lower than all the others, while for solution 2 it is slightly higher than CoSTA. We also observe, especially for solution 4, that the error develops more smoothly. (PBM, DDM, CoSTA, PGNN-CoSTA)

5. Conclusion and future work

In the current work we combined our previously developed corrective source term approach (CoSTA) and physics-guided neural network (PGNN) to develop a new hybrid approach called the physics-guided neural network-assisted corrective source term approach to hybrid analysis and modeling (HAM). Through the application of this hybrid approach to a series of heat diffusion problems, we concluded the following:

- Accuracy: In the absence of knowledge of the source term, pure physics based modeling (PBM) has the worst accuracy. Data-driven modeling is better than the PBM on some cases, but does occasionally and seemingly unpredictably give very bad predictions. CoSTA and PGNN-CoSTA are several orders of magnitude more accurate than both PBM and DDN. The combined PGNN-CoSTA approach turns out to be most consistently accurate across all the scenarios investigated. The injection of physical parameters also suppresses the noise, resulting in smoother profiles.
- Uncertainty: Unlike the deterministic PBM, the HAMs and DDMs have random initializations and therefore uncertain predictions. The DDM has the largest uncertainty in most cases, while the CoSTA methods have much lower uncertainty. Injection of known physical parameters in the CoSTA gives a lower uncertainty in most cases.
- Generalization: While the PBM is untrained and not affected by extrapolation, the other models generally perform worse in the extrapolation cases than in the interpolation cases. The DDM is most affected by this. The PGNN-CoSTA generalizes anywhere from as good as the unguided CoSTA, to much better than it.

These results demonstrate a great potential in using CoSTA



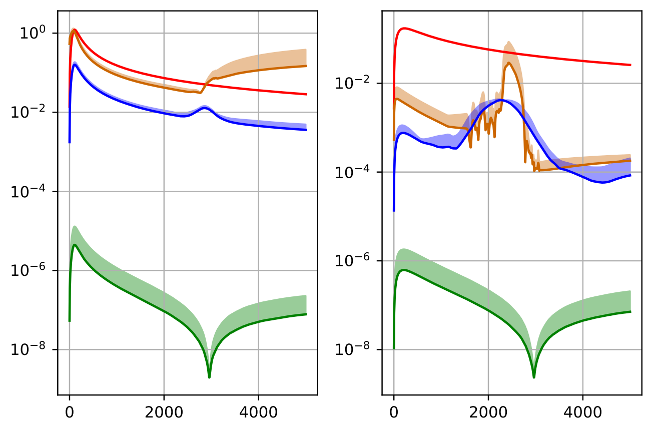
(a) Solution 1 with $\alpha = 0.7$ (left) and $\alpha = 1.5$ (right)

(b) Solution 2 with $\alpha = 0.7$ (left) and $\alpha = 1.5$ (right)

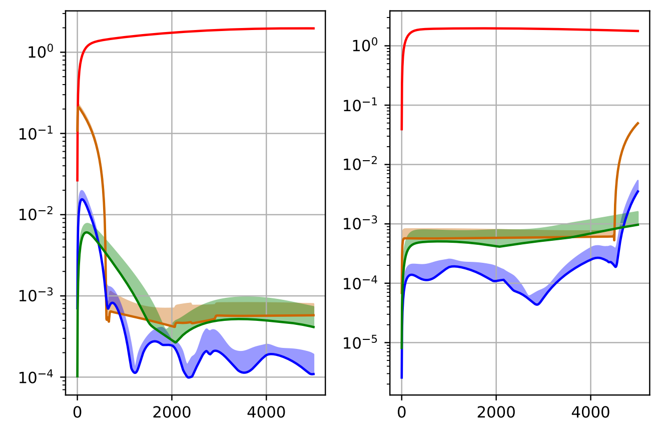
(c) Solution 3 with $\alpha = 0.7$ (left) and $\alpha = 1.5$ (right)

(d) Solution 4 with $\alpha = 0.7$ (left) and $\alpha = 1.5$ (right)

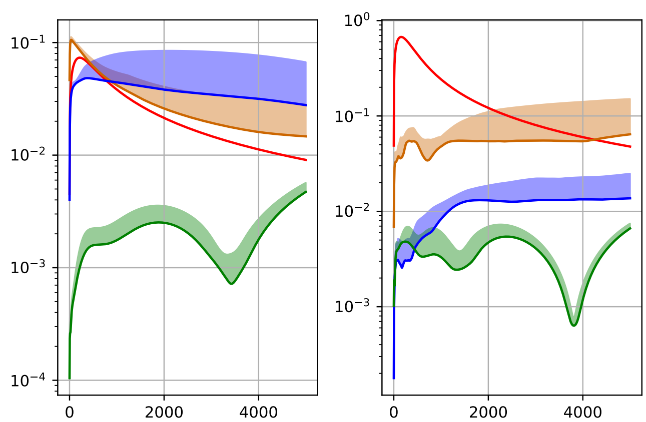
Figure 3: Final solution for solutions 1-4 in interpolation scenarios. The mean of CoSTA and PGNN-CoSTA get quite good predictions in all cases, but the CoSTA has a large uncertainty for solution 3. The pure DNN has very varying performance. (Exact, PBM, DDM, CoSTA, PGNN-CoSTA)



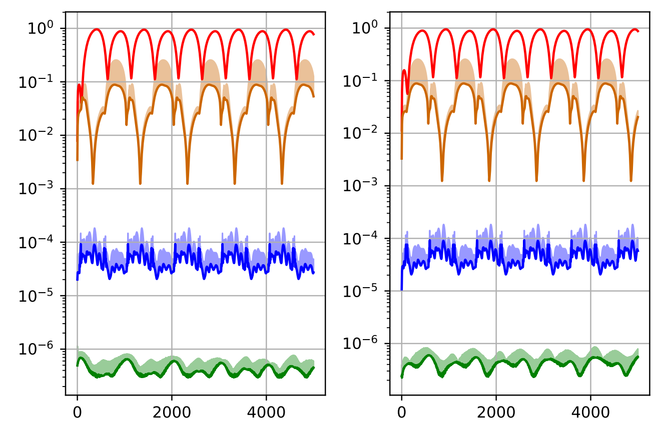
(a) Solution 1 with $\alpha = -0.5$ (left) and $\alpha = 2.5$ (right)



(b) Solution 2 with $\alpha = -0.5$ (left) and $\alpha = 2.5$ (right)

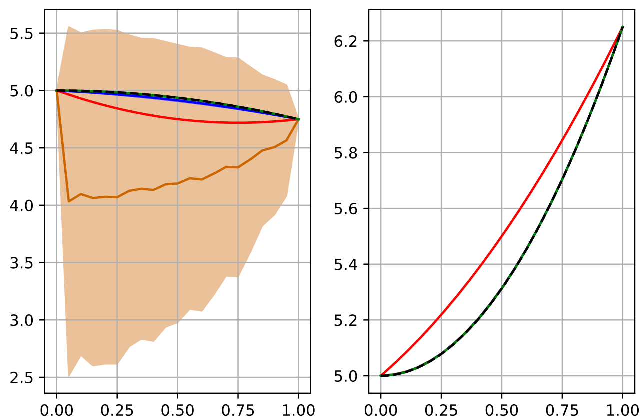


(c) Solution 3 with $\alpha = -0.5$ (left) and $\alpha = 2.5$ (right)

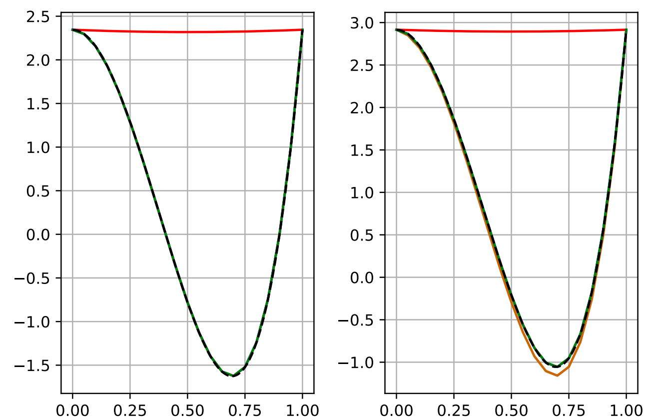


(d) Solution 4 with $\alpha = -0.5$ (left) and $\alpha = 2.5$ (right)

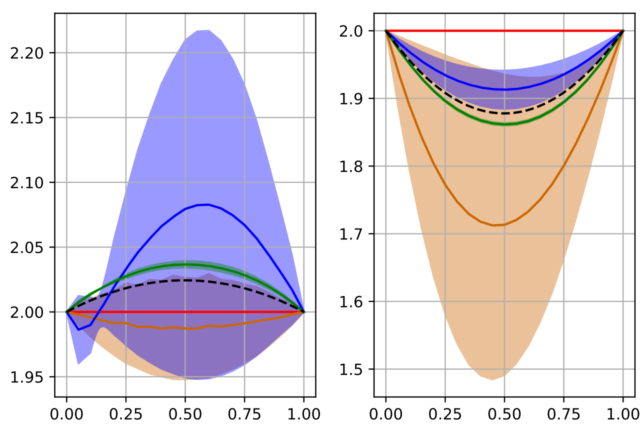
Figure 4: Temporal development of relative l_2 error for solutions 1-4 in extrapolation scenarios. As for the inpterpolation cases, the error of PGNN-CoSTA is smoother and considerably lower than the other ones, except for solution 2 where CoSTA is better. (PBM, DDM, CoSTA, PGNN-CoSTA)



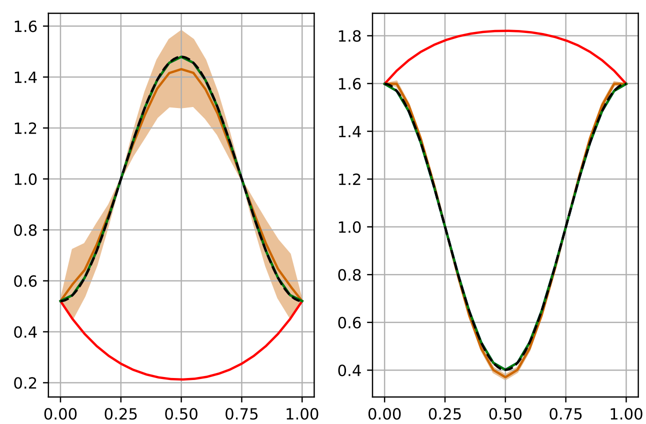
(a) Solution 1 with $\alpha = -0.5$ (left) and $\alpha = 2.5$ (right)



(b) Solution 2 with $\alpha = -0.5$ (left) and $\alpha = 2.5$ (right)



(c) Solution 3 with $\alpha = -0.5$ (left) and $\alpha = 2.5$ (right)



(d) Solution 4 with $\alpha = -0.5$ (left) and $\alpha = 2.5$ (right)

Figure 5: Final solution for solutions 1-4 in extrapolation scenarios. Solution 3 with $\alpha = -0.5$ seem to be the most difficult one, and PGNN-CoSTA seem to be the only method consistently getting a qualitatively correct shape, while also only missing slightly on the amplitude. (Exact, PBM, DDM, CoSTA, PGNN-CoSTA)

with a physics guided architecture for modelling problems where both empirical data and partial knowledge of the physics is available.

There are several potentially interesting directions for future research on this topic. Primarily, the methods should be tested on a larger variety of problems. The methods are generic, and should be applicable on systems governed by other equations, and in more dimensions. A larger variety of the nature of the unknown physics could be tried out. In this article we used unknown source term, but the method should generalize to situations where the differential operator and/or the boundary conditions are guessed. There are also a lot of different variations of the method, and it could be improved by testing other choices or configurations of the PBM or DDM to use in the CoSTA. The less fluctuating error might indicate that the PGNN-CoSTA method is more stable than the others, hence it might work well also for coarser temporal discretizations. This could open for less computationally demanding, yet still accurate predictions, which is a crucial requirement for many real time applications of digital twins.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X., 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. URL: https://www.tensorflow.org/. software available from tensorflow.org.

Blakseth, S.S., 2021. Introducing CoSTA: A Deep Neural Network Enabled Approach to Improving Physics-Based Numerical Simulations. Master's thesis. NTNU.

Blakseth, S.S., Rasheed, A., Kvamsdal, T., San, O., 2022. Deep neural network enabled corrective source term approach to hybrid analysis and modeling. To appear Neural Networks .

Brenner, S.C., Scott, L.R., 2008. The mathematical theory of finite element methods. 3rd ed., Springer, New York.

Brenowitz, N.D., Bretherton, C.S., 2019. Spatially extended tests of a neural network parametrization trained by coarse-graining. Journal of Advances in Modeling Earth Systems 11, 2728–2744.

Cheng, M., Fang, F., Pain, C., Navon, I., 2020. An advanced hybrid deep adversarial autoencoder for parameterized nonlinear fluid flow modelling. Computer Methods in Applied Mechanics and Engineering 372, 113375.

Courant, R., 1943. Variational methods for the solution of problems of equilibrium and vibrations. Bulletin of the American Mathematical Society 49, 1–23.

Ghobakhloo, M., Fathi, M., Iranmanesh, M., Maroufkhani, P., Morales, M.E., 2021. Industry 4.0 ten years on: A bibliometric and systematic review of concepts, sustainability value drivers, and success determinants. Journal of Cleaner Production 302, 127052. URL: https://www.sciencedirect.com/science/article/pii/S0959652621012713, doi:https://doi.org/10.1016/j.jclepro.2021.127052.

Grover, A., Kapoor, A., Horvitz, E., 2015. A deep hybrid model for weather forecasting, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 379–386.

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

Liu, Y., Peng, Y., Wang, B., Yao, S., Liu, Z., 2017. Review on cyber-physical systems. IEEE/CAA Journal of Automatica Sinica 4, 27–40. doi:10.1109/JAS.2017.7510349.

Maas, A.L., Hannum, A.Y., Ng, A.Y., 2013. Rectifier nonlinearities improve neural network acoustic models.

Mansor, H., Shukor, M.H.A., Meskam, S.S., Rusli, N.Q.A.M., Zamery, N.S., 2013. Body temperature measurement for remote health monitoring system, in: 2013 IEEE International Conference on Smart Instrumentation, Measurement and Applications (ICSIMA), pp. 1–5. doi:10.1109/ICSIMA.2013.6717956.

Mou, C., Koc, B., San, O., Rebholz, L.G., Iliescu, T., 2021. Data-driven variational multiscale reduced order models. Computer Methods in Applied Mechanics and Engineering 373, 113470.

Nusrat, I., Jang, S.B., 2018. A comparison of regularization techniques in deep neural networks. Symmetry 10. URL: https://www.mdpi.com/2073-8994/10/11/648, doi:10.3390/sym10110648.

Pawar, S., San, O., Aksoylu, B., Rasheed, A., Kvamsdal, T., 2021. Physics guided machine learning using simplified theories. Physics of Fluids 33, 011701.

Prajapati, H., Salvi, S.S., Ravoori, D., Jain, A., 2020. Measurement of the in-plane temperature field on the build plate during polymer extrusion additive manufacturing using infrared thermometry. Polymer Testing 92, 106866. URL: https://www.sciencedirect.com/science/article/pii/S014294182032095X, doi:https://doi.org/10.1016/j.polymertesting.2020.106866.

Quarteroni, A., 2017. Numerical models for differential problems. 3rd ed., Springer International Publishing.

Rasheed, A., San, O., Kvamsdal, T., 2020. Digital twin: Values, challenges and enablers from a modeling perspective. IEEE Access 8, 21980–22012. doi:https://doi.org/10.1109/ACCESS.2020.2970143.

Roache, P.J., 2001. Code Verification by the Method of Manufactured Solutions . Journal of Fluids Engineering 124, 4–10.

San, O., Rasheed, A., Kvamsdal, T., 2021. Hybrid analysis and modeling, eclecticism, and multifidelity computing toward digital twin revolution. GAMM-Mitteilungen 44, e202100007.

Shahidi, A., Gupta, L.A., Kovacs, A., Peroulis, D., 2013. Wireless temperature and vibration sensor for real-time bearing condition monitoring, in: 2013 IEEE MTT-S International Microwave Symposium Digest (MTT), pp. 1–4. doi:10.1109/MWSYM.2013.6697670.

Srivastava, P.K., Han, D., Ramirez, M.R., Islam, T., 2013. Machine learning techniques for downscaling SMOS satellite soil moisture using MODIS land surface temperature for hydrological application. Water Resources Management 27, 3127–3144.

Svendsen, D.H., Martino, L., Campos-Taberner, M., García-Haro, F.J., Camps-Valls, G., 2017. Joint gaussian processes for biophysical parameter retrieval. IEEE Transactions on Geoscience and Remote Sensing 56, 1718–1727.

Willard, J., Jia, X., Xu, S., Steinbach, M., Kumar, V., 2020. Integrating physics-based modeling with machine learning: A survey. arXiv preprint

arXiv:2003.04919 .
Zhang, J., Ma, W., 2020. Data-driven discovery of governing equations for fluid dynamics based on molecular simulation. Journal of Fluid Mech 892, A5.