Parallel Deep ResNets for Chemically Reacting Flows

Thomas S. Brown*, Harbir Antil †, Rainald Löhner ‡, and Deepanshu Verma §
George Mason University, Fairfax, Virginia, 22030, USA

Fumiya Togashi¶
Applied Simulations, Inc., McLean, Virginia, 22101, USA

This article continues the program initiated by the authors in [1] with an eventual goal to replace the existing chemistry software packages by Deep Neural Networks (DNNs). Chemically reacting flows are common in engineering, such as hypersonic flow, combustion, explosions, manufacturing processes and environmental assessments. For combustion, the number of reactions can be significant (over 100) and due to the very large CPU requirements of chemical reactions (over 99%) a large number of flow and combustion problems are presently beyond the capabilities of even the largest supercomputers.

Motivated by this, novel DNNs are introduced. At first, approximation capabilities of these DNNs are established on trigonometric functions. Visualization of the neuron functions has been carried out. These simple illustrations clearly show connections with finite element functions and are believed to be the first steps towards understanding of largely black-box DNN approaches.

Next, these DNNs are applied to multiple species and reactions common in chemically reacting flows, such as H_2-O_2 and methane-O_2 reactions. Experimental results show that it is helpful to account for the physical properties of species while designing DNNs. The proposed approach is shown to generalize well.

I. Introduction

This article continues the program initiated by the authors in [1] with an eventual goal to replace the existing chemistry software packages by Deep Neural Networks (DNNs). Chemically reacting flows are common in many fields of engineering, such as hypersonic flow, combustion, explosions, manufacturing processes, and environmental assessments [2–5]. For hydrocarbon combustion and explosions the numbers of species and reactions can reach into hundreds and thousands respectively. Even with so-called reduced models [6–11], which try to keep the main species and reactions while neglecting those that are not important, typically over 100 reactions need to be updated. An expedient (and widely used) way to compute flows with chemical reactions is to separate the advection and diffusion of species from the actual reactions. In this way, the vastly different timescales of the reactants can be treated in a separate, stiff ODE solver. Such chemical reaction solvers take the given species u^{n-1} at the $n-1$th time step and desired timestep δt and update the species to u^n. In terms of a 'black box solver' this implies:

$$u^n = Chem(\delta t, u^{n-1})\,, \tag{1}$$

where $Chem$ stands for the ODE integrator of chemical reactions. Compared to a typical 'cold' flow case, the presence of these chemical reactions implies an increase of computing requirements that can exceed factors of 1:100, i.e. 2 orders of magnitude. This makes many of these flow simulations so expensive that entire classes of problems have been sidelined, waiting for faster computers to be developed in years to come. The goal here is to replace the 'black box' solver given in (1) by novel, robust Deep Neural Networks (DNNs) without sacrificing accuracy.

*Postdoctoral Associate, Center for Mathematics and Artificial Intelligence, Center for Computational Fluid Dynamics, College of Science. George Mason University, Fairfax, VA 22030.

†Director, Center for Mathematics and Artificial Intelligence, Professor, Department of Mathematical Sciences, College of Science. George Mason University, Fairfax, VA 22030.

‡Director, Center for Computational Fluid Dynamics, Professor, Department of Physics and Astronomy, College of Science. George Mason University, Fairfax, VA 22030. AIAA Member

§Center for Mathematics and Artificial Intelligence, Department of Mathematical Sciences, College of Science. George Mason University, Fairfax, VA 22030.

¶Chief Combustion Scientist, 1211 Pine Hill Road, McLean, VA 22101, USA.

The list of references on using DNNs in computational fluid dynamics (CFD) is growing fast, see for example, [12, 13]. However, the results on using DNNs in chemical kinetics are scarce. A popular approach to solve PDEs and ODEs is through the use of so-called Physics-Informed Neural Networks (PINNs) [14, 15]. The goal of PINNs is to minimize the PDE/ODE residual by using a neural network as a PDE/ODE solution Ansatz. The inputs to the network are space-time variables (x, t) and all the derivatives are computed using automatic differentiation. See [16] for an example of a PINN for stiff ODE systems where the only input is time.

The approach presented here fundamentally differs from the aforementioned approaches. Instead of *Physics-Informed-Neural-Networks*, the goal is to pursue *Learn-from-Physics/Chemistry*. For instance in (1), DNNs will be used to learn $Chem$ from a given dataset coming from physics/chemistry simulations. Such an approach to learn $Chem$ has also been considered recently in [17–19] where the authors employ an architecture that is motivated by standard feed forward networks. The authors of [20] consider a similar problem, but use an autoencoder, which is a type of DNN used to reduce the dimension of the system. Notice that the proposed approach will allow for the chemical reactions described by (1) to start at any point in time without knowing a reference time. The latter is crucial for the application under consideration.

The DNNs used in this paper have been motivated by the Residual Neural Network (ResNet) architecture. ResNets have been introduced in [21–23] in the context of data/image classification, see also [24] for parameterized PDEs and [25] where the (related) so-called Neural ODE Nets [26] have been used to solve stiff ODEs. The ResNet architecture is known to overcome the vanishing gradient problem, which has been further analyzed using fractional order derivatives in [23]. The key feature of a ResNet is that in the continuum limit, it becomes an optimization problem constrained by an ODE. Such a continuous representation further enables the analysis of the stability of the network using nonlinear ODE theory. In addition, standard approaches from optimization, such as the Lagrangian approach can be used to derive the sensitivities with respect to the unknown weights and biases.

This paper considers the following aspects:

- It illustrates the approximation capabilities of the proposed DNNs on trigonometric functions. Visualization of the neuron functions has also been carried out. These simple illustrations clearly show connections with finite element functions and are believed to be the first steps towards understanding of largely black-box DNN approaches.
- Motivated by chemically reacting flows, the goal is to create networks that can learn multiple reactions propagating multiple species. To achieve this, parallel ResNets are constructed where the data corresponding to multiple quantities is used as input for each network but the output is only a single species. Similar approaches for chemical kinetics can be found in [17, 18], where the authors use standard feed forward networks.
- The novel DNNs are applied to non-trivial chemically reacting flows. Experimental results show that it is helpful to know the underlying properties of the species while designing the networks.
- A study to compare various initializations for DNN training for the application under consideration. The 'box initialization' introduced in [27] appears to have better generalization / prediction properties over the existing approaches.

The remainder of the paper is organized as follows: In Section II, the DNNs used to approximate systems of type (1) are introduced, and training strategies are discussed. In section III, the approximation capabilities of ResNets are tested on trigonometric functions. Comparison between various initializations, while training of H_2-O_2 reactions are also discussed. Various scenarios (varying temperatures, equivalence ratios etc.) have been considered. Finally, details on a more challenging case of methane-O_2 chemical reaction have been provided.

II. Deep Residual Neural Networks

A. Problem Formulation

Consider an input-to-output map

$$u \mapsto S(u),$$

where S could be $Chem$ in (1). The goal is to learn an approximation $\widehat{S}$ of S using Deep Neural Networks (DNNs). In particular, the proposed DNNs are motivated by Deep Residual Neural Networks (ResNets). See [22, 23] for examples of ResNets for classification problems, [24] for an application to parameterized PDEs, and [1] for previous work on chemically reacting flows.

Given a training dataset $\{(u^n, S(u^n))\}_{n=0}^{N-1}$, the DNN approximation $\widehat{S}$ of S is given by the output of the DNN

(ResNet)

$$\begin{cases} y_\ell = P_{\ell-1} y_{\ell-1} + \tau\sigma(K_{\ell-1} y_{\ell-1} + b_{\ell-1}), & 1 \le \ell \le L-1, \\ y_L = K_{L-1} y_{L-1}, \end{cases} \tag{2}$$

i.e., $\widehat{S}(u) = y_L$. Here $y_0 \in \mathbb{R}^{n_0}$ represents the input vector, so that during training, y_0 is taken to be from the training data set $\{u^n\}_{n=0}^{N-1}$. The scalar $\tau > 0$ is a given parameter called the skip connection parameter. The number of layers (depth of the network) is denoted by L. Layer 0 is referred to as the input layer, layers 1 through $L-1$ as hidden layers and layer L as the output layer.

The nonlinear function σ denotes an activation function. The networks in this work are trained using the framework of PDE constrained optimization. This framework requires that the activation function be differentiable. In order to have a globally differentiable activation function, in this study σ is taken to be a smooth quadratic approximation of the ReLU function, i.e.,

$$\sigma(x) = \begin{cases} \max\{0, x\} & |x| > \varepsilon, \\ \frac{1}{4\varepsilon}x^2 + \frac{1}{2}x + \frac{\varepsilon}{4} & |x| \le \varepsilon. \end{cases}$$

Figure 1 (left) shows that $\sigma(x)$ is a good approximation of ReLU. Here, ε is taken to be 10^{-4}. This is not the only choice for activation function. In fact, some experiments (not included in this brief report) have been conducted using hyperbolic tangent as the activation function.

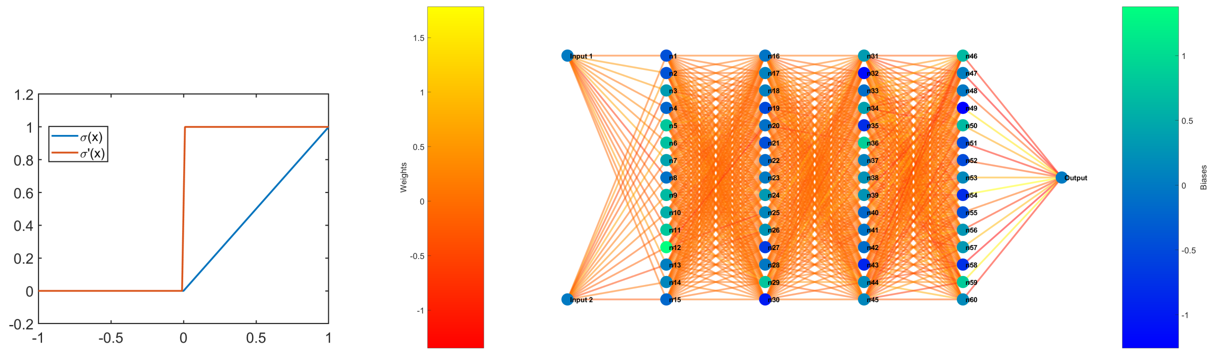

Fig. 1 Left: Smooth ReLU and its derivative for $\varepsilon = 10^{-4}$. Right: Example of a typical Deep ResNet used in the experiments.

The quantities $\{K_\ell\}_{\ell=0}^{L-1}, \{b_\ell\}_{\ell=0}^{L-2}$ denote the weights and biases, i.e. the parameters that need to be determined. In this setting K_ℓ is a matrix and b_ℓ is a vector, together they introduce an affine transformation. If $y_\ell \in \mathbb{R}^{n_\ell}$ for $\ell = 0, \dots, L$, then $K_\ell \in \mathbb{R}^{n_{\ell+1} \times n_\ell}$ and $b_\ell \in \mathbb{R}^{n_{\ell+1}}$. The dimension n_ℓ is also referred to as the width of the ℓ-th layer. The quantities P_ℓ represent operators, which for this application are either taken to be the zero operator (for P_0) or the identity operator (for $P_\ell, \ell > 0$). Notice that for $0 < \ell < L$ using the identity for P_ℓ means that $n_\ell = n_{\ell+1}$, i.e., the hidden layers of the network have uniform width. This is not the only choice for P_ℓ and can be easily generalized, see [22]. Nevertheless, in the current setup the dimension of the input y_0 and output y_L can be different.

An example of a typical DNN is shown in Figure 1 (right) with depth 5, width 15, input dimension 2, and output dimension 1. The values of the weights are given by the color of the lines connecting the neurons, the values of the biases are given by the color of the neurons, and the color of the input layer has been set to zero.

The question remains of how to compute the weights $\{K_\ell\}_{\ell=0}^{L-1}$ and biases $\{b_\ell\}_{\ell=0}^{L-2}$ in order to obtain a good approximation $\widehat{S}$ of S. Following [22, 23], these weights are computed by solving

$$\min_{\{K_\ell\}_{\ell=1}^{L-1}, \{b_\ell\}_{\ell=0}^{L-2}} \left\{ J(K_\ell, b_\ell) := \frac{1}{2N} \sum_{n=0}^{N-1} \|y_L^n - S(u^n)\|^2 \right\} \quad \text{subject to constraints} \quad (2), \tag{3}$$

where $y_L^n = \widehat{S}(u^n)$, that is, y_0 in (2) is taken to be the training data $\{u^n\}_{n=0}^{N-1}$. The problem (3) is a constrained optimization problem. To solve this problem, the gradient of J with respect to K_ℓ and b_ℓ needs to be evaluated. This requires introducing the so-called adjoint equation (also known as back-propagation in machine learning literature). See

[22–24] for complete details. Having obtained the gradient, an approximate Hessian is computed via the BFGS routine. Both the gradient and the approximate Hessian are used to solve the minimization problem.

In the definition given in (2), taking the operators P_ℓ all to be zero and $\tau = 1$ results in the standard feedforward deep neural network (FFDNN). In section III some results are presented comparing ResNets with FFDNNs.

Scaling. The training data in the examples below has been computed using CHEMKIN [28]. Before training the networks, the data is scaled in the following manner: For a data set $\{x_j\}_{j=0}^N$, each entry x_i is scaled as

$$\widehat{x}_i := \frac{x_i - \min_j x_j}{\max_j x_j - \min_j x_j}, \tag{4}$$

so that the resulting data set $\{\widehat{x}_j\}_{j=1}^N$ lies in the interval $[0,1]$. Given that chemical reactions follow an exponential Arrhenius-type law, a logarithmic scaling was also tried. This implies performing the above scaling on the dataset $\{\log x_j\}$ instead of $\{x_j\}$.

Architecture. For the DNNs implemented below, the input dimension will often be given by

$$n_0 = \underset{\text{(temperature)}}{1} + \underset{\text{(\# of species)}}{M} + \underset{\text{(time increment, }\delta t\text{)}}{1} = M + 2.$$

Rather than using a single DNN to approximate the entire solution map S, a parallel ResNet architecture is implemented in which a separate ResNet is created to correspond to each one or more desired output. For many of the results, a ResNet is built for each desired output, and therefore the output dimension for each ResNet is 1, but there are $M + 1$ ResNets implemented in parallel. The inputs to all of the parallel ResNets are the same, and so the parallel architecture can also be thought of as a single large ResNet (with $n_L = M + 1$) that is not fully connected.

The choice to use parallel ResNets was motivated by the work reported in [17, 18]. Previous results coming from a single ResNet had difficulties learning the different species, suggesting that larger, more robust networks were needed. Rather than attempt to train a large single network, the choice was made to use parallel networks. There is a further advantage in using smaller parallel networks over a larger single network, as the parallel networks can be (and are) trained in parallel.

Initialization. Before training the networks, the weights and biases must be initialized. In the experiments presented in Section III, two different methods are used to initialize the parameters. The first method is the *Xavier* or *Glorot* initialization method (see [29]). The *Box* initialization of Cyr et al., [27], is also implemented. Experiments comparing these two initializations are presented below in Section III.

Loss Function and Training. In the case of a parallel ResNet architecture, each parallel network is associated to a separate loss function. The same form of the loss function is used for each network. Letting $\theta^{(i)}$ represent the concatenation of all weight and bias parameters associated to the i-th parallel network, the loss function takes the form

$$\frac{1}{2N}\sum_{n=0}^{N-1} \|y_L^{(i)} - S(u^n)^{(i)}\|_2^2 + \frac{\lambda}{2}\left(\|\theta^{(i)}\|_1 + \|\theta^{(i)}\|_2^2\right), \qquad i = 1,\ldots,M+1. \tag{5}$$

In other words, the process of training each network is the process of finding the parameters $\theta^{(i)}$ which minimize the mean squared error between the network output and the training data, while also using both ℓ^1 and ℓ^2 regularizations to penalize the size of the parameters. As indicated in section II, a gradient based method (BFGS in particular) is used to solve the constrained optimization problem.

Validation. DNNs are trained with validation data in order to overcome the overfitting issue. The validation data is a subset of the training data that is not used to update the weights and biases. Instead, a separate loss function is formed that computes the mean squared error between the ResNet output and the validation data, without regularization terms. This is a way to test how the ResNet performs on unseen data during the training process itself. If the validation error increases, the training continues for a certain number of additional iterations, called the *patience*. During this time, if the validation error decreases to a new minimum, the patience resets to zero and training continues. If the validation error does not attain a new minimum and the full number of patience iterations is reached, then the training process is terminated.

Testing. After training and validation comes the testing phase in which the DNN approximations are used to carry out time marching. Given a known initial condition, u^0, the parallel ResNets are used to compute all subsequent values, that is

$$\widehat{u}^n = \widehat{S}(\widehat{u}^{n-1}) \qquad n > 0.$$

Applications. The above methods are applied separately to two systems of ODEs that model different chemical reactions. The first system of ODEs model a hydrogen-oxygen reaction. In particular, the reduced hydrogen-air reaction model with 8 species and 18 reactions [30] is used. This model is simpler than the hydrocarbon reaction model mentioned in section I. However, it can still capture several essential key features. The second system is a reduced methane-O_2 reaction model comprising of 15 species and 66 reactions. This model has been reduced from a model with 53 species and 325 reactions for the purpose of training the neural networks. This is a first step in using the parallel ResNet architecture to capture these reactions. Increasing the number of species and reactions will be part of future work.

III. Results

In this section a variety of results are presented that represent different approaches and architectures. The DNN implementation used in this paper was carried out in MATLAB and a BFGS routine with an Armijo line-search was used to solve the training (optimization) problem. Unlike many neural network software packages, this code uses the framework and closely resembles code that is used to solve optimal control problems [31]. The first set of results compare a feedforward deep neural network and a Deep ResNet that were trained to learn the function $\sin(x)\sin(y)$ on the domain $[0, 2\pi]^2$. Next, parallel ResNet approximations of a system that models H_2-O_2 reactions are presented. Different experiments were performed for these results using Xavier initialization and box initialization, and a comparison between the two is made. The final set of results pertain to the more difficult problem of modeling methane-O_2 reactions.

In all of the experiments, the loss function is as given in (5) with $\lambda = 1e-7$, the skip connection parameter (in (2)) $\tau = 2/(L-1)$, where L is the depth of the network, and a constant time increment of $\delta t = 1e-7$. Unless otherwise specified, all training data was scaled by the method outlined in (4). All of the results presented below were achieved by using the MATLAB implementation of the ResNets described above. The blue curves in the plots below represent 'true solution' (data generated using CHEMKIN), while the dashed red curves represent DNN output.

A. Comparison between feedforward DNNs and ResNets.

The first results compare feedforward DNN and ResNet approximations of the function $\sin(x)\sin(y)$. Note that for this set of results, only a single neural network is used for each approximation. The reason for the inclusion of this is example is to illustrate approximation capabilities of DNNs and to compare the neuron functions with finite element bases. A grid of 10,000 points was generated in the square $[0, 2\pi]^2$ by taking the Cartesian product of 100 evenly spaced points in both the $x-$ and $y-$ directions. These 10,000 points and their image under the function $\sin(x)\sin(y)$ were then randomly split into training data (4,000 points), validation data (2,000 points), and testing data (4,000 points). The patience used for the validation data was 400 iterations.

Using the 4,000 training points and 2,000 validation points a ResNet and feed forward DNN (FFDNN) were trained to approximate the function $\sin(x)\sin(y)$. Each network had a depth of 5 and the hidden layers for each network had a width of 15. The ResNet trained for 9,895 BFGS iterations, and the FFDNN for 13,977 iterations. In Figure 2, the results of the trained networks on the 4,000 test points is shown with the ResNet approximation on the left and the FFDNN approximation on the right. The exact values are in blue and the neural network approximations are in red. In order to get a better picture of the approximations, eight side views of the approximation are also shown. While the ResNet trained for fewer iterations, the approximation produced is more accurate as can be seen using from the error shown in the first panels, respectively.

In Figure 3 the neuron functions in the last hidden layer are presented for each network. These are the functions prior to being weighted, and added together, by the weight matrix in the final layer. To produce these images, 50 evenly spaced points were generated in the interval $[0, 2\pi]$. Two copies of these points were then used to create grid of 2500

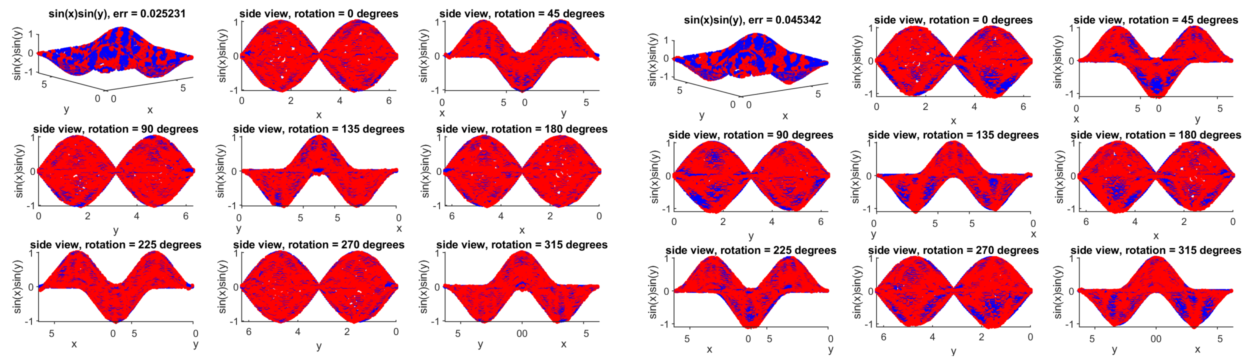

Fig. 2 Results (upper left plots) and rotated side views of the results for the ResNet (left) and FFDNN (right) approximations. The ResNet outperforms the FFDNN in terms of the accuracy.

points by Cartesian product. This grid of 2500 points was then propagated through each network to produce the images in Figure [3]. The functions produced by the ResNet are more global, i.e., have larger support, in comparison to those produced by FFDNN. This is expected and is a result of the skip connections employed in the ResNet architecture. Additionally it can be seen that several of the neurons are inactive in the entire domain (identically zero) for the FFDNN, while this is not the case for the ResNet. The could further explain the higher accuracy of ResNets.

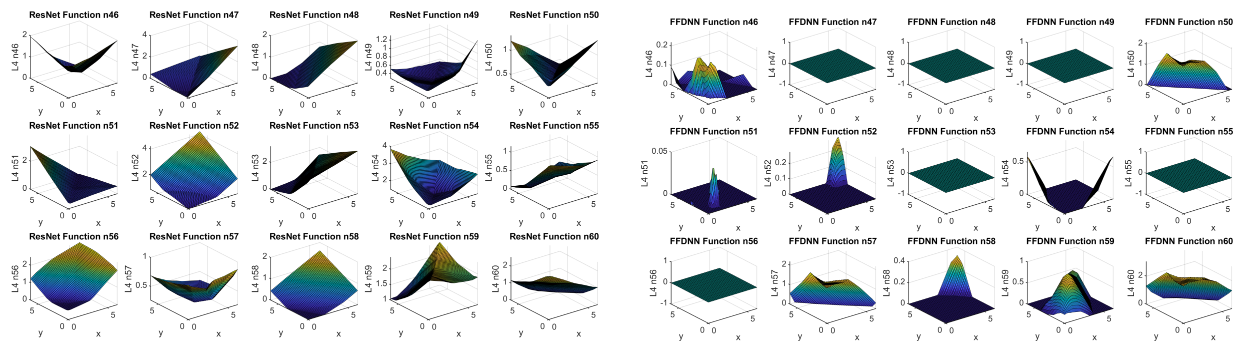

Fig. 3 Neuron functions for the fourth layer of each network (neurons 46-60). Plots of functions produced by the ResNet are grouped on the left and those for functions produced by the FFDNN are on the right.

B. H_2-O_2 reaction model using data with fixed equivalence ratio and varying initial temperature.

For the first application to chemical kinetics, parallel ResNets were trained on data sets that had initial conditions with the same equivalence ratio, but different initial temperatures. Specifically, the entire data set consists of 13 subsets with each subset having 499 points and an equivalence ratio of 1. Each subset has a different initial temperature beginning with 1,200K and increasing by increments of 100K to 2,400K. For the experiment shown in Figure [4], the ResNets were trained with the sets corresponding to initial temperatures 1,200K, 1,500K, 1,800K, 2,100K, and 2,400K.

The results of an experiment using DNNs trained on this data can be seen in Figure [4]. Both sets of 9 plots show results from the same set of parallel ResNets with a depth of 6 and width of 30. On the left the ResNets were tested using the initial condition with temperature 1,700K, while the results on the right were created by testing the ResNets with the initial condition with temperature 2,200K. For both sets of plots, the x-axis is scaled logarithmically in order to show the details in the HO_2 curve which happen quickly at the beginning of the reaction. Notice that while the DNN results (dotted red curves) on the right match the known data (solid blue curves) quite well, the results on the left barely reacted at all. The discussion in the next section focuses on using a new initialization recently introduced in [27] instead of the Xavier's initialization. This will significantly improve the results for the 1,700K case.

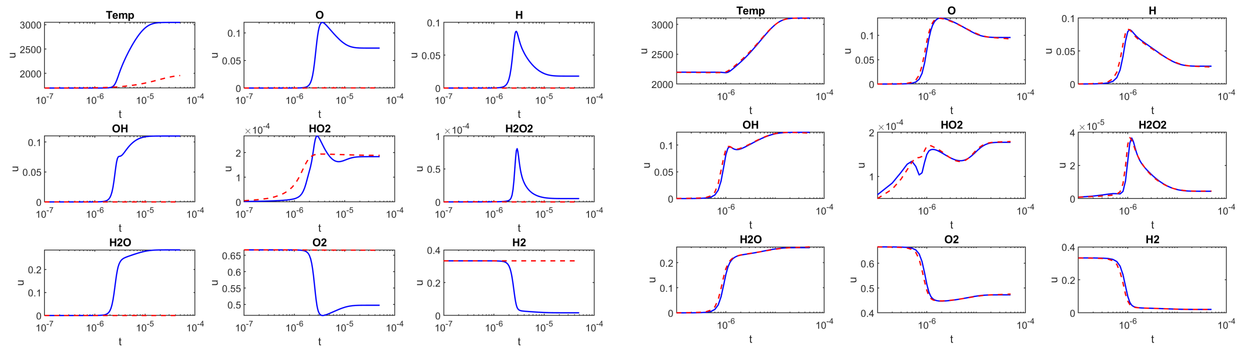

Fig. 4 Testing results from ResNets trained on data with variable initial temperatures. *Left:* Initial temperature of 1,700K and *Right:* Initial temperature of 2,200K. Both these temperatures were not part of the training data. All plots show the results of marching in time (known data in blue, ResNet results in red). The plots were produced from ResNets with a depth of 6 and a width of 30, and were trained with validation data with patience of 400 iterations.

C. Comparison using box initialization

The ResNets in the previous subsection were obtained using DNNs trained using Xavier initialization. For the next set of results, the same data sets have been used to train parallel ResNets, but instead of Xavier, *box initialization* is used to train the DNN. This new type of initialization has been recently introduced in [27]. Figure 5 shows the new testing results as dotted yellow curves. The solid red curves are same as in Figure 4. The only difference between the ResNets used to produce the results are the method of parameter initialization, all other hyperparameters are kept identical. These results show that the networks initialized by box initialization are able to generalize better to unseen data. This is even the case for the plots on the right, as the ResNets using box initialization have produced more accurate results. See in particular the results for HO_2 (middle plot). Unless otherwise specified, all remaining results in this work were created by using box initialization to determine the initial parameters.

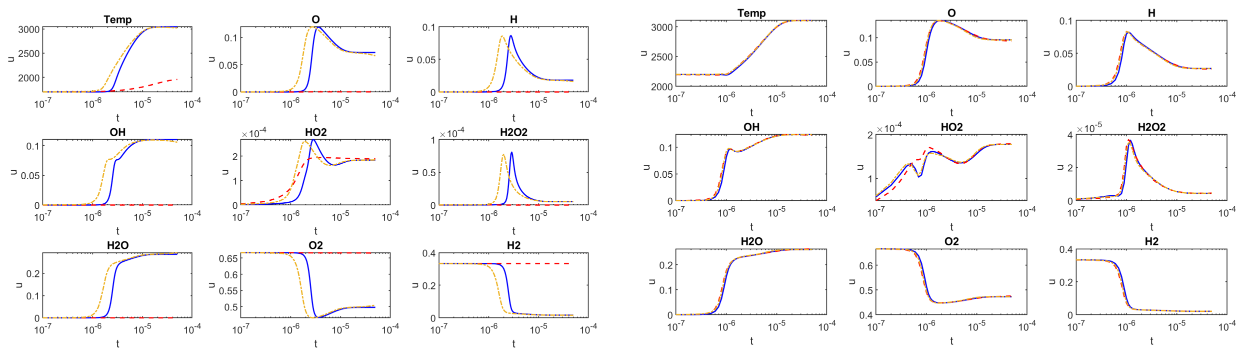

Fig. 5 Comparison of testing results where the training parameters were initialized by Xavier initialization (dashed red curves) and by box initialization (dotted yellow curves). The left and right panels, respectively, show results 1,700K and 2,200K (cf. Figure 4). A significant improvement can be seen in the 1,700K case, but even the results for 2,200 case are better as can be seen in the HO_2 panel on the right.

D. Grouping training data based on equivalence ratio.

The next set of results use a data set from the H_2-O_2 reaction model consisting of subsets with two initial conditions and varying equivalence ratio. Experimental results showed that it was difficult for the ResNets to learn the data with a wide range of equivalence ratios. In order to overcome this difficulty, the larger data set is split into three groups: fuel lean (equivalence ratio ≤ 0.1), fuel balanced (equivalence ratio between 0.1 and 2), and fuel rich (equivalence ratio > 2).

These groupings are described in Table 1. This data was used to create the results shown in Figure 6. The ResNets trained on fuel lean sets (top left set of 9 plots) were trained with subsets 1, 3, 4, 6, and 9, and then tested with the initial condition from set 8. The ResNets trained on the fuel balanced sets were trained on subsets 1, 3, 5, 7, 9, and 12, and subsequently tested with the initial condition from set 2. Finally the networks trained on fuel rich sets were trained with subsets 1, 2, 3, 5, and 7, and then tested with the initial condition from set 4. All of the hidden layers of the ResNets used to produce the results have a width of 30. The depths of the ResNets were 6, 13, and 5 for the fuel lean, balanced, and rich results respectively. Furthermore, these plots are presented with a logarithmically scaled x-axis, as for some of the data the reactions happen very quickly. All of the plots in Figure 6 show results of marching in time given only an initial condition from known data that was not seen by the networks during training. While these results are less accurate than those shown in Figure 5, they are more accurate than those seen prior to employing the fuel-based grouping.

Fuel Lean Sets									
Set	1	2	3	4	5	6	7	8	9
Number of points	7,999	5,999	4,998	4,999	4,999	4,999	4,999	4,999	4,999
Equivalence ratio	0.01	0.05	0.1	0.01	0.02	0.04	0.06	0.08	0.1
Initial temperature	1,200	1,200	1,200	1,500	1,500	1,500	1,500	1,500	1,500

Fuel Balanced Sets												
Set	1	2	3	4	5	6	7	8	9	10	11	12
Number of points	1,998	998	998	998	4,999	4,999	4,999	4,999	4,999	4,999	4,999	4,999
Equivalence ratio	0.25	0.5	1	2	0.2	0.4	0.5	0.75	0.9	1	1.5	2
Initial temperature	1,200	1,200	1,200	1,200	1,500	1,500	1,500	1,500	1,500	1,500	1,500	1,500

Fuel Rich Sets							
Set	1	2	3	4	5	6	7
Number of points	1,998	3,998	4,999	4,999	4,999	4,999	4,999
Equivalence ratio	5	10	2.5	3	3.5	4.5	5
Initial temperature	1,200	1,200	1,500	1,500	1,500	1,500	1,500

Table 1 Description of the training data with fuel-based grouping that was used for the results in Figure 6.

E. Methane-O_2 reaction

For the next set of results, networks are trained with data generated from a methane-O_2 reaction model with 15 species and temperature. The generated data contains 27 data subsets with a fixed equivalence ratio of 1 and initial temperatures varying from 1,200K to 2,500K in increments of 50K. Table 2 contains information about the size of the data subsets. For the results in Figure 7 16 parallel ResNets with depth 8 and width 30 were trained on data sets with initial temperatures 1,200K, 1,350K, 1,500K, 1,650K, 1,800K, 1,950K, 2,100K, 2,250K, 2,400K, and 2,500K. For this experiment only ℓ^2 regularization was used in the loss function (refer to (5)). For the left set of plots the resulting ResNets were tested with an initial condition with temperature 1,250K, while the plots on the right were produced from an initial condition with temperature 2,350K. In the plots on the left, the approximations by parallel ResNets start the reaction early. This is one of the challenges that has been observed from these experiments for data with lower initial temperatures.

While training parallel networks is faster than training a single neural network, using a single network for each species and temperature separately becomes increasingly expensive as the number of species involved in the reaction grows. For instance, while the results involving the H_2-O_2 model utilized 9 parallel ResNets, the methane-O_2 model requires 16 parallel networks if trained in the same fashion. With future work including models with 30 species or more, other methods must be investigated to learn these models.

One proposed method to reduce the number of parallel ResNets used is to combine the six minor species HO_2, H_2O_2, CH_3, HCO, CH_2O, and CH_3O into a single quantity. For the results in Figures 8 and 9, 10 parallel ResNets were trained on this modified methane-O_2 reaction model data. The input dimension of each network was 12, corresponding to temperature, the 9 major species, the sum of the 6 minor species, and the time increment δt. The output dimension of each ResNet remained 1, and separate networks were used to learn the temperature and the 9 major species.

After training, the temperature and major species are marched forward in time from a known initial condition just as

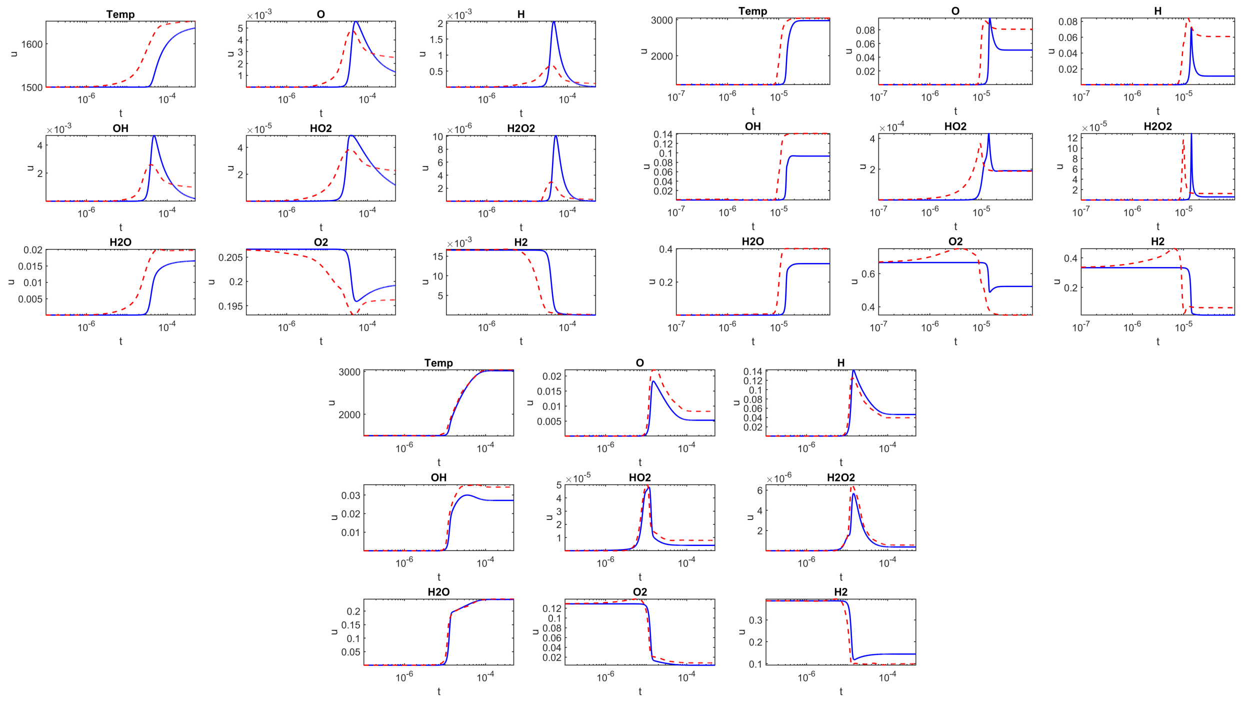

Fig. 6 Testing results from marching initial conditions forward in time using ResNets trained on fuel lean data (top left), fuel balanced (top right), and fuel rich (bottom) data sets.

with the other results presented above. A ResNet was not used to update the sum of the minor species. Instead, using the property that the density of the 15 species in the data was fixed at 1, the sum of the minor species was updated by subtracting the density of the updated major species from 1. In other words, let $\mathbf{v}^n = (v_k^n)_{k=1}^9$ represent the value of the 9 major species at the n-th timestep. The sum of the minor species at the n-th timestep, w^n, is then updated as

$$w^n = 1 - \sum_k v_k^n.$$

Once updated, the sum of the minor species, w^n, along with the major species, temperature, and δt are input into the parallel ResNets to predict the major species and temperature at the next timestep.

The results in Figures 8 and 9 were produced by 10 parallel ResNets trained on the same data sets as the results presented in Figure 7, except with the minor species added together as described above. In Figure 8 Xavier initialization was used to produce the initial weights and bias, whereas box initialization was used to produce the results in Figure 9. In both sets of results, the loss function employed both ℓ^1 and ℓ^2 regularization on the paramters. The ResNets used to produce the results in Figure 8 have a depth of 4 and width of 35. After training, the networks were tested by marching forward in time known initial conditions with temperatures of 1,750K and 2,200K. The behavior of the minor species was accurately captured by the networks. In the results corresponding to 1,750K the predicted reaction occurs earlier than the known data. This remains a challenge for the results using the methane-O_2 model. For the results in Figure 9 the ResNets used have a depth of 4 for the results in the left panel and a depth of 6 for the results in the right panel. All hidden layers in both sets of results have a width of 35. After training, the networks were tested by marching forward in time known initial conditions with temperatures of 1,850K for the results on the left and 2,200K for those on the right.

IV. Conclusions and Future Directions

A number of ResNet based approaches to approximate the stiff ODEs arising in chemical kinetics were developed. The experiments presented for the H_2-O_2 model indicate that when the equivalence ratio is fixed and initial temperature is varied (this case has been considered in [18, 19]), the proposed DNNs can (almost) perfectly generalize to unseen data. This capacity for generalization deteriorates, however, when the initial temperature is kept fixed and the equivalence ratio is varied. In order to overcome this issue, the training data was separated into fuel lean, fuel balanced, and fuel

Init. temp.	Number of points	Init. temp.	Number of points	Init. temp.	Number of points
1,200	30,000	1,650	4,999	2,100	399
1,250	30,000	1,700	4,999	2,150	199
1,300	30,000	1,750	999	2,200	199
1,350	30,000	1,800	999	2,250	199
1,400	10,000	1,850	999	2,300	399
1,450	4,999	1,900	399	2,350	199
1,500	4,999	1,950	399	2,400	199
1,550	4,999	2,000	399	2,450	199
1,600	4,999	2,050	399	2,500	199

Table 2 Description of the training data for the methane-O_2 reaction model that was used for the results in Figures 7 and 8.

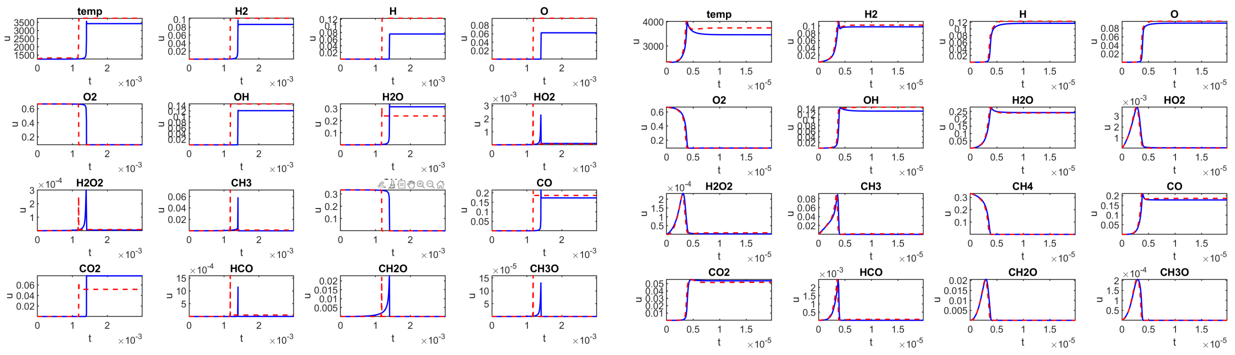

Fig. 7 Testing results from ResNets trained on data that varies the initial temperature. The ResNets used have width of 30, a depth of 8. The plots show the results of marching in time from an initial condition with temperature 1,250K (left plots) and 2,350K (right plots).

rich based on the equivalence ratio. This approach has led to encouraging results. Additionally, the networks can be improved by using the box method to initialize the weights and biases at the beginning of the training procedure.

Experiments representing a first step in using DNNs to capture the more complicated problem of a reduced methane-O_2 reaction model are also presented. This model presents additional challenges due to the increased number of species and complexity of the reactions trying to be captured. The experiments show that updating the minor species by using the fixed density of the species rather than using separate neural networks leads to comparably accurate results with the advantage of reduced training costs.

There are several questions that are currently being investigated:

- All DNNs are dependent on the quality of training data. The data used for experiments reported here were generated using CHEMKIN. In view of the high dimensionality of the data, and the nature of chemical reactions (large periods of inactivity followed by a sudden reaction followed by a convergence to a steady state), quality criteria need to be developed to make sure redundant data is avoided.
- In the case of the H_2-O_2 model, it will be interesting to see if further dividing the training data based on equivalence ratio is helpful.
- The usefulness of training with noisy data is also being explored. For some classification problems, this approach is known to increase the robustness of ResNets.
- For both models considered above, it is of interest to see how the proposed approach generalizes to more reactions and species.

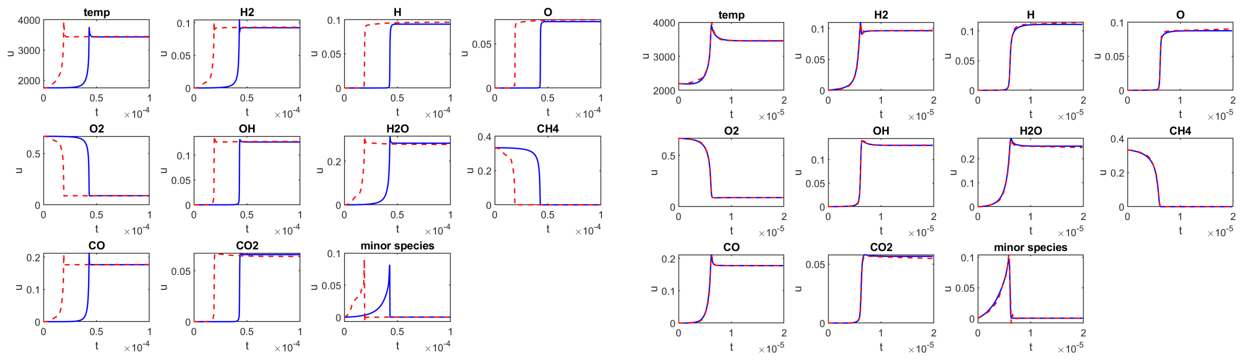

Fig. 8 Testing results from ResNets trained on data that combines the minor species in the methane-O_2 reaction model. The ResNets used have width of 35 and a depth of 4. The plots show the results of marching in time from an initial condition with temperature 1,750K (left plots) and 2,200K (right plots).

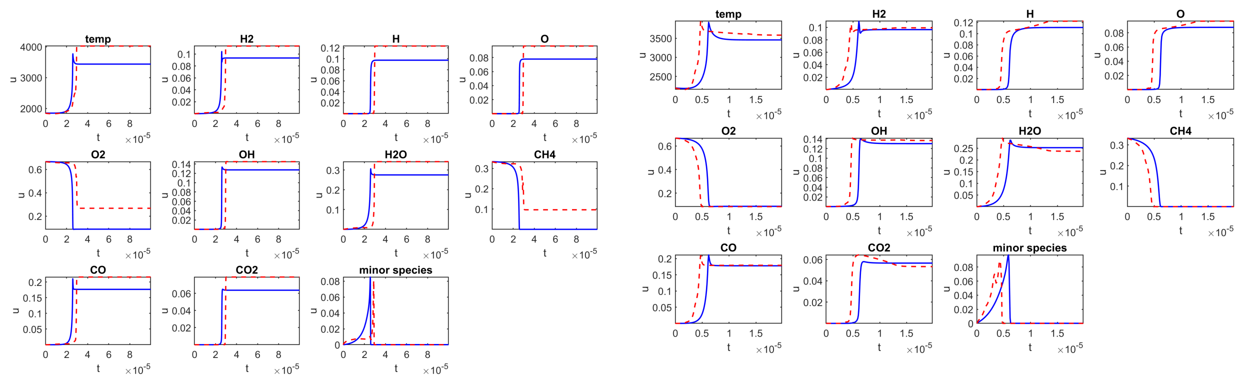

Fig. 9 Testing results from ResNets trained on data that combines the minor species in the methane-O_2 reaction model. The ResNets used have width of 35 and a depths of 4 (left plots) and 6 (right plots). The plots show the results of marching in time from an initial condition with temperature 1,850K (left plots) and 2,200K (right plots).

Acknowledgements

This work was supported by the Defense Threat Reduction Agency (DTRA) under contract HDTRA1-15-1-0068. Jacqueline Bell served as the technical monitor.
It is a pleasure to acknowledge a number of fruitful discussions with Drs. Adam Moses, Alisha J. Sharma and Ryan F. Johnson from the Naval Research Laboratory that lead to several improvements in the techniques discussed in this paper.

References

[1] Brown, T. S., Antil, H., Löhner, R., Togashi, F., and Verma, D., "Novel DNNs for Stiff ODEs with Applications to Chemically Reacting Flows," *arXiv preprint arXiv:2104.01914*, 2021.

[2] Togashi, F., Lohner, R., and Tsuboi, N., "Numerical Simulation of H2/Air Detonation Using Detailed Reaction Models," *44th AIAA Aerospace Sciences Meeting and Exhibit*, 2006-954. https://doi.org/10.2514/6.2006-954, URL https://arc.aiaa.org/doi/abs/10.2514/6.2006-954.

[3] Stück, A., Camelli, F. F., and Löhner, R., "Adjoint-based design of shock mitigation devices," *International Journal for Numerical Methods in Fluids*, Vol. 64, No. 4, 2010, pp. 443–472.

[4] Camelli, F., and Löhner, R., "Assessing maximum possible damage for contaminant release events," *Engineering computations*, Vol. 21, No. 7, 2004, pp. 748–760.

[5] Löhner, R., and Camelli, F., "Optimal placement of sensors for contaminant detection based on detailed 3D CFD simulations," *Engineering computations*, Vol. 22, No. 3, 2005, pp. 260–273.

[6] Vajda, S., Valko, P., and Turányi, T., "Principal Component Analysis of Kinetic Models," *International Journal of Chemical Kinetics*, Vol. 17, 2004, pp. 55 – 81. https://doi.org/10.1002/kin.550170107.

[7] Keck, J. C., "Rate-controlled constrained-equilibrium theory of chemical reactions in complex systems," *Progress in Energy and Combustion Science*, Vol. 16, No. 2, 1990, pp. 125–154. https://doi.org/https://doi.org/10.1016/0360-1285(90)90046-6, URL https://www.sciencedirect.com/science/article/pii/0360128590900466.

[8] Maas, U., and Pope, S., "Simplifying chemical kinetics: Intrinsic low-dimensional manifolds in composition space," *Combustion and Flame*, Vol. 88, No. 3, 1992, pp. 239–264. https://doi.org/https://doi.org/10.1016/0010-2180(92)90034-M, URL https://www.sciencedirect.com/science/article/pii/001021809290034M.

[9] Lam, S. H., and Goussis, D. A., "The CSP method for simplifying kinetics," *International Journal of Chemical Kinetics*, Vol. 26, No. 4, 1994, pp. 461–486. https://doi.org/https://doi.org/10.1002/kin.550260408, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/kin.550260408.

[10] Lu, T., and Law, C., "A directed relation graph method for mechanism reduction," *Proceedings of the Combustion Institute*, Vol. 30, 2005, pp. 1333–1341. https://doi.org/10.1016/j.proci.2004.08.145.

[11] Sun, W., Chen, Z., Gou, X., and Ju, Y., "A path flux analysis method for the reduction of detailed chemical kinetic mechanisms," *Combustion and Flame*, Vol. 157, No. 7, 2010, pp. 1298–1307. https://doi.org/https://doi.org/10.1016/j.combustflame.2010.03.006, URL https://www.sciencedirect.com/science/article/pii/S0010218010000842.

[12] Lye, K. O., Mishra, S., and Ray, D., "Deep learning observables in computational fluid dynamics," *Journal of Computational Physics*, Vol. 410, 2020, p. 109339. https://doi.org/https://doi.org/10.1016/j.jcp.2020.109339, URL https://www.sciencedirect.com/science/article/pii/S0021999120301133.

[13] Grimberg, S. J., and Farhat, C., *Hyperreduction of CFD Models of Turbulent Flows using a Machine Learning Approach*, ???? https://doi.org/10.2514/6.2020-0363, URL https://arc.aiaa.org/doi/abs/10.2514/6.2020-0363.

[14] Raissi, M., Perdikaris, P., and Karniadakis, G. E., "Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. Comput. Phys.*, Vol. 378, 2019, pp. 686–707. https://doi.org/10.1016/j.jcp.2018.10.045, URL https://doi.org/10.1016/j.jcp.2018.10.045.

[15] Cheng, C., and Zhang, G.-T., "Deep Learning Method Based on Physics Informed Neural Network with Resnet Block for Solving Fluid Flow Problems," *Water*, Vol. 13, No. 4, 2021. https://doi.org/10.3390/w13040423, URL https://www.mdpi.com/2073-4441/13/4/423.

[16] Ji, W., Qiu, W., Shi, Z., Pan, S., and Deng, S., "Stiff-PINN: Physics-Informed Neural Network for Stiff Chemical Kinetics," *arXiv preprint arXiv:2011.04520*, 2020.

[17] Peng, W. Y., and Pinkowski, N. H., "Efficient and accurate time-integration of combustion chemical kinetics using artificial neural networks," 2017.

[18] Sharma, A. J., Johnson, R. F., Kessler, D. A., and Moses, A., "Deep Learning for Scalable Chemical Kinetics," *AIAA Scitech 2020 Forum*, 2020-0181. https://doi.org/10.2514/6.2020-0181, URL https://arc.aiaa.org/doi/abs/10.2514/6.2020-0181.

[19] Owoyele, O., and Pal, P., "ChemNODE: A Neural Ordinary Differential Equations Approach for Chemical Kinetics Solvers," *arXiv preprint arXiv:2101.04749*, 2021.

[20] Zhang, P., Sankaran, R., Stoyanov, M., Lebrun-Grandie, D., and Finney, C. E., *Reduced Models for Chemical Kinetics derived from Parallel Ensemble Simulations of Stirred Reactors*, ???? https://doi.org/10.2514/6.2020-0177, URL https://arc.aiaa.org/doi/abs/10.2514/6.2020-0177.

[21] He, K., Zhang, X., Ren, S., and Sun, J., "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778. https://doi.org/10.1109/CVPR.2016.90.

[22] Ruthotto, L., and Haber, E., "Deep neural networks motivated by partial differential equations," *J. Math. Imaging Vision*, Vol. 62, No. 3, 2020, pp. 352–364. https://doi.org/10.1007/s10851-019-00903-1, URL https://doi.org/10.1007/s10851-019-00903-1.

[23] Antil, H., Khatri, R., Lohner, R. L., and Verma, D., "Fractional deep neural network via constrained optimization," *Machine Learning: Science and Technology*, 2020. URL http://iopscience.iop.org/10.1088/2632-2153/aba8e7.

[24] Antil, H., Elman, H. C., Onwunta, A., and Verma, D., "Novel Deep neural networks for solving Bayesian statistical inverse," *arXiv preprint arXiv:2102.03974*, 2021.

[25] Ghosh, A., Behl, H. S., Dupont, E., Torr, P. H. S., and Namboodiri, V., "STEER: Simple Temporal Regularization For Neural ODEs," *arXiv preprint arXiv:2006.10711*, 2020.

[26] Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K., "Neural Ordinary Differential Equations," *Advances in Neural Information Processing Systems*, Vol. 31, edited by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/69386f6bb1dfed68692a24c8686939b9-Paper.pdf.

[27] Cyr, E. C., Gulian, M. A., Patel, R. G., Perego, M., and Trask, N. A., "Robust Training and Initialization of Deep Neural Networks: An Adaptive Basis Viewpoint," *Proceedings of The First Mathematical and Scientific Machine Learning Conference*, Proceedings of Machine Learning Research, Vol. 107, edited by J. Lu and R. Ward, PMLR, Princeton University, Princeton, NJ, USA, 2020, pp. 512–536. URL http://proceedings.mlr.press/v107/cyr20a.html.

[28] Kee, R. J., Rupley, F. M., Miller, J., Coltrin, M., Grcar, J., Meeks, E., Moffat, H., Lutz, A., Dixon-Lewis, G., Smooke, M., Warnatz, J., Evans, G., Larson, R. S., Mitchell, R., Petzold, L., Reynolds, W., Caracotsios, M., Stewart, W., Glarborg, P., Wang, C., and Adigun, O., "CHEMKIN Collection, Release 3.6," , 2000.

[29] Glorot, X., and Bengio, Y., "Understanding the difficulty of training deep feedforward neural networks," *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research, Vol. 9, edited by Y. W. Teh and M. Titterington, PMLR, Chia Laguna Resort, Sardinia, Italy, 2010, pp. 249–256. URL http://proceedings.mlr.press/v9/glorot10a.html.

[30] Petersen, E. L., and Hanson, R. K., "Reduced Kinetics Mechanisms for Ram Accelerator Combustion," *Journal of Propulsion and Power*, Vol. 15, No. 4, 1999, pp. 591–600. https://doi.org/10.2514/2.5468, URL https://doi.org/10.2514/2.5468.

[31] Antil, H., Kouri, D. P., Lacasse, M.-D., and Ridzal, D. (eds.), *Frontiers in PDE-constrained optimization*, The IMA Volumes in Mathematics and its Applications, Vol. 163, Springer, New York, 2018. https://doi.org/10.1007/978-1-4939-8636-1, URL https://doi.org/10.1007/978-1-4939-8636-1, papers based on the workshop held at the Institute for Mathematics and its Applications, Minneapolis, MN, June 6–10, 2016.