ChemNODE: A Neural Ordinary Differential Equations Approach for Chemical Kinetics Solvers

Ope Owoyele*, Pinaki Pal

Energy Systems Division, Argonne National Laboratory, Lemont, IL 60439

* Contact author: oowoyele@anl.gov, phone number: 630-252-2132

Abstract

The main bottleneck when performing computational fluid dynamics (CFD) simulations of combustion systems is the computation and integration of the highly non-linear and stiff chemical source terms. In recent times, machine learning has emerged as a promising tool to accelerate combustion chemistry, involving the use of regression models to predict the chemical source terms as functions of the thermochemical state of the system. However, combustion is a highly nonlinear phenomenon, and this often leads to divergence from the true solution when the neural network representation of chemical kinetics is integrated in time. This is because these approaches minimize the error during training without guaranteeing successful integration with ordinary differential equation (ODE) solvers. In this work, a novel neural ODE approach to combustion modeling, ChemNODE, is developed to address this issue. The source terms predicted by the neural network are integrated during training, and by backpropagating errors through the ODE solver, the neural network weights are adjusted accordingly to minimize the difference between the predicted and actual ODE solutions. It is shown that even when the dimensionality of the thermochemical manifold is trimmed to remove redundant species, the proposed approach accurately captures the correct physical behavior and reproduces the results obtained using the full chemical kinetic mechanism.

Keywords: neural ordinary differential equations, machine learning, chemical kinetics.

Introduction

Chemical kinetic mechanisms for practical hydrocarbons fuels can contain hundreds of species and thousands of chemical reactions that describe the evolution of the species in time. Solving for these species in space and time is computationally challenging due to the wide range of spatio-temporal scales over which turbulent combustion occurs. Therefore, solving chemistry remains the main bottleneck in computational fluid dynamics (CFD) simulations of combustion systems. For practical engine-size geometries of interest to the automotive and aero-propulsion applications, very simplified reduced or skeletal kinetic mechanisms are typically used to maintain computational tractability. However, these models sacrifice accuracy and predictive capability.

Several studies have applied machine learning to solve these issues. Christo et al. [1] used artificial neural networks (ANNs) to represent a 4-step H_2/CO_2 chemical mechanism to perform simulations of a turbulent jet diffusion flame. Sen at al. [2] employed ANNs for modeling chemical kinetics by using them to predict the subgrid species source terms in the large eddy simulation linear eddy model (LES-LEM). In addition to these studies, neural networks have also been used to predict the chemical source terms of data-derived scalars within low-dimensional manifolds [3-5]. More recently, Ranade et al. [6] used neural networks to capture the process of pyrolysis of complex hydrocarbons, and Wan et al. [7] applied convolutional neural networks to the direct numerical simulation of a turbulent non-premixed flame interacting with a cooled wall. The overall methodology of source term predictions using machine learning methods in these previous studies is as follows. First, data is generated by running a simple problem using a complex chemical mechanism. By learning from the examples presented in the dataset from lower-dimensional simulations, the machine learning model learns to predict the source terms as functions of the thermo-chemical state. If the mean squared error between the predicted and actual source terms are below an acceptable threshold with limited scatter on the parity plots, it is assumed that the neural network can be coupled with a numerical solver and integrated to recover the true solution. However, this is seldom the case in practice. Oftentimes, the predicted solution diverges from the true solution and becomes unstable when coupled with a numerical solver. Since combustion is a highly nonlinear phenomenon, even small errors in the predictions of the source terms, especially if occurring during an early time instance, can lead to very erroneous solutions.

In this study, a novel approach toward machine learning-based calculation of chemical kinetics is proposed. As opposed to separating the learning and numerical validation phases, the approach used in the study combines them. This novel approach, termed as ChemNODE, captures chemical kinetics using a recent class of deep learning models known as neural ordinary differential equations (NODEs) [8]. ChemNODE calculates the loss function based on the actual and predicted solutions and directly learns to predict the source terms that lead to an accurate ODE solution. A description of the ChemNODE approach and demonstration studies for a canonical homogeneous auto-ignition problem are presented in this paper. The paper ends with some directions for future studies.

ChemNODE approach for chemical kinetics

This study uses NODEs to learn to predict the evolution of chemical species. During training, an ANN is used to learn the chemical source terms of species as functions of the temperature and species mass fractions. A neural network is a machine learning model that is loosely inspired by biological neural networks. It consists of successive layers that are obtained by performing linear operations and non-linear transformations of the preceding layers. The value of the *j*th hidden layer is given by:

$$\boldsymbol{h}^j = f\left(\boldsymbol{h}^{j-1}\boldsymbol{W}^j + \boldsymbol{b}^j\right) \tag{1}$$

In Eq. (1), $\boldsymbol{W}$ and $\boldsymbol{b}$ represent the weights and biases of the neural network, and f is the activation function. These weights and biases are initialized using a desired probability distribution, and are progressively tuned during training. In other words, the process of training a neural network is an optimization problem, where the goal is to find the weights and biases that minimize a loss function of interest. For regression-type problems, this loss function is typically a measure of the error between the predicted and actual values of the target variable.

The conventional practice when developing machine learning models for reacting flow simulations starts by generating data that covers a space of interest. While many earlier proof-of-concept studies performed training and validation studies using the same CFD configuration, more recent studies have generated data using approaches such as stochastic micro-mixers [7] and a variant of the pairwise mixing stirred reactor [5, 9]. Using these simpler and less expensive simulations, snapshots of data are collected at different points during the simulations. This data is used to build a database that consists of various thermochemical species and their corresponding source terms. After performing some process of manifold dimensionality reduction, either by combining or eliminating variables, a neural network is trained to learn the source terms as closed-form functions of the thermochemical state by using the database generated from inexpensive simulations as an example. After training, the neural network library is coupled with a numerical solver to compute the source terms during unsteady CFD simulations. The problem is one of finding the weights and biases, and minimize the following loss function:

$$L = \left\|\dot{\boldsymbol{\omega}}_{\Psi} - \hat{\dot{\boldsymbol{\omega}}}_{\Psi}\right\|_2^2 \tag{2}$$

In the above equation, $\dot{\boldsymbol{\omega}}_{\Psi}$ refers to the chemical source terms obtained from the full chemical mechanism, while $\hat{\dot{\boldsymbol{\omega}}}$ corresponds to the chemical source terms predicted by the machine learning model. On the other hand, the approach used in this study differs in that it involves finding the weights and biases that minimize the following loss function:

$$L = \left\|\boldsymbol{\Psi} - \widehat{\boldsymbol{\Psi}}\right\|_2^2 \tag{3}$$

In Eq. (3), $\boldsymbol{\Psi}$ and $\widehat{\boldsymbol{\Psi}}$ are vectors containing the solutions to the thermo-chemical system of equations, as obtained from the full chemical mechanism and ChemNODE, respectively. As such, the loss function in Eq. (3) measures the difference between the actual and predicted solutions.. This is in contrast to Eq. (2) where the loss indicates how well the neural network predicts the chemical source terms. The ChemNODE approach used in this study was implemented in Julia Language [10] using native packages as well as modules and functions developed in-house.

To train the neural network by minimizing the loss function described in Eq. (3), the sensitivity of the ODE solution with respect to the neural network weights and biases, $\frac{\partial L}{\partial \boldsymbol{W}}$ and $\frac{\partial L}{\partial \boldsymbol{b}}$, need to be computed. Calculating these quantities invariably involves calculating the derivatives of the ODE solution with respect to the weights and biases, $\frac{\partial \widehat{\boldsymbol{\Psi}}}{\partial \boldsymbol{W}}$ and $\frac{\partial \widehat{\boldsymbol{\Psi}}}{\partial \boldsymbol{b}}$. There are a number of ways in which this has been done in the literature. One approach involves a method known as adjoint sensitivity analysis [11], which involves framing an auxiliary ordinary differential equation whose solution gives the derivatives of $\widehat{\boldsymbol{\Psi}}$ with respect to the neural network parameters. The solution can be obtained by solving this auxiliary ODE backward in time [8], but this approach suffers from extreme errors under certain conditions.

The ODE can also be solved by performing multiple forward passes [12, 13], a process that can be made more efficient by using a checkpointing scheme [14]. In this study, due to the small size of the neural networks, the sensitivity is calculated using a forward mode automatic differentiation [15] using the implementation in Julia's ForwardDiff.jl package [16]. Numerical integration was performed using an A-L stable stiffly-accurate 4th order ESDIRK method from Julia's DifferentialEquations.jl library [17]. The neural networks were optimized using an in-house Julia code, which implements a variant of the Levenberg-Marquardt algorithm [18]. The ChemNODE approach used in this study is illustrated in Fig. 1.

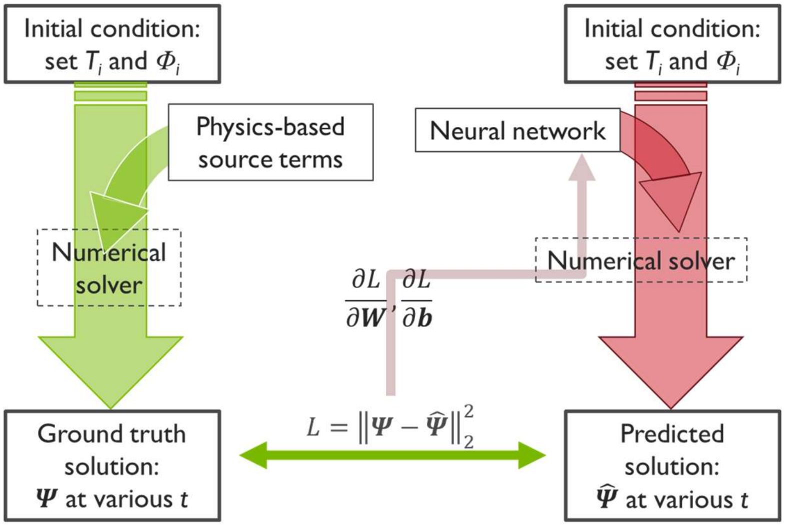

Figure 1. Illustration of machine learning approach used in this study.

Results

As an initial validation study to test the capabilities of ChemNODE to accurately capture chemical kinetics, the problem of a simple homogenous zero-dimensional reactor at constant pressure is considered in this study. This system has no convective and diffusion terms, and is described by:

$$\frac{d\boldsymbol{\Psi}}{dt} = \dot{\boldsymbol{\omega}}_{\Psi}\left(\boldsymbol{\Psi}\right) \tag{4}$$

In this work, $\dot{\boldsymbol{\omega}}_{\Psi}$, which is obtained from chemical mechanisms, is replaced $\widehat{\dot{\boldsymbol{\omega}}}_{\Psi} = \mathcal{N}(\boldsymbol{\Psi}, \boldsymbol{W}, \boldsymbol{b})$ where $\mathcal{N}$ represents a neural network. Separate neural networks, each consisting of a single layer and 10 neurons, were trained for each species. In this study, the results are based on hydrogen-air combustion at 1 atm. The composition space used for training of the neural network involved running the reactor to steady-state. The initial temperature, T_i, was varied between 950 K and 1200 K, while the equivalence ratio was varied from 0.5 to 1.5. The chemical kinetic mechanism used for training comprised of 9 species and 19 chemical reactions [19]. Before training, the species H, HO_2, and H_2O_2 were excluded due to these species having the lowest maximum mass fractions. This led to a thermochemical vector of $\boldsymbol{\Psi} = (T, H_2, O_2, OH, O, H_2O, N_2)^{\boldsymbol{T}}$.

Figures 2—4 show comparisons of the solutions obtained from ChemNODE and those obtained using the full chemistry mechanism. In the figures, the plots on top are for the temperature and reactants, which exhibit monotonic behavior in time. At the bottom, the radicals O and OH, and the product of combustion, H_2O, are shown. The scalars in the top plots are normalized because temperature exists on a different scale from the species, while the plots at the bottom are raw values. In all the plots, the log values of the scalars are displayed. The lines represent the actual solution, while the symbols represent the predicted solution.

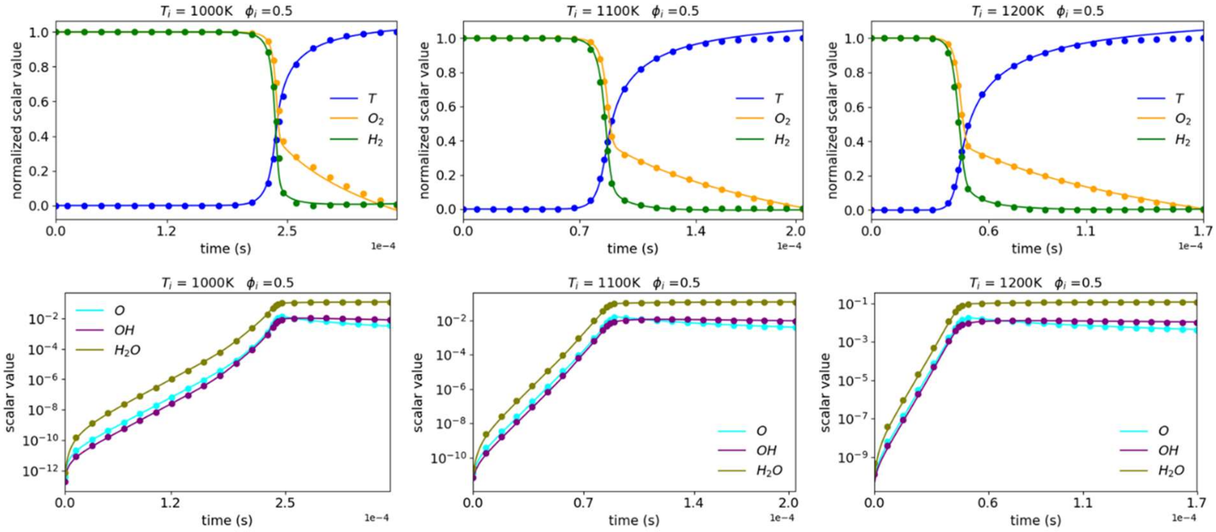

Figure 2. Comparison of ChemNODE and chemical mechanism solutions at $\Phi_i = 0.5$. The lines represent the actual solution, while the symbols represent the predicted solution.

Figure 2 shows the evolution of the chemical species at a lean condition of $\Phi_i = 0.5$, at various initial temperatures, T_i. From the figures, it can be seen that ChemNODE captures the correct behavior under different conditions. The trends for monotonically increasing or decreasing scalars (T, H_2, and O_2) are captured, and so are for the intermediate species. In general, there is a lag between the zone of rapid O_2 and H_2 consumption, with the oxidizer lagging behind the fuel. ChemNODE captures this behavior very well. The most visually discernable error is underprediction in the final temperature, but the maximum error that occurs for temperature is only about 3.0%. Overall, the average mean absolute error normalized by the mean values of the species (AMAE) over the temperatures considered in the figure are 7.14×10^{-4}, 4.72×10^{-3}, 2.18×10^{-3}, 9.14×10^{-3}, 8.51×10^{-3}, and 1.17×10^{-2} for T, H_2, O_2, O, OH, and H_2O, respectively.

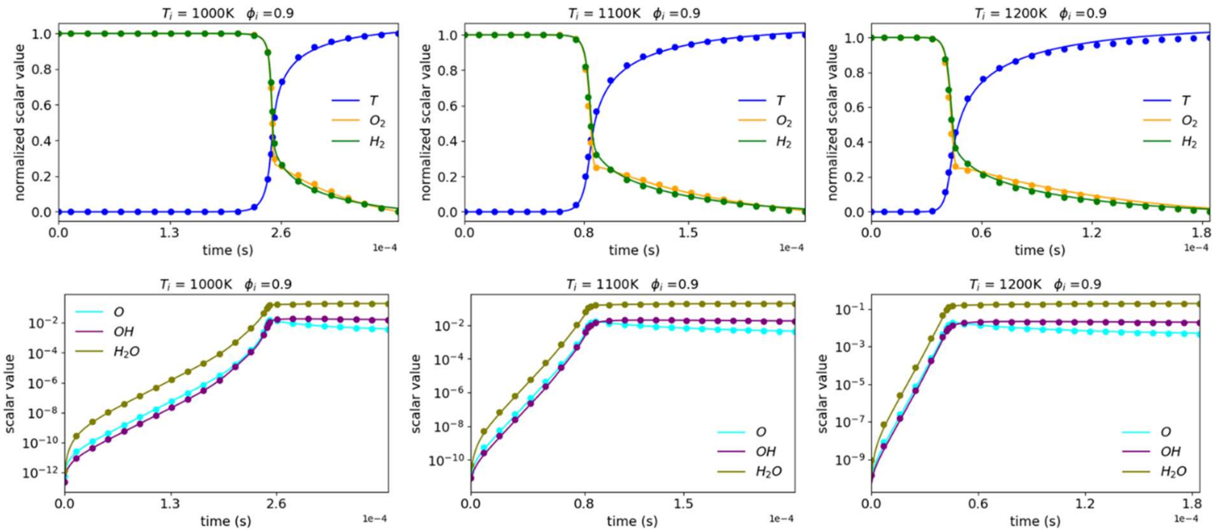

Figure 3. Comparison of ChemNODE and chemical mechanism solution at $\Phi_i = 0.9$. The lines represent the actual solution, while the symbols represent the predicted solution.

Figure 3 shows the evolution of the chemical species at a slightly lean condition of $\Phi_i = 0.9$, at various initial temperatures. Once more, it can be seen that ChemNODE captures the correct behavior. While the final temperature is once more underpredicted by 1.3% on average, the MAE in the prediction of the species is 2.51×10^{-3}, 3.40×10^{-3}, 9.72×10^{-3}, 1.00×10^{-2}, and 1.44×10^{-2} for H_2, O_2, O, OH, and H_2O, respectively. Figure 4 shows the same information as Figs. 2 and 3, but at a rich condition of $\Phi_i = 1.5$. Here, the consumption of the fuel, H_2, lags behind the consumption of the oxidizer. This phenomenon is well-captured by ChemNODE. Overall, the AMAE of the species over the temperatures considered in the figure are 8.70×10-

^{4}, 2.76×10^{-3}, 7.34×10^{-3}, 9.20×10^{-3}, 9.96×10^{-3}, and 1.55 ×10^{-2} for T, H_2, O_2, O, OH, and H_2O, respectively.

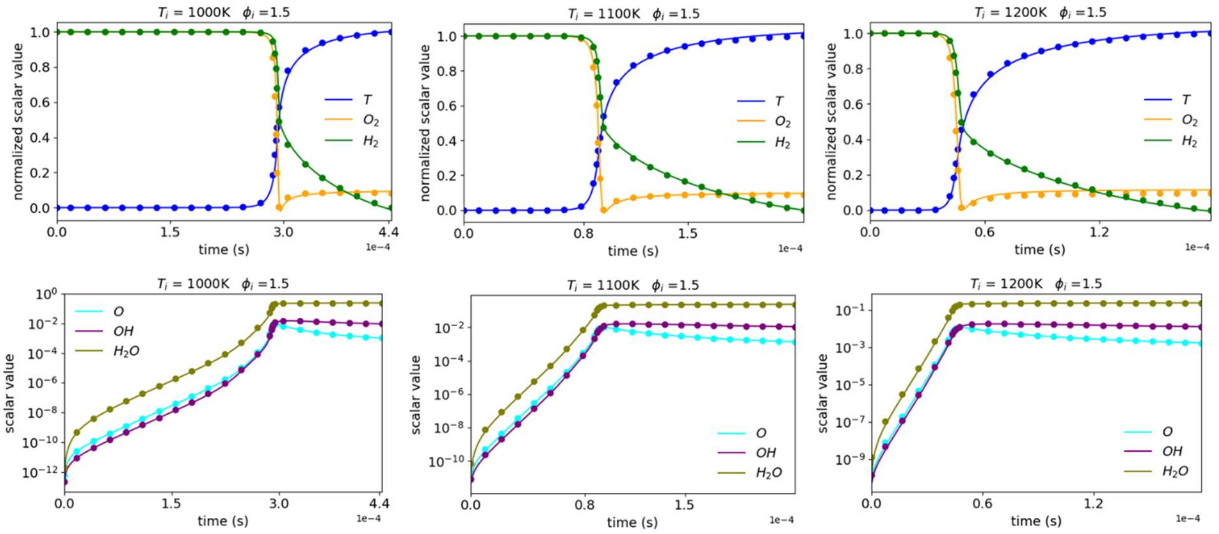

Figure 4. Comparison of ChemNODE and chemical mechanism solution at Φ_i = 1.5. The lines represent the actual solution, while the symbols represent the predicted solution.

Figure 5 shows the ignition delay as a function of the initial equivalence ratio, Φ_i, at different levels of initial temperature, T_i. The red symbols are the actual values while the blue circles are the predicted values. The ignition delay is defined as the time when the maximum rate of temperature increase occurs during combustion. At higher values of T_i, the ignition delay profile has a u-shaped profile with respect to the equivalence ratio. At lower values of T_i, however, the ignition delay increases monotonically with equivalence ratio. These behaviors of ignition delay under different conditions are accurately captured by ChemNODE.

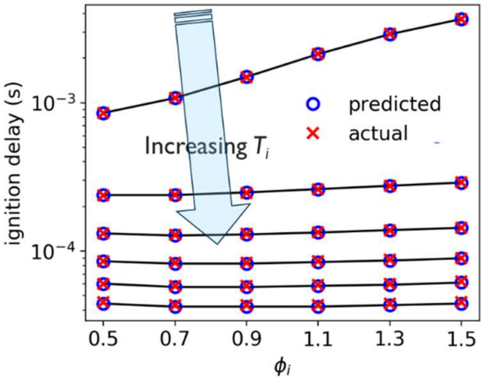

Figure 5. Ignition delay as a function of T_i and Φ_i.

Finally, Fig. 6 shows a comparison of the time taken to advance the solution to steady-state using ChemNODE and the full chemical mechanism with initial conditions selected across various values of T_i and Φ_i. The heights of the bars in the plot represent how long the different methods for calculating source terms take to reach the final time step. Even for a small mechanism involving hydrogen-air combustion used in this study, ChemNODE leads to a speed-up of about 2.3, compared to using the full mechanism that uses the Arrhenius reaction rates. It can, therefore, be expected that for higher hydrocarbon fuels with several species and chemical reactions, much more significant savings can be achieved with the NODE-embedded chemical kinetic solver. Future work will involve validation of this approach for complex hydrocarbon fuels, and demonstration in three-dimensional CFD simulations of internal combustion engines and gas turbine combustors.

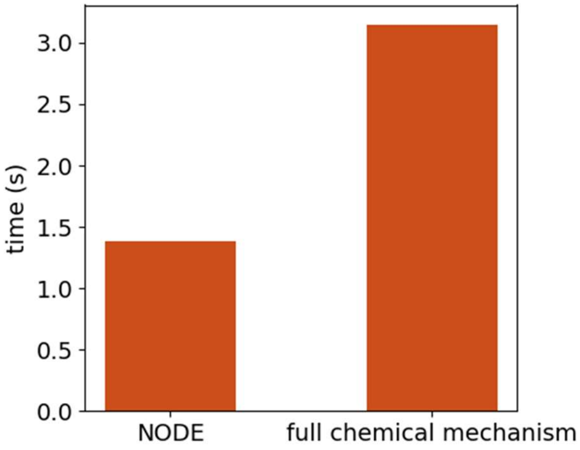

Figure 6. Time taken to reach steady-state by ChemNODE and chemical mechanism.

Conclusion

In this study, a novel neural ordinary differential equation approach to predicting the evolution of chemical species in combustion systems was presented. The approach employs a neural network to learn the appropriate source terms that lead to the correct ODE solution. By calculating the sensitivities of the ODE solution to the neural network parameters, the weights and biases of the neural network were progressively adjusted to obtain an accurate solution. The ChemNODE approach was used to learn the source terms for a zero-dimensional homogeneous constant pressure reactor with hydrogen-air combustion. The results showed that ChemNODE was able to capture the correct time evolution for all species under the conditions considered. It was also shown that the ignition delay, and its variation as a function of initial equivalence ratio and temperature, was well predicted. Lastly, the results demonstrated that ChemNODE was about 2.3 times faster than full hydrogen-air chemical mechanism, indicating its promise for providing even more significant savings if applied to higher hydrocarbon fuels with more complex chemistry and larger kinetic mechanisms. Future work will involve validation of the approach for complex hydrocarbon fuels, and demonstration in three-dimensional CFD simulations of internal combustion engines and gas turbine combustors.

Acknowledgments

The submitted manuscript has been created by UChicago Argonne, LLC, Operator of Argonne National Laboratory (Argonne). The U.S. Government retains for itself, and others acting on its behalf, a paid-up nonexclusive, irrevocable world-wide license in said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the Government. This work was supported by the U.S. Department of Energy, Office of Science under contract DE-AC02-06CH11357. The research work was funded by Argonne's LDRD Innovate project #2020-0203. The authors acknowledge the computing resources available via "Blues," a high-performance computing cluster operated by the Laboratory Computing Resource Center (LCRC) at Argonne National Laboratory.

References

[1] Christo F, Masri A, Nebot E, Pope S. An integrated PDF/neural network approach for simulating turbulent reacting systems. Proc Combust Inst. 1996;26:43-8.

[2] Sen BA, Hawkes ER, Menon S. Large eddy simulation of extinction and reignition with artificial neural networks based chemical kinetics. Combust Flame 2010;157:566-78.

[3] Echekki T, Mirgolbabaei H. Principal component transport in turbulent combustion: A posteriori analysis. Combust Flame 2015;162:1919-33.

[4] Owoyele O, Echekki T. Toward computationally efficient combustion DNS with complex fuels via principal component transport. Combust Theor Modell. 2017:1-29.

[5] Ranade R, Echekki T. A framework for data-based turbulent combustion closure: A posteriori validation. Combust Flame 2019;210:279-91.

[6] Ranade R, Alqahtani S, Farooq A, Echekki T. An ANN based hybrid chemistry framework for complex fuels. Fuel. 2019;241:625-36.

[7] Wan K, Barnaud C, Vervisch L, Domingo P. Chemistry reduction using machine learning trained from non-premixed micro-mixing modeling: Application to DNS of a syngas turbulent oxy-flame with side-wall effects. Combustion and Flame. 2020;220:119-29.

[8] Chen RT, Rubanova Y, Bettencourt J, Duvenaud DK. Neural ordinary differential equations. Advances in neural information processing systems2018. p. 6571-83.

[9] Pope S. Computationally efficient implementation of combustion chemistry using in situ adaptive tabulation. Combustion Theory and Modelling. 1997;1:41-63.

[10] Bezanson J, Edelman A, Karpinski S, Shah VBJSr. Julia: A fresh approach to numerical computing. 2017;59:65-98.

[11] Pontryagin LS, Mishchenko E, Boltyanskii V, Gamkrelidze R. The mathematical theory of optimal processes. 1962.

[12] Andersson JA, Gillis J, Horn G, Rawlings JB, Diehl MJMPC. CasADi: a software framework for nonlinear optimization and optimal control. 2019;11:1-36.

[13] Zhang H, Sandu A. FATODE: a library for forward, adjoint, and tangent linear integration of ODEs. SIAM Journal on Scientific Computing. 2014;36:C504-C23.

[14] Serban R, Hindmarsh AC. CVODES: An ODE solver with sensitivity analysis capabilities. Technical Report UCRL-JP-200039, Lawrence Livermore National Laboratory; 2003.

[15] Rackauckas C, Ma Y, Dixit V, Guo X, Innes M, Revels J, et al. A comparison of automatic differentiation and continuous sensitivity analysis for derivatives of differential equation solutions. 2018.

[16] Revels J, Lubin M, Papamarkou T. Forward-mode automatic differentiation in Julia. arXiv preprint arXiv:07892. 2016.

[17] Rackauckas C, Nie Q. Differentialequations. jl–a performant and feature-rich ecosystem for solving differential equations in julia. Journal of Open Research Software. 2017;5.

[18] Marquardt DW. An algorithm for least-squares estimation of nonlinear parameters. Journal of the society for Industrial and Applied Mathematics. 1963;11:431-41.

[19] Ó Conaire M, Curran HJ, Simmie JM, Pitz WJ, Westbrook CK. A comprehensive modeling study of hydrogen oxidation. International journal of chemical kinetics. 2004;36:603-22.