Fuel 324 (2022) 124560

Contents lists available at ScienceDirect

Fuel

journal homepage: www.elsevier.com/locate/fuel

Full length article

SGD-based optimization in modeling combustion kinetics: Case studies in tuning mechanistic and hybrid kinetic models

Weiqi Ji [a,*], Xingyu Su [b], Bin Pang [c], Yujuan Li [c], Zhuyin Ren [b], Sili Deng [a,*]

[a] Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA, 02139, USA
[b] Center for Combustion Energy, Tsinghua University, Haidian, Beijing, 100084, China
[c] Weichai Power Co. Ltd., Weifang, Shandong, China

ARTICLE INFO

Keywords:
Stochastic gradient descent (SGD)
Data-driven modeling
Chemical kinetics
Hybrid modeling
Auto-differentiation

ABSTRACT

Chemical kinetic modeling is an integral part of combustion simulation, and extensive efforts have been devoted to developing high-fidelity yet computationally affordable models. Despite these efforts, modeling combustion kinetics is still challenging due to the demand for expert knowledge and high dimensional optimization against experiments. Therefore, data-driven approaches that enable efficient discovery and calibration of kinetic models have received much attention in recent years, the core of which is the high-dimensional optimization based on big data. Evolutionary algorithms are usually adopted for optimizing chemical kinetic models, although they usually suffer from high computational costs and are limited to a small number of parameters. Meanwhile, gradient-based optimizations, especially the stochastic gradient descent (SGD) methods, have shown success in developing complex models by training large-scale deep learning models. Therefore, this work explores the applications of SGD-based optimizations in tuning mechanistic kinetic models and learning hybrid kinetic models. We first showed that SGD-based optimizations could substantially save computational cost compared to evolutionary algorithms when the number of kinetic parameters in mechanistic models reached about one hundred. We then demonstrated that the SGD-based optimization enabled us to use a neural network model to represent the pyrolysis of the Hybrid Chemistry and optimize the associated hundreds of weights in the neural network. These proof-of-concept studies showed that the SGD-based optimization is more efficient than evolutionary algorithms, is a promising approach for developing chemical kinetic models with high dimensional parameters, and is capable of developing hybrid mechanistic-machine learning kinetic models.

1. Introduction

The optimization of model parameters plays a critical role in the development of chemical kinetic models. The widely adopted optimization techniques in combustion kinetic modeling can be categorized into heuristic algorithms and response surface techniques [1,2]. Heuristic algorithms [3–8] such as genetic algorithms usually perform well with less than 100 parameters and small datasets. This is due to fact that the genetic algorithms require a large number of samples to explore the parameter space, and the computational cost of genetic algorithms scales with the number of parameters and number of samples in the dataset. Response surface techniques [9] alleviate the intensive computational cost in performing target simulations with the proposed kinetic models by building a function approximation that maps the model parameter space to model predictions. Similar to heuristic algorithms, however, response surface techniques are also limited to low dimensional model parameter space [10] and small datasets, for the cost of building a response surface scales with the dimension of parameter space and the number of quantities of interest. Meanwhile, recent development in diagnostic techniques and experiment automation [11] have significantly increased the efficiency of experimental data generation and driven combustion research into the big data regime. Therefore, optimization methods that can handle large dataset effectively and efficiently would greatly benefit the development of chemical kinetic models in addition to high-throughput experimentation.

While seldom exploited in combustion modeling, optimization algorithms based on stochastic gradient descent (SGD) has shown promise in nonconvex optimization for complex nonlinear models, and SGD has played a central role in driving the boom of deep learning in the last decade [12]. Furthermore, various techniques have been developed in conjunction with SGD to increase the generalization performance of the optimized model. For instance, a modern deep learning optimizer

* Corresponding authors.
E-mail addresses: weiqiji@mit.edu (W. Ji), silideng@mit.edu (S. Deng).

https://doi.org/10.1016/j.fuel.2022.124560
Received 13 December 2021; Received in revised form 16 April 2022; Accepted 7 May 2022
Available online 25 May 2022

not only focuses on minimizing the loss functions but also regularizes the model to increase the extrapolation capability. Generalization to different conditions and tasks is an important feature for classical physics-based chemical models. With good generalization capability, a chemical model that was developed based on canonical combustion experiments should also work reasonably well in simulating the reactions in practical combustion systems. Therefore, SGD is potentially not only more efficient but also more generalizable compared to heuristic optimization algorithms.

One of the major obstacles for exploiting SGD in combustion modeling is the lack of software ecosystems that can efficiently and accurately compute the gradient of simulation output to model parameters. For instance, the finite difference method (often termed the brute-force method) usually suffers from both computational inefficiency, as the cost scales with the number of parameters, and inaccuracy due to truncation error. Conversely, stochastic gradient descent based on auto-differentiation (AD) has shown both efficiency and accuracy in the training of large-scale deep neural network models [12]. Many open-source AD packages have been developed in the last decade, including TensorFlow [13], Jax [14] backed by Google, PyTorch [15] backed by Facebook, ForwardDiff.jl [16] and Zygote.jl [17] in Julia. Fortunately, an open-source AD-powered differentiable combustion simulation package Arrhenius.jl [18] has been developed recently to enable differentiable programming in combustion modeling. The package incorporates combustion physics models into AD ecosystems in Julia and thus enable auto-differentiation across combustion models, such as computing the gradient of species concentrations with respect to the kinetic parameters. This work thus employs Arrhenius.jl to explore the opportunities of SGD-based optimization in modeling complex combustion kinetics.

In this work, we explore the capability of SGD-based optimization in two optimization tasks: optimizing a mechanistic kinetic model and learning a hybrid neural-mechanistic kinetic model. Optimization of a mechanistic model is usually an integral part of calibrating kinetic models against experimental data [6] or a reference model. For example, optimizing an empirical kinetic model or an overly reduced kinetic model against a detailed kinetic model [7,19]. It proceeds by tuning the Arrhenius parameters to enable the predictions of the optimized kinetic model to agree with the target detailed kinetic model.

For the hybrid kinetic model [20], a neural network model is used to represent part of the kinetic process, e.g., the pyrolysis of large hydrocarbon fuels, and thus one can train a neural network model to predict the combustion process without providing a reaction template based on expert knowledge. When simulating combustion processes, the neural network takes the species concentrations and temperature as inputs and predicts the production rate of each species, similar to a mechanistic model. Learning a hybrid neural network/mechanistic model is a relatively new adventure since it is computationally infeasible to train the neural network model using conventional evolutionary optimization algorithms. While SGD-based optimization should be able to handle the computational load, the training is not trivial. Since the neural network is coupled with the mechanistic model, one has to take the mechanistic model into the gradient calculation as well. Previous work [20] has assumed that all intermediate species are measurable and thus one can decouple the neural network model from the mechanistic model during training. However, this assumption often does not hold due to the complexity of the chemical kinetics and limited diagnostic capability. Here we adopt the Arrhenius.jl to enable computing the gradient for the hybrid model and explore the possibility of learning the hybrid model using SGD-based optimization.

This paper is structured as follows: we shall first briefly introduce the numerical approaches in Section 2, including the package of Arrhenius.jl Section 2.1 and the equations for various gradient calculations in Section 2.2. We then present the application in optimizing mechanistic kinetic model and hybrid neural network/mechanistic kinetic model in Section 3. Finally, we draw conclusions in Section 4.

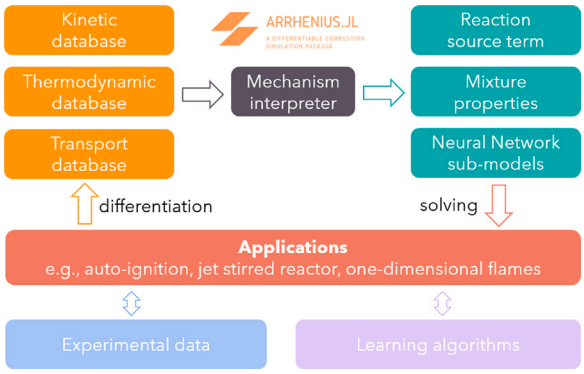

Fig. 1. Schematic showing the structure of the Arrhenius.jl package.

2. Numerical approaches

2.1. Arrhenius.jl

Arrhenius.jl [18] is built in the programming language of Julia to leverage the rich ecosystems of auto-differentiation and differential equation solvers. Arrhenius.jl does two types of differentiable programming: (i) it can differentiate elemental computational blocks. For example, it can differentiate the reaction source term with respect to kinetic and thermodynamic parameters as well as species concentrations. Arrhenius.jl leverages the language-wide auto-differentiation of Julia programming language to do the auto-differentiation and thus preclude the need of deriving the formula of the gradient/Jacobian matrix. (ii) It can also differentiate the numerical solvers in various ways, such as solving the continuous sensitivity equations [21] as done in CHEMKIN [22] and Cantera [23] and in adjoint methods [24,25]. An example of this type of differentiation is computing the gradients of species concentrations with respect to chemical kinetic model parameters and initial compositions. The first type of differentiation is usually the basis of the second type of higher-level differentiation. Arrhenius.jl offers the core functionality of combustion simulations in native Julia programming, such that users can conveniently build applications on top of Arrhenius.jl and exploit various approaches to do high-level differentiation.

Fig. 1 shows a schematic of the structure of the Arrhenius.jl package. Arrhenius.jl reads in the chemical mechanism files in YAML format maintained by the Cantera [23] community; the chemical mechanism files contain the kinetic model, thermodynamic, and transport databases. The core functionality of Arrhenius.jl is to compute the reaction source terms and mixture properties, such as heat capacities, enthalpies, entropies, Gibbs free energies, etc. In addition, Arrhenius.jl offers flexible interfaces for users to define neural network models as submodels and augment them with existing physical models. For example, one can use a neural network submodel to represent unknown reaction pathways and exploit various scientific machine learning methods to train the neural network models, such as neural ordinary differential equations [26–29] and physics-informed neural network models [30,31]. One can then implement the governing equations for different applications with these core functionalities and solve the governing equations using classical numerical methods or neural-network-based solvers, such as physics-informed neural networks [30]. Arrhenius.jl provides solvers for canonical combustion problems, such as simulating the auto-ignition in constant volume/pressure reactors and oxidation in jet-stirred reactors. The governing equations implemented in the package is generally following those of CHEMKIN [22] and Cantera [23]. The package has also been validated against Cantera [23] in various canonical programs, such as pyrolysis, ignition, and sensitivity analysis of one-dimensional flame.

In contrast to legacy combustion simulation packages, Arrhenius.jl can not only provide predictions given the physical models but also optimize model parameters given experimental measurements. By efficiently and accurately evaluating the gradient of the solution outputs to the model parameters, experimental data can be incorporated into the simulation pipeline to enable data-driven modeling with deep learning algorithms. Note that one can also achieve differentiable capability with existing combustion software, such as CHEMKIN [22] and Cantera [23]. For example, ADIFOR [32] developed in 1970s can be employed to do auto-differentiation over subroutines in CHEMKIN. However, modern optimization and machine learning packages are usually in high-level languages (e.g., Python and Julia [33]), and thus one has to deal with the compatibility issues among modern high-level languages and low-level languages (e.g., C++ and FORTRAN). On the contrary, Arrhenius.jl can conveniently connecting combustion modeling with optimization and machine learning without introducing the complexity of low-level languages.

2.2. Gradient calculation

In this section, we briefly discuss how Arrhenius.jl can enhance various approaches for computing the gradient (sensitivity) in jet-stirred reactors, shock tubes, and laminar flame experiments. In general, we deal with two kinds of gradient calculations [22], i.e., steady-state solutions and transient solutions.

Examples of steady-state solutions are the modeling of species profiles in jet-stirred reactors and steady laminar flames. Without loss of generality, we can write down the governing equations in vector form as

$$F(\phi(\alpha);\alpha) = 0, \tag{1}$$

where F corresponds to the residual vector, ϕ corresponds to the solution vector, and α corresponds to model parameters, such as the kinetic, thermodynamic, and transport parameters. By differentiating Eq. (1) with respect to the α, we obtain matrix equations for the gradients.

$$\frac{\partial F}{\partial \phi}\frac{\partial \phi}{\partial \alpha} + \frac{\partial F}{\partial \alpha} = 0, \tag{2}$$

where $\frac{\partial \phi}{\partial \alpha}$ are the Jacobian matrices of the solution vector with respect to model parameters. Normally, for optimization, we have a scalar loss function defined as $L(\phi)$. The gradient of the loss function with respect to model parameters can be readily achieved via

$$\frac{\partial L}{\partial \alpha} = \frac{\partial L}{\partial \phi}\frac{\partial \phi}{\partial \alpha}. \tag{3}$$

With Arrhenius.jl, we can leverage AD to compute the two Jacobian matrices of $\frac{\partial F}{\partial \phi}$ and $\frac{\partial F}{\partial \alpha}$. Unlike packages for calculating the analytical Jacobian [34], efficient computation of the Jacobian can be achieved without developing the analytical form. In addition, general AD enables us to differentiate over neural network submodels, which is difficult to implement using analytical approaches. If the solution variables are discretized in a computational domain, e.g., the one-dimensional freely propagating flame, one can readily leverage multi-threading to evaluate the two Jacobians without complex code re-factorization using parallel computing.

Examples of transient solutions are auto-ignition and fuel pyrolysis in shock tubes. The governing equations are in the general form of

$$\frac{d\phi}{dt} = F(\phi, t; \alpha), \tag{4}$$

where t is the time. A natural approach to compute the gradient $W = \frac{\partial \phi}{\partial \alpha}$ is solving the governing equations of W, i.e.,

$$\frac{dW}{dt} = \frac{\partial F}{\partial \phi}W + \frac{\partial F}{\partial \alpha}. \tag{5}$$

In addition to solving Eq. (5), Arrhenius.jl leverages various adjoint sensitivity algorithms provided in DifferentialEquations.jl. For comprehensive comparisons of various algorithms in computing the gradient of solution variables for transient solutions, readers shall consult [24].

In general, calculating the gradient for transient solutions involving stiff chemical kinetic models is expensive, as computation time usually scales with both the number of species and the number of parameters. Meanwhile, global combustion behaviors, such as ignition delay times (IDTs), are usually more experimentally accessible compared to measurements of concentration profiles. Recent work [21,35–37] has substantially advanced the algorithms for computing the gradient of IDT. Therefore, the current work employs the sensBVP method [35], which converts the initial value problem (IVP) to a boundary value problem (BVP) by treating the temperature at ignition as a boundary condition and the IDT as a free variable to solve. In other words, the problem is converted from a transient problem to a steady-state problem, similar to the solution of a one-dimensional freely propagating flame.

3. Results and discussion

We now present case studies using SGD-based optimization for optimizing complex combustion kinetic modelings where the dimension of parameters is too high for traditional optimization algorithms, such as genetic algorithms.

3.1. Optimizing mechanistic kinetic model

Most skeletal chemical models are obtained by removing unimportant species and the associated reactions from the master model, leaving the kinetic parameters of the remaining pathways unchanged after the reduction. Various systematic mechanism reduction approaches have been developed to generate skeletal mechanism [38–40]. There has also been increasing interest in optimizing the kinetic parameters in overly reduced reaction models to compensate for the error introduced by over-reduction [7,19]. Regardless of the controversy of tuning the kinetic parameters during optimization, this approach has the potential to produce a smaller model with higher fidelity than the ones obtained via traditional reduction methods. The reduction-optimization approach proceeds by sampling from target conditions and then optimizing the reduced model to achieve similar predictions as the master mechanism under all sampled conditions. This work thus takes this optimization process as a case study for SGD-based optimization, i.e., using SGD-based optimization to minimize the discrepancies between reduced model and original detailed mechanism.

We demonstrate SGD-based optimization in reducing and optimizing two chemical models for natural gas and *n*-heptane, the master models of which are the GRI3.0 mechanism [41] and the Nordin1998 mechanism [42], respectively. As previously discussed, overly reduced models using classical reduction approaches with a large threshold or intuition are firstly obtained, with the number of species in GRI3.0 reduced from 53 to 23, and that in Nordin1998 reduced from 41 to 34. The reduction is targeted for simulating natural gas engines and diesel engines with the commercial software of Converge [43]. For GRI3.0, we first removed the following species using an iterative reduction involving DRG [44], DRGEP [45], PFA [40], and sensitivity analysis: C, CH_3OH, C_2H, C_2H_2, HCCO, CH_2CO, HCCOH, NH, NH_2, NH_3, N_2O, HNO, CN, HCN, H_2CN, HCNN, HCNO, HOCN, HNCO, NCO, Ar, and CH_2CHO. We then further removed the species of CH_3CHO, NO_2, NO, NNH, N, C_2H_3, and CH_2OH, CH by removing NO-chemistry for Converge has a built-in NO module and following the reduction of DRM19 [46]. The overly reduced model is denoted as SK23 with 23 species. For Nordin1998, the following species were removed based on the reduction of GRI3.0: C_3H_5, C_3H_4, C_2H_6, CH_4O_2, CH_3O_2, CH_3O, and C_2H_2. The overly reduced model is denoted as SK34 with 34 species. Note that the way to produce an overly reduced model can be regarded

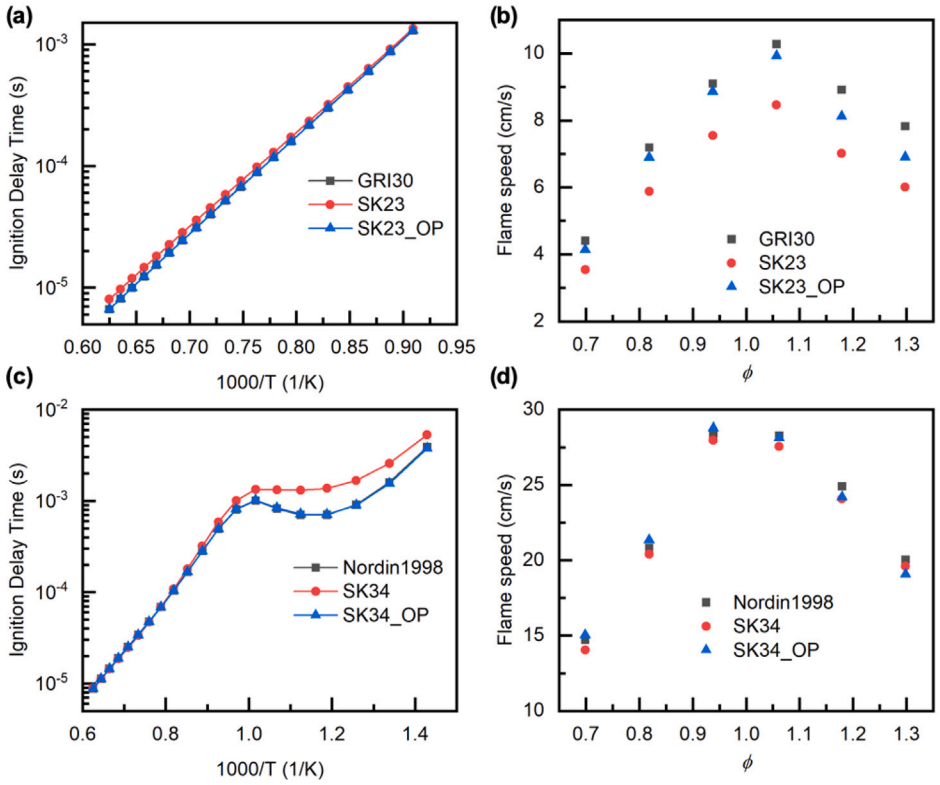

Fig. 2. Predicted ignition delay times and flame speeds for representative cases: (a-b) the mixture of natural gas/air using the master mechanism GRI3.0, skeletal mechanism SK23 and optimized SK23_OP. The ignition delay time was simulated using the fuel composition of CH_4 : C_2H_4 : C_3H_8 = 0.85 : 0.1 : 0.05 by volume, pressure of 40 atm, equivalence ratio of 0.9. The flame speed was simulated at 40 atm and 300 K, and (c-d) the mixture of *n*-heptane/air using the master mechanism Nordin1998, skeletal mechanism SK34 and optimized SK34_OP. The ignition delay time was simulated at pressure of 40 atm, equivalence ratio of 1.2. The flame speed was simulated at 40 atm and 500 K.

as a hyper-parameters subject to explore. This work focuses on the demonstration of the optimization algorithms, and we leave the optimal choice of removed species to future studies. It should be noted that one can also optimize an existing empirical semi-global reaction model against a detailed model without consulting the skeletal mechanism reduction.

The kinetic parameters of these overly reduced models were subsequently optimized to retain the predictability of the master models. The ignition delay times and laminar flame speeds (SLs) are utilized as performance metrics to validate the overly reduced and optimized models against the corresponding master models. As shown in Fig. 2, for GRI3.0, SK23 over-predicts IDT at high temperatures and under-predicts SL at all equivalence ratios. For Nordin1998, SK34 significantly over-predicts the IDT at low temperatures, especially within the negative temperature coefficient region, while the flame speeds are hardly affected by the reduction.

The optimization of the kinetic parameters of the over-reduced models was conducted on all three Arrhenius parameters, namely, A, b, E_a. Although both IDT and SL could be selected as targets for optimization, we only utilized the IDT for its relatively lower computational cost than SL. Moreover, the top ten reactions selected based on the sensitivity analysis for the SL were excluded from the optimization, such that the optimization will not change these key reactions for SL. We then randomly sampled 500 initial conditions covering a wide range of mixture compositions and thermodynamic states for training. For GRI3.0, the range of the pressure, initial temperature, and equivalence ratios are 1–60 atm, 1100–2000 K, and 0.5–1.8, respectively, and the fuel composition is set as CH_4 : C_2H_4 : C_3H_8 = 0.85 : 0.1 : 0.05 by volume. Similarly, for Nordin1998, the ranges of the pressure, initial temperature, and equivalence ratio are 1–60 atm, 850–1800 K, and 0.5–1.5, respectively.

The datasets were split into training and validation datasets with a ratio of 70:30. During each parameter update, one case was randomly sampled to evaluate its IDT, and this process can be viewed as mini-batching with the batch size of one. Instead of optimizing the Arrhenius parameters directly, we optimized the relative changes of Arrhenius parameters compared to their nominal values, i.e.,

$$p = [ln(A/A_0), b - b_0, Ea - Ea_0], \tag{6}$$

where the subscript $_0$ refers to the base model. The units of Ea are specified as cal/mol as we tried to minimize the changes in Ea. However, if one wants to ensure the change in Ea is relatively comparable to the change in A, one may specify the unit of Ea as kcal/mol, as a change of 2 kcal/mol in Ea is close to e times of change in A, such that the changes in A and Ea will be balanced and avoid stiffness in the parameter space.

The loss function was defined as the mean square error (MSE) between the predicted IDTs in the logarithmic scale using the reduced model and the master model:

$$Loss = MSE\left(log(IDT^{sk}), log(IDT^{master}).\right) \tag{7}$$

The gradients of IDT to kinetic parameters were computed using the sensBVP method proposed in [35]. The Adam [47] optimizer with the default learning rate of 0.001 was adopted. Weight decaying and early stopping were employed to regularize the parameters, such that the optimization prefers kinetic parameters that are close to their original values. Fig. 3 shows the training history of the loss function and the L_2-norm of the model parameters for the Nordin1998 model. We trained the reduced model for 100 epochs and stopped the training when the loss function as well as the model parameters reached a plateau.

The performance of the optimized skeletal mechanisms is also shown in Fig. 2. For the IDT, as targeted in the optimization, the optimized models agree with the master models very well for both two fuels. One interesting observation is that the optimized models also work well for SL, although SL is not targeted for optimization. This

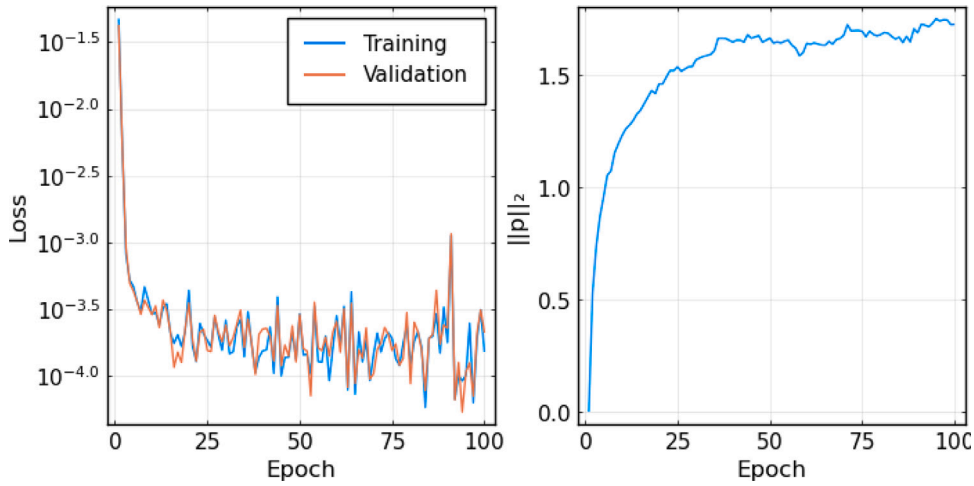

Fig. 3. Training history of loss functions, L_2-norm of model parameters. Regularization settings: weight-decay of 1e-4 with a learning rate of 1e-3.

could be attributed to several reasons. For GRI3.0, the optimization compensates for the errors in the high-temperature chemistry seen in the predicted IDT, and those re-calibrated high-temperature chemical pathways lead to accurate predictions of SL. For Nordin1998, the skeletal models already accurately predict the IDT at high temperatures as well as the SL, and the optimization does not degrade the prediction of SL thanks to the regularization. It is worth noting that the optimized SK23 still under-predicts SL at fuel-rich conditions, potentially because some of the sensitive reactions for fuel-rich conditions are not fixed during optimization; further refinement of the fixed reactions is therefore suggested. Furthermore, the optimization is computationally efficient. Qualitatively speaking, previously employed genetic algorithms have to be performed on clusters [19], while the current work was performed on an ordinary workstation within an hour. In summary, these two case studies demonstrate the ability of SGD-based optimization to optimize complex reaction models with high accuracy, good generalization capability, and high efficiency. Such optimization capability will help augment current mechanism reduction techniques.

3.2. Optimizing hybrid neural network/mechanistic kinetic models

We then explore the optimization of hybrid neural network/mechanistic kinetic model to develop a neural-network-based pyrolysis submodel [20] within the HyChem model framework [48]. Our recently developed Chemical Reaction Neural Network (CRNN) [28] approach was employed to develop the neural network model for its interpretability, such that the learned model complies with fundamental physical laws and provides chemical insights, as well as its compatibility with large-scale combustion simulation package/software. The conventional HyChem-based pyrolysis submodels require expert knowledge on the chemical kinetics which takes years to develop. On the contrary, the CRNN approach aims to autonomously discover the reaction pathways and kinetic parameters simultaneously to accelerate high-fidelity chemical model development. In the following demonstration, the CRNN-HyChem approach was utilized to model the jet fuel of JP10.

As shown in Fig. 4, the CRNN-HyChem approach models the fuel chemistry of JP10 with two submodels, similar to the original HyChem concept. The CRNN submodel models the breakdown of JP10 fuel molecules into smaller hydrocarbons up to C_6H_6, and the submodel for C_0-C_6 describes the oxidation chemistry. For proof-of-concept, we chose the same species in the original HyChem pyrolysis submodel [48] to be included in the CRNN model; however, it should be noted that the such chosen species can be treated as hyper-parameters to circumvent the need for expert knowledge and achieve potentially better performance. In the original CRNN approach [28], the Law of Mass Action and

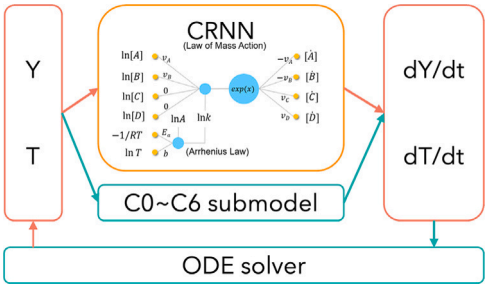

Fig. 4. Schematic showing the structure of the CRNN-HyChem approach.

Arrhenius Law are enforced by the design of the structure of the neural network. Reaction orders are assumed to be equal to the stoichiometric coefficients for the reactants. In the present study, elemental conservation is further guaranteed by projecting the stoichiometric coefficients into the elemental conservation space. For better convergence, the stoichiometric coefficients for JP10 are fixed as -1, and during the training, the stoichiometric coefficients are regularized to achieve better numerical stability. The training data were generated by simulating the IDT using the original JP10 HyChem model. A wide range of thermodynamic conditions were considered: pressures of 1–60 atm, initial temperatures of 1100–1800 K, and equivalence ratios of 0.5–1.5. In total, 500 thermodynamic conditions were randomly generated using the Latin hypercube sampling (LHS) method. The dataset was split into training and validation datasets with a ratio of 70:30, respectively.

The learned stoichiometric coefficients and kinetic parameters are shown in Table 1, where negative and positive stoichiometric coefficients correspond to reactants and products, respectively. Qualitatively, most of the learned pathways are H-abstraction reactions, which is consistent with the expert-derived HyChem models. However, quantitatively, the learned pathways are not the same as those in the HyChem model, and further efforts will be directed to extracting physical insights from the learned pathways. Fig. 5 compares the results of the learned CRNN model and the IDTs generated using the original HyChem model [48], and they agree very well. The results thus demonstrate the capability of SGD-based optimization in learning CRNN models with hundreds of parameters, which is computationally challenging with evolutionary algorithms. With the increasing demand for rapidly developed kinetic models for new renewable fuels for screening and fuel design, hybrid neural network/mechanistic modeling provides an elegant approach to autonomously derive kinetic models from experimental data. SGD-based optimization will play

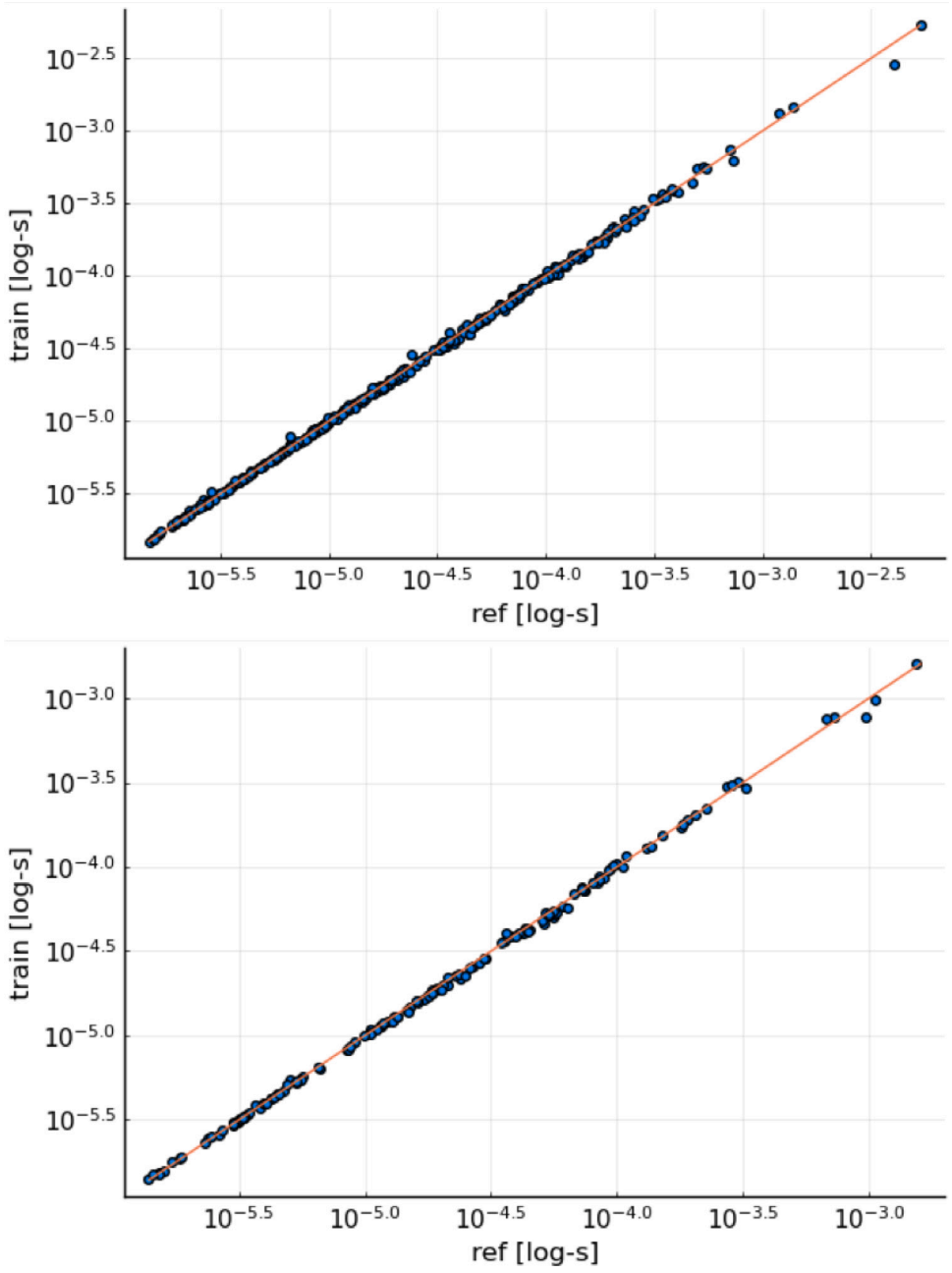

Fig. 5. Comparisons between the predicted ignition delay times using the learned CRNN model (Y-axis) and the HyChem model (X-axis) for both the training and validation datasets.

a vital role in enabling such kind of autonomous model discovery algorithms. While this manuscript focuses on learning the reaction pathways from scratch, one can also develop a data-driven model based on existing kinetic models for similar fuels and utilize SGD-based optimization for the training, as demonstrated in [49].

4. Conclusions

This work explores SGD-based optimization for optimizing complex chemical kinetic models, including both mechanistic kinetic models and hybrid neural network/mechanistic kinetic models. The results show that SGD-based optimization requires significantly less computational resources compared to traditional evolutionary optimization approaches for hundreds of model parameters. Furthermore, SGD-based optimization enables us to augment neural network models to represent the unknown reaction pathways and optimize the involved hundreds of parameters, which is computationally intractable with traditional genetic algorithms.

We expect that SGD-based optimization could greatly facilitate the integration of modern deep learning techniques into combustion modeling, especially the physics-informed machine learning [50] that takes the advantage of both physics-based and data-driven modeling. This work is proof-of-concept study to take advantage of SGD-based optimization for optimizing combustion models and only deterministic forward problems are presented. There are a lot of opportunities and open challenges to extend the approaches to inverse problems and design of experiments using Bayesian inference which also heavily relies on differential programming and SGD-based optimization [51,52].

Table 1
Learned stoichiometric coefficients and kinetic parameters. Reaction orders are assumed to be equal to the stoichiometric coefficients for reactants.

Stoichiometric Coefficients														Ea	b	ln (A)
C10H16	H	CH3	OH	HO2	O	CH4	C2H4	C3H6	C5H6	C6H6	H2O2	O2	H2O	(kcal/mol)		
-1	0.11	0.32	0.06	-0.08	-0.12	0.44	0.42	0.63	0.6	0.59	0.12	-0.11	0.18	24.52	-0.29	15.33
-1	0.12	0.33	0.1	0.08	-0.16	0.46	0.43	0.65	0.6	0.57	-0.09	-0.07	0.23	19.93	-0.1	16.06
-1	0.33	0.26	-0.09	-0.02	-0.18	0.5	0.45	0.7	0.6	0.54	-0.12	0.16	0.22	17.63	0.07	18.48
-1	0.1	0.31	-0.09	0.03	-0.24	0.44	0.42	0.63	0.61	0.58	0.14	-0.11	0.2	19.04	-0.26	17.21
-1	0.14	0.3	0.1	-0.14	-0.23	0.44	0.44	0.58	0.6	0.61	0.16	-0.07	0.23	17.09	-0.05	17.2
-1	0.35	0	0	-0.17	0.01	0.55	0.77	0.65	0.37	0.68	-0.07	0.12	0.23	14.45	0.12	18.64
-1	0.14	0.34	-0.02	0.06	-0.15	0.46	0.43	0.66	0.6	0.56	-0.07	-0.02	0.23	19.10	-0.19	16.3
-1	0.2	0.21	0.08	-0.18	0.05	0.53	0.54	0.64	0.5	0.63	-0.05	0.04	0.26	16.41	-0.08	17.37
-1	0.1	0.3	-0.16	0.04	-0.29	0.44	0.41	0.63	0.61	0.58	0.17	-0.09	0.22	21.74	-0.1	17.72
-1	0.52	0.28	0.01	0	0	0.76	0.39	0.29	0.98	0.4	-0.23	0.11	0.23	12.98	0	20.09

CRediT authorship contribution statement

Weiqi Ji: Conceptualization, Methodology, Software, Writing – original draft. **Xingyu Su:** Conceptualization, Methodology, Software, Data curation. **Bin Pang:** Conceptualization, Reviewing. **Yujuan Li:** Conceptualization, Reviewing. **Zhuyin Ren:** Supervision, Writing – review & editing. **Sili Deng:** Supervision, Conceptualization, Investigation, Reviewing and editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

WJ and SD would like to acknowledge the funding support by Weichai Power Co., Ltd. ZR and XS would like to acknowledge the support from National Natural Science Foundation of China No. 52025062. WJ would like to thank Dr. Ji-Woong Park for fruitful discussions on the deep mechanism reduction, Dr. Vyaas Gururajan on the implementation of the sensBVP method, Dr. Travis Sikes for discussions on mechanism optimization, Dr. Christopher Rackauckas on the usages of DifferentialEquations.jl.

References

[1] Frenklach M, Wang H, Rabinowitz MJ. Optimization and analysis of large chemical kinetic mechanisms using the solution mapping method—combustion of methane. Prog Energy Combust Sci 1992;18(1):47–73.
[2] Sheen DA, Wang H. Combustion kinetic modeling using multispecies time histories in shock-tube oxidation of heptane. Combust Flame 2011;158(4):645–56.
[3] Rein G, Lautenberger C, Fernandezpello A, Torero J, Urban D. Application of genetic algorithms and thermogravimetry to determine the kinetics of polyurethane foam in smoldering combustion. Combust Flame 2006;146(1–2):95–108.
[4] Bertolino A, Fürst M, Stagni A, Frassoldati A, Pelucchi M, Cavallotti C, et al. An evolutionary, data-driven approach for mechanism optimization: Theory and application to ammonia combustion. Combust Flame 2021;229:111366.
[5] Elliott L, Ingham D, Kyne A, Mera N, Pourkashanian M, Wilson C. Genetic algorithms for optimisation of chemical kinetics reaction mechanisms. Prog Energy Combust Sci 2004;30(3):297–328.
[6] Ryu JI, Kim K, Min K, Scarcelli R, Som S, Kim KS, et al. Data-driven chemical kinetic reaction mechanism for F-24 jet fuel ignition. Fuel 2021;290:119508.
[7] Mittal A, Wijeyakulasuriya SD, Probst D, Banerjee S, Finney CEA, Edwards KD, et al. Multi-dimensional computational combustion of highly dilute, premixed spark-ignited opposed-piston gasoline engine using direct chemistry with a new primary reference fuel mechanism. In: Internal combustion engine division fall technical conference, vol. 2. 2017, V002T06A022.
[8] Cailler M, Darabiha N, Fiorina B. Development of a virtual optimized chemistry method. Application to hydrocarbon/air combustion. Combust Flame 2020;211:281–302.
[9] Sheen DA, Wang H. The method of uncertainty quantification and minimization using polynomial chaos expansions. Combust Flame 2011;158(12):2358–74.
[10] Ji W, Wang J, Zahm O, Marzouk YM, Yang B, Ren Z, et al. Shared low-dimensional subspaces for propagating kinetic uncertainty to multiple outputs. Combust Flame 2018;190:146–57.
[11] Tao M, Lynch PT, Zhao P. Kinetic modeling of ignition in miniature shock tube. Proc Combust Inst 2019;37(1):593–601.
[12] Bengio Y, Goodfellow I, Courville A. Deep learning, vol. 1. Cambridge, MA: MIT Press; 2017.
[13] Yu Y, Abadi M, Barham P, Brevdo E, Burrows M, Davis A, et al. Dynamic control flow in large-scale machine learning. In: Proceedings of the thirteenth EuroSys conference. ACM; 2018, p. 265–83.
[14] Bradbury J, Frostig R, Hawkins P, Johnson MJ, Leary C, Maclaurin D, et al. JAX: Composable transformations of Python+NumPy programs. 2018, http://github.com/google/jax.
[15] Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, et al. Pytorch: An imperative style, high-performance deep learning library. 2019, arXiv:1912.01703.
[16] Revels J, Lubin M, Papamarkou T. Forward-mode automatic differentiation in julia. 2016, arXiv:1607.07892.
[17] Innes M. Don't unroll adjoint: Differentiating SSA-form programs. 2018, arXiv:1810.07951.
[18] Ji W, Deng S. Arrhenius.jl: A differentiable combustion simulation package. GitHub; 2021, https://github.com/DENG-MIT/Arrhenius.jl.
[19] Kelly M, Bourque G, Dooley S. Toward machine learned highly reduce kinetic models for methane/air combustion. 2021, arXiv:2103.08377.
[20] Ranade R, Alqahtani S, Farooq A, Echekki T. An ANN based hybrid chemistry framework for complex fuels. Fuel 2019;241:625–36.
[21] Ji W, Ren Z, Law CK. Evolution of sensitivity directions during autoignition. Proc Combust Inst 2019;37(1):807–15.
[22] Kee RJ, Rupley FM, Miller JA. Chemkin-II: A Fortran chemical kinetics package for the analysis of gas-phase chemical kinetics. Sandia National Lab., Livermore, CA; 1989.
[23] Goodwin DG, Speth RL, Moffat HK, Weber BW. Cantera: An object-oriented software toolkit for chemical kinetics, thermodynamics, and transport processes. 2021, Version 2.5.1, http://dx.doi.org/10.5281/zenodo.4527812, https://www.cantera.org.
[24] 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, arXiv:1812.01892.
[25] Rackauckas C, Nie Q. Differentialequations.jl–a performant and feature-rich ecosystem for solving differential equations in julia. J Open Res Softw 2017;5(1).
[26] Chen RTQ, Rubanova Y, Bettencourt J, Duvenaud D. Neural ordinary differential equations. 2018, arXiv:1806.07366.
[27] Rackauckas C, Ma Y, Martensen J, Warner C, Zubov K, Supekar R, et al. Universal differential equations for scientific machine learning. 2020, arXiv:2001.04385.
[28] Ji W, Deng S. Autonomous discovery of unknown reaction pathways from data by chemical reaction neural network. J Phys Chem A 2021;125(4):1082–92.
[29] Owoyele O, Pal P. ChemNODE: A neural ordinary differential equations framework for efficient chemical kinetic solvers. Energy AI 2021;100118.
[30] Raissi M, Perdikaris P, Karniadakis G. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys 2019;378:686–707.
[31] Ji W, Qiu W, Shi Z, Pan S, Deng S. Stiff-pinn: Physics-informed neural network for stiff chemical kinetics. J Phys Chem A 2021;125(36):8098–106.
[32] Bischof C, Carle A, Corliss G, Griewank A, Hovland P. ADIFOR–generating derivative codes from Fortran programs. Sci Program 1992;1(1):11–29.
[33] Bezanson J, Karpinski S, Shah VB, Edelman A. Julia: A fast dynamic language for technical computing. 2012, arXiv preprint arXiv:1209.5145.
[34] Niemeyer KE, Curtis NJ, Sung C-J. Pyjac: Analytical Jacobian generator for chemical kinetics. Comput Phys Comm 2017;215:188–203.
[35] Gururajan V, Egolfopoulos FN. Direct sensitivity analysis for ignition delay times. Combust Flame 2019;209:478–80.

[36] Lemke M, Cai L, Reiss J, Pitsch H, Sesterhenn J. Adjoint-based sensitivity analysis of quantities of interest of complex combustion models. Combust Theory Model 2018;23(1):180–96.
[37] Almohammadi S, Hantouche M, Le Maître OP, Knio OM. A tangent linear approximation of the ignition delay time. I: Sensitivity to rate parameters. Combust Flame 2021;230:111426.
[38] Lu T, Law CK. Toward accommodating realistic fuel chemistry in large-scale computations. Prog Energy Combust Sci 2009;35(2):192–215.
[39] Pepiot-Desjardins P, Pitsch H. An efficient error-propagation-based reduction method for large chemical kinetic mechanisms. Combust Flame 2008;154(1–2):67–81.
[40] Sun W, Chen Z, Gou X, Ju Y. A path flux analysis method for the reduction of detailed chemical kinetic mechanisms. Combust Flame 2010;157(7):1298–307.
[41] Smith GP, Golden DM, Frenklach M, Moriarty NW, Eiteneer B, Goldenberg M, et al. GRI-Mech 3.0. 1999, http://www.me.berkley.edu/gri_mech/.
[42] Nordin N. Numerical simulations of non-steady spray combustion using a detailed chemistry approach [Thesis for the Degree of Licentiate of Engineering], Goteborg, Sweden: Chalmers University of Technology; 1998.
[43] Richards K, Senecal P, Pomraning E. Converge 3.0; convergent science: Madison, WI, USA. 2021.
[44] Lu T, Law CK. On the applicability of directed relation graphs to the reduction of reaction mechanisms. Combust Flame 2006;146(3):472–83.
[45] Pepiot P, Pitsch H. Systematic reduction of large chemical mechanisms. In: 4th Joint meeting of the US sections of the combustion institute, vol. 2123. 2005.
[46] Kazakov A, Frenklach M. Reduced reaction sets based on GRI-Mech 1.2. Berkeley, CA: University of California at Berkeley; 1994, http://www.me.berkeley.edu/drm.
[47] Kingma DP, Ba J. Adam: A method for stochastic optimization. 2014, arXiv:1412.6980.
[48] Tao Y, Xu R, Wang K, Shao J, Johnson SE, Movaghar A, et al. A Physics-based approach to modeling real-fuel combustion chemistry–III. Reaction kinetic model of JP10. Combust Flame 2018;198:466–76.
[49] Ji W, Zanders J, Park J-W, Deng S. Data-driven approaches to learn HyChem models. In: Internal combustion engine division fall technical conference, vol. 85512. American Society of Mechanical Engineers; 2021, p. V001T06A011.
[50] Karniadakis GE, Kevrekidis IG, Lu L, Perdikaris P, Wang S, Yang L. Physics-informed machine learning. Nat Rev Phys 2021;3(6):422–40.
[51] Ge H, Xu K, Ghahramani Z. Turing: a language for flexible probabilistic inference. In: International conference on artificial intelligence and statistics. 2018, p. 1682–90, URL http://proceedings.mlr.press/v84/ge18b.html.
[52] Hoffman MD, Gelman A, et al. The No-U-Turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo. J Mach Learn Res 2014;15(1):1593–623.