Clustering-Enhanced Deep Learning Method for Computation of Full Detailed Thermochemical States via Solver-Based Adaptive Sampling

Xi Chen, Cédric Mehl, Thibault Faney, Florent Di Meglio

Clustering-enhanced deep learning method for computation of full detailed thermochemical states via solver-based adaptive sampling

Xi Chen,*,†,‡ Cédric Mehl,‡ Thibault Faney,‡ and Florent Di Meglio†

†*Centre Automatique et Systèmes, Mines Paris - PSL, 60 Bd Saint-Michel, 75272, Paris, France*

‡*IFPEN, 1 et 4 av. du Bois Preau, 92852, Rueil-Malmaison, France, Institut Carnot IFPEN Transports Energie*

E-mail: xi.chen1@minesparis.psl.eu

Abstract

Detailed chemistry computations are indispensable in numerous complex simulation tasks, which focus on accurately capturing the ignition process or predicting pollutant levels. Machine learning method is a modern data-driven approach for predicting full detailed thermochemical state-to-state behavior in reacting flow simulations. By combining unsupervised clustering algorithms to subdivide the composition space, the complexity of adaptive regression models for temporal dynamics can be significantly reduced. In this article, a more compact dataset is generated, which is essential for the clustering algorithm, by leveraging the adaptive CVODE solver time steps for data augmentation for stiff reactive states. A learning workflow that utilizes a deep residual network model (ResNet) in conjunction with an adaptive clustering algorithm is proposed. This approach aims to replace the stiff ODE direct integration solver traditionally used for computing thermochemical species' state-to-state temporal evolution

for detailed chemistry simulations. The learning models are adaptively trained using the K-Means clustering algorithm in the nonlinear transformation space for different subspaces of dynamic systems. Three test cases: H_2 (9 species), C_2H_4 (32 species), and CH_4 (53 species), are investigated, each exhibiting varying complexities. The study demonstrates that the iterative predictions of thermochemical states align well with the results obtained from direct numerical integration. Additionally, employing multiple adaptive regression models in subdomains yields superior performance compared to a single regression model prediction case.

1 Introduction

With the growth of computing power, simulating reacting flows in space and time at various scales has become feasible. However, typical chemical processes entail a multitude of species and a vast number of chemical reactions, coupled with diffusive and convective transport phenomena. As a result, computationally demanding tasks, such as simulating industrial processes like combustion in fuel engines or glass furnaces, and simulation in complex chemical reactors, present significant challenges. Modeling chemical processes involves non-linear stiff ordinary differential equations (ODEs) and requires appropriate solvers like CVODE.[1] Integrating the chemical ODEs often becomes the bottleneck in reactive flow simulations.

State-of-the-art methods for chemistry accelerations traditionally rely on chemistry reductions and chemistry tabulation methods. In the case of chemistry reductions, only the major chemical species are considered, while unimportant species and reactions are excluded from the complex mechanisms.Various approaches, such as the Quasi-Steady State Approximation (QSSA)[2] or Directed Relation Graph (DRG),[3] have been proposed to derive reduced mechanisms. These reduced mechanisms are then integrated using direct integration solvers, utilizing available computational resources. However, reduced mechanisms containing a large number of species are still necessary for studying pollutants like nitrogen compounds. Alternatively, tabulation methods, such as Look-Up Tables (LUT)[4] and In-Situ Adaptive Tab-

ulation (ISAT),[5] can be employed to pre-compute chemical solutions based on canonical problems using detailed chemical mechanisms. The pre-computed terms are stored in tables with a reduced set of variables and utilized later for chemistry integrations. Nevertheless, the curse of dimensionality limits their application to simple cases, as the storage memory exponentially increases with the number of chemical species.

In addition to traditional chemistry tabulation, machine learning algorithms using artificial neural networks offer an alternative method for accelerating detailed chemistry integration. Machine learning has recently gained significant interest in the applied science and engineering fields.[6] It has been widely applied to computer vision (CV), natural language processing (NLP) for several years, and more recently to various scientific computation problems, such as for reactive flow simulations.[7,8] In theory, deep learning models can serve as surrogate models capable of fitting any non-linear model[9].[10] Previous studies have demonstrated that neural networks can replace traditional integration methods for chemical accelerations.[11,12] One notable advantage of utilizing deep neural networks (DNNs) over chemical look-up tables is their low memory requirement.[13] The number of parameters needed for a DNN is much smaller compared to the number of multi-dimensional points that would have to be stored in a table for a complex chemical reaction. Complex neural network models with moderate memory storage capacity can handle high-dimensional inputs, making them suitable for large chemical mechanisms, whereas Look-Up Tables and ISAT store only a limited number of input variables. Further studies have extended the application of neural networks to even more complex cases. Sen and Menon[14] employed neural networks for chemistry integration in turbulent premixed and non-premixed abstract problems, including syngas combustion with a 14-species mechanism and methane combustion with a 16-species mechanism. Wan et al.[15] utilized ANN chemistry integration for the direct numerical simulation of a syngas turbulent oxy-flame with side-wall effects, incorporating an 11-species reduced mechanism.

Most of these research studies demonstrate the efficacy of deep learning models in accel-

erating the computation of source terms compared to direct numerical simulations. Recent investigations have further expanded the scope by focusing on more complex fuels and problems. For instance, Ranade et al.[16] employed DNN models to reconstruct the temporal pyrolysis and oxidation processes of complex fuels based on experimental data. Similarly, Ding et al.[17] applied multiple DNN models to predict the time evolution of each of the 32 states involved in CH_4 combustion, with individual neural networks dedicated to each chemical species. Furthermore, An et al.[18] employed neural networks which replace the direct integration of the reduced skeletal mechanism of kerosene with $C_{10}H_{22}$ in the context of engine system simulations, where the total dimension number is strongly reduced to 41 with 132 elemental reactions neglecting many minor species. These studies reveal the potential of employing neural networks to accelerate chemical integration for complex fuels while maintaining good prediction accuracy. In addition, embedding augmented physical information into the loss function of neural networks represents a more recent approach to ensure that the prediction results align with the numerical solution of ODEs.[19] However, research on this approach has so far been limited to simple cases, necessitating further extensions and investigations. Moreover, several studies also combine machine learning methods and traditional approach to tackle the complex fuel problems, including using data-driven method to reduce the full chemistry to subspace manifold by linear and nonlinear models,[20–23] and using neural networks to predict the flamelet generated manifolds.[24–27]

Splitting the chemical manifold into several subdomains and training multiple ANNs can improve learning efficiencies by facilitating the training process and reducing local model complexity. Various methods have been employed to efficiently separate subdomains and enhance prediction accuracy. Physical descriptions of the combustion process, such as progress variables[28] or the rate of temperature increase,[29] can effectively partition the subdomains and improve prediction accuracy. Another approach is to utilize data-driven classification algorithms. One such algorithm is the self-organizing map (SOM) with a neural network structure, which preserves the topological structure of the data and produces a low-dimensional

representation to partition input data points.[18,30,31] Another option is the K-Means partitioning algorithm. This algorithm partitions the data by calculating the distance between each data point and a fixed number of centroids, which represent different clusters. In earlier research, Perini et al.[32] applied an optimal K-Means algorithm to partition the full chemistry states into subdomains with similar reactive conditions, successfully accelerating the simulation time of combustion with high-dimensional full chemistry in engine environments. Similarly, Barwey et al.[33] used a predefined number of clusters to partition the data and demonstrated its ability to separate stiff reactive regions from non-stiff regions in combustion processes. Additionally, Nguyen et al.[34] employed a hierarchical K-Means algorithm with predefined cluster numbers. These advances underscore the effectiveness of employing non-supervised partitioning algorithms in complex combustion simulations, enabling improved prediction accuracy and efficient handling of different combustion regimes.

The performance of ANN training results strongly relies on the quality of the dataset used. Generating an appropriate dataset that includes information from different scales of chemical species is a challenging task. In early studies of simple chemical mechanisms, researchers employed random sampling of chemical species within predefined operating ranges[11,12]. Han et al. applied a fixed small-scale sampling time step to ensure the capture of fast ignition and heat release processes. Additionally, they introduced input simulation noise during each resolution time step to generate a more robust dataset. In a study by Zhang et al.,[35] different strategies for data-driven generative sampling methods were compared. The authors proposed a new multi-scale sampling method for different scales of chemical species. In this approach, the species were classified into major and minor groups, and random sampling was performed at fixed intervals using either normal or logarithmic distributions. Each of these methods aims to ensure the dataset captures the essential features of the chemical processes at different scales.

The review of these studies highlights the potential of using ANN-based chemistry integration for more complex fuels. However, combining non-supervised K-Means clustering

algorithm with local adaptive temporal full detailed chemistry computation for high dimension complex fuels has not been evaluated. Additionally, multi-scale data sampling becomes more challenging for high-dimensional complex fuels with numerous minor species. In this paper, we propose a workflow for learning 0D chemical kinetics using unsupervised clustering combined with deep residual networks. The performance of models by varying the number of clusters and network hyperparameters is systematically analysed. To generate a compact dataset, the adaptive time marching steps of the CVODE solver is applied to generate the sampling sequence, which allows to obtain more data points in the fast ignition and reaction regions. Multiple models are trained individually for different composition spaces, with subdomains of the dynamical system identified using the K-Means clustering algorithm in logarithmic space. The results, along with the analysis of model training and iterative inference statistics, demonstrate that more accurate predictions can be achieved by empirically selecting optimal cluster numbers as hyperparameters for different combustion cases. Furthermore, the deep residual network outperforms traditional multi-layer perceptrons (MLPs) in terms of training performance.

The remaining sections of the paper are organized as follows. Section 2 provides a general description of the chemistry evolution equations. Section 3 presents a novel strategy for generating the training datasets based on a canonical multiple initial conditions numerical experiment. Section 4 introduces the general workflow, including data pre-processing, clustering algorithm, and the proposed regression model. Section 5 focuses on the evaluation of data clustering, model training, and statistical inference based on actual 0D ignition simulations.

2 Physical problem formulation

This section details the chemical kinetics problem. The ODEs describing the problem are detailed in Sec. 2.1 and the numerical solving in Sec. 2.2.

2.1 Combustion equations

Reactive flows involve the coupling between convection, diffusion and chemical reactions. A wide range of physical scales are present in the flow and the resolution of the fully coupled system is often out of reach. To address this challenge, a common approach is to employ operator splitting methods[36],[37] which solves the chemistry and transport phenomena separately. In a multi-dimensional simulation, the chemical system in each computational cell is described by a set of ODEs in time:

$$\begin{aligned} \frac{dY_s}{dt} &= \frac{M_s}{\rho}\dot{\omega}_s \qquad \forall s \in \{1, ..., N_s\} \\ \frac{dT}{dt} &= -\frac{1}{\rho C_p}\sum_{s=1}^{N_s} h_s \dot{\omega}_s \end{aligned} \tag{1}$$

where N_s represents the total number of chemical species. The variables Y_s, h_s, M_s, and $\dot{\omega}_s$ denote the mass fraction, molar enthalpy, molar weight, and chemical reaction rate for species s, respectively. The variables T, C_p, and ρ represent the temperature, specific heat capacity at constant pressure, and density of the gas mixture, respectively. The reactors are assumed to be adiabatic and homogeneous. The system is expressed as follows in a generic form:

$$\begin{aligned} \dot{\mathbf{S}}(t) &= \mathbf{f}(\mathbf{S}(t), q) \qquad t \in [0, t_{n+1} - t_n] \\ \mathbf{S}(t=0) &= [T(t_n), Y_1(t_n), ..., Y_{N_s}(t_n)]^T \end{aligned} \tag{2}$$

where q is a variable regrouping all thermodynamic constants.

2.2 Chemical kinetics solver

A common solver for solving the system described in 2 is CVODE.[1] This solver is particularly employed in CANTERA,[38] which is a chemical computation software used for calculating thermodynamic and chemical species terms in reactive flow systems, including the present work. CVODE is a multi-step solver with variable order and step sizes, and it utilizes

dynamic adaptive time stepping. To approximate the solution $\mathbf{S}(t_n) = \mathbf{S}^{(n)}$ at time t_n, CVODE solves the following algebraic equation:

$$\sum_{i=0}^{K_1} \alpha_{n,i} \mathbf{S}^{(n-1)} + h_n \sum_{i=0}^{K_2} \beta_{n,i} \dot{\mathbf{S}}^{(n-i)} = 0 \tag{3}$$

where $h_n = t_{n+1} - t_n$ is the time step. To tackle stiff problems, Backward Differentiation Formulas (BDF) is specifically considered in fixed-leading coefficient (FLC) form,[1] defined by $K_1 = q$ and $K_2 = 0$. The order q ranges from 1 and 5. The coefficients are fixed by the method type, its order, the recent history of the step sizes and the normalization $\alpha_{n,0} = -1$. The standard CVODE chemical solver in CANTERA effectively handles thermochemical states using implicit and combined stiff/non-stiff solvers.[38] By adaptively resolving the stiff and non-stiff regions with different time steps, the CVODE solver refines the step sizes in areas with rapid reaction rates. Figure 1 illustrates the simulations of temperature with dynamically adjusted time steps during the temporal evolution of H_2, C_2H_4, and CH_4 cases, where the x axis for time evolution is plotted under the logarithmic scale. The initial temperature and equivalence ratio are set to $(T_0, \phi) = (1700K, 1.0)$. It can be observed that the CVODE solver refines the time step dt_{cvode} in fast-reacting regions while using larger time steps in the starting and equilibrium regions.

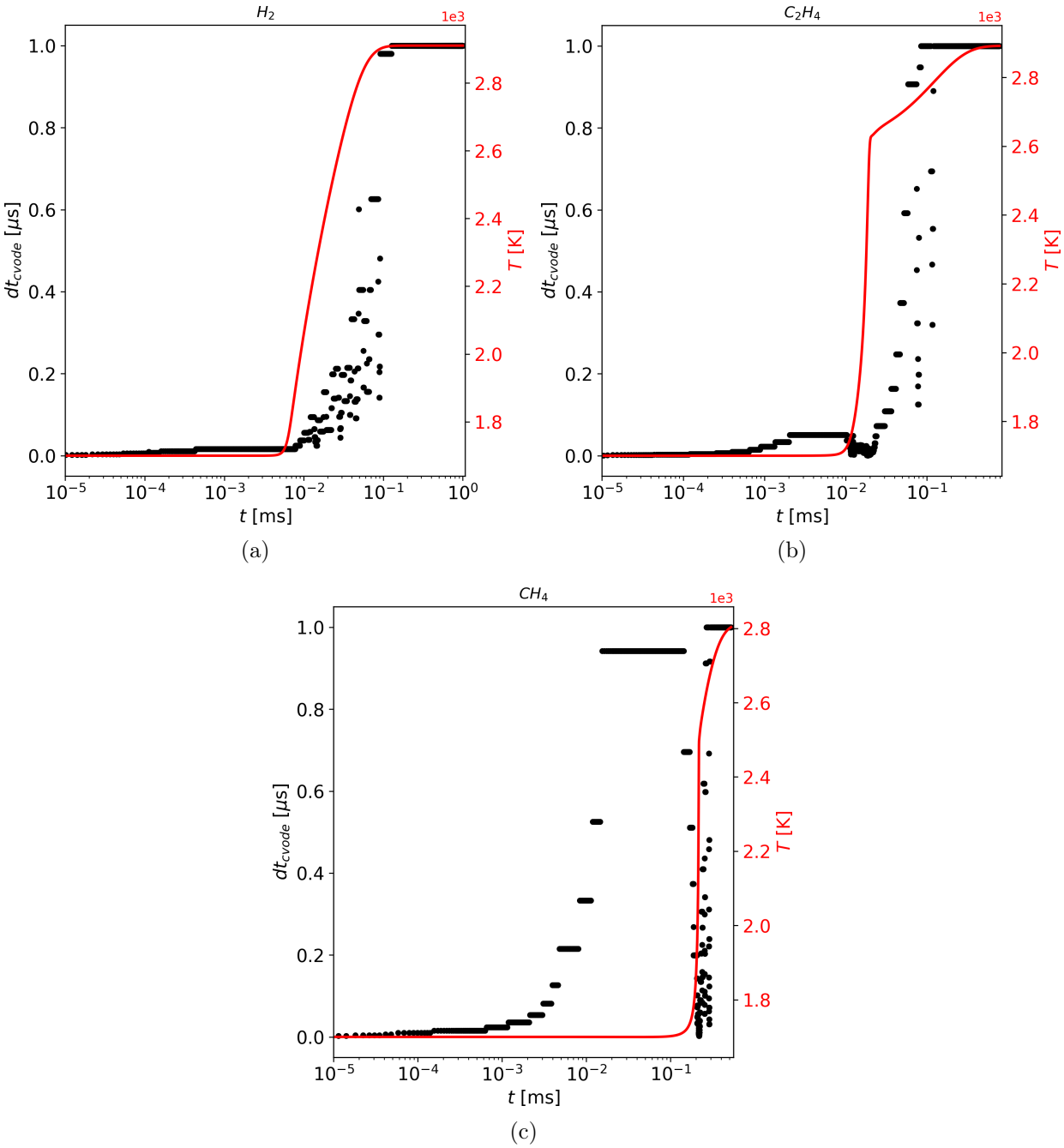


Figure 1: Dynamic adaptive time steps used by CVODE solver with (a)H_2/air, (b)C_2H_4/air, and (c)CH_4/air cases, where the simulations are set with $T_0 = 1700.0K$ and $\phi = 1.0$ The red lines represent the numerical solution of temperature, and the black dot points represent the local adaptive time steps given by CVODE solver

During a 3D simulation using operator splitting, the ODE solver typically performs numerous adaptive time steps within a single computational fluid dynamics (CFD) time step

$dt_{cfd} = t_{n+1} - t_n$. The goal is to replace these individual time steps with a single function call that approximates $S(t + dt_{cfd})$ based on $S(t)$. In this paper, dt_{cfd} is set to a fixed value of $dt_{cfd} = 10^{-6}$ seconds. In simulations using operator splitting approaches, the time step is usually limited by stability limits on convection and diffusion (CFL and Fourier numbers are typically defined) and thus dictated by the CFD solver. The extension of the current strategy to variable dt_{cfd} is out-of-the-scope of the present paper and will be tackled in future research. The constant dt_{cfd} value selected here is a typical value found in large eddy simulations of systems such as gas turbines or engines. The following section outlines how CVODE is utilized to generate the dataset that will be used for training a model capable of achieving this objective.

The following section outlines how CVODE is utilized to generate the dataset that will be used for training a model capable of achieving this objective.

3 Dataset generation

In this paper, the study of three different fuel combustion scenarios involving the fuels H_2, C_2H_4 and CH_4, as outlined in table 1.The selection of these cases progressively increases the complexity of the thermochemical reaction system.

Table 1: Summary of three different studied fuel cases

Mechanism	number of species	number of reactions
H_2 [39]	9	19
C_2H_4 [40]	32	206
CH_4 [41]	53	325

3.1 Generation of simulation trajectories

The design of experiments to generate the training dataset consists in specifying multiple initial conditions for the temperature and species mass fractions for the set of ODE 2.

However, not all initial species' mass fractions are physically relevant: in this work, we assume that the initial composition is a pure mixture of fuel and air, where the mass fractions for all other species are set to zero. A common practice in combustion is to specify the initial mass fractions values with a more significant variable called the equivalence ratio ϕ, which is a measure of the excess of fuel in the mixture with respect to stoichiometry, and expressed as:

$$\phi = \frac{m_{fuel}/m_{ox}}{(m_{fuel}/m_{ox})_{st}} \tag{4}$$

where m are masses and n are numbers of moles, and the suffix st refers to stoichiometric conditions. The equivalence ratio provides a measure of the excess or deficiency of fuel in the mixture. By specifying the equivalence ratio, we can effectively control the initial mass fractions of the species in the combustion simulation.The range of initial conditions (IC) for the simulations is limited. The pressure is kept constant at the standard atmospheric pressure of 1 atm. The temperature ranges from $1600K$ to $1800K$, while the equivalence ratio varies from 0.7 to 1.5. To generate the training dataset, a total of 1000 initial conditions are randomly selected within these intervals using Latin Hypercube Sampling (LHS), as illustrated in Figure 2. Each initial condition is then used to simulate a trajectory that represents the evolution of species mass fractions and temperature over time. The resulting database consists of pairs $(S(t), S(t + dt_{cfd}))$ sampled along these trajectories. Since our objective is to simulate over multiple time steps starting from a new given initial condition $(T_0, \phi)_{\text{new}}$, it is important to split the database based on trajectories rather than individual pairs across all trajectories. all pairs of data $(S(t), S(t + dt_{cfd}))$ from a specific training trajectory will be assigned to the training database. The total dataset is splitted with a ratio of 75%/15%/10% for train, validation, and test datasets, respectively, based on trajectories. A criterion is determined to end a given simulation by defining the following variable:

$$\tau = \left| \frac{T_{eq} - T(t)}{T_{eq}} \right| \tag{5}$$

Where $T(t)$ denotes the temperature of the local time step and T_{eq} denotes the temperature at the equilibrium state that is determined for given initial conditions (ϕ, T_0) by a simple thermodynamic equilibrium computation. The simulations are terminated when the total simulation time reaches $10^{-3}s$. At this point, it is assumed that all variables have reached their equilibrium states. Alternatively, if the simulation reaches convergence with a given tolerance τ, the simulation is also considered complete. In this work, the tolerance τ is set to 10^{-4} to ensure that all simulations from different initial conditions reach their final equilibrium states. In the next section, a novel strategy for generating the dataset is presented, which takes advantage of the time step adaptation feature of the CVODE solver.

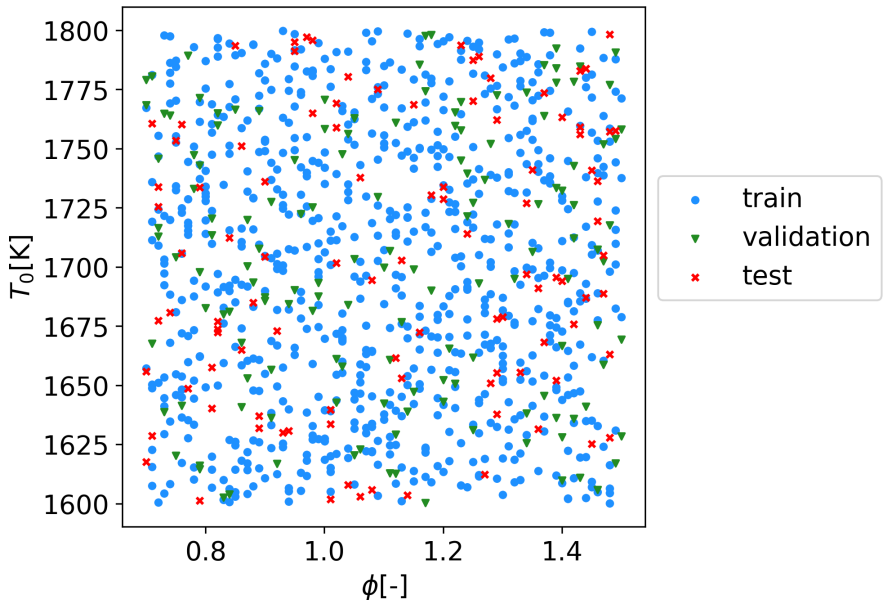


Figure 2: Initial conditions distribution

3.2 Data acquisition from simulation trajectories

The data pairs $(\mathbf{S}(t), \mathbf{S}(t + dt_{cfd}))$ are acquired within the generated trajectories in the chemical manifold. However, the states evolution rates vary with time. Roughly speaking, reactions are "slow" close to the equilibrium and "fast" after ignition. A naive strategy

would consist in selecting sampling points from the target trajectory during the simulation with a regular time step, noted as dt_s. In other words, the dataset would contain pairs of the form $(\mathbf{S}(kdt_s), \mathbf{S}(kdt_s + dt_{cfd}))$, where dt_s denotes the uniform sampling time step.The straightforward approach of selecting data points from the target trajectory with a regular time step (dt_s) may result in an imbalanced dataset. This method would result in fewer data points in the ignition region, where chemical reactions occur rapidly. As a consequence, the dataset would be imbalanced with a skewed distribution of data points. This imbalance can be observed in the temperature and CO_2 mass fraction distributions, as shown in Figure 3 for the case of C_2H_4.

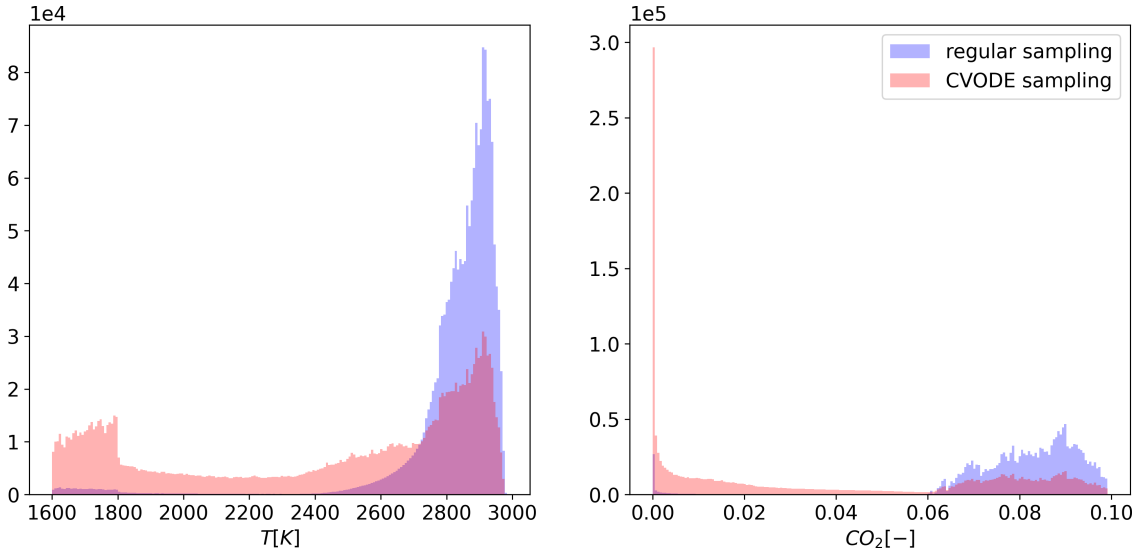


Figure 3: The sampling points distribution of (a) temperature and (b) CO_2 values generated by regular sampling method (blue) and CVODE sampling method (red) for the C_2H_4/air case. The total size of data points generated by two strategies is around 1.5×10^6.

In this research, the dynamic time step adaptation feature of the CVODE solver is utilized to generate the sampling sequence. This feature allows us to generate a more balanced dataset by leveraging the time step adaptation performed by CVODE during the simulation, as presented in section 2.2. To generate the dataset, a format of $(\mathbf{S}(t_{cvode}), \mathbf{S}(t_{cvode} + dt_{cfd}))$ is applied, where t_{cvode} represents the time sequence obtained from the CVODE solver's time step adaptation. To achieve this, the simulations by CVODE run twice at each time step:

once to obtain the points $\mathbf{S}(t_{cvode})$, and a second time to obtain the sequence $\mathbf{S}(t_{cvode}+dt_{cfd})$. By taking adaptive time steps, this approach ensures that there are more sampled points in fast reaction regions, which is beneficial for training the model. Furthermore, the maximum CVODE evolution time step is limited to dt_{cfd} to ensure a sufficient resolution in the near-equilibrium region. This restriction helps maintain accuracy in those regions. The resulting dataset is better balanced compared to using a constant time step, as demonstrated in Figure 3. This balanced dataset is crucial for training neural networks effectively. In the next section, a detailed description of the training pipeline is provided.

4 Learning methodology

The objective in this study is to replace the direct integration by deep ANN models (DNN) for each CFD time step resolution, as illustrated in 4. The objective is to have a model that can predict the thermochemical states at time $t_0+k\Delta t$ as $\mathbf{S}(t_0+k\Delta t) = \mathbf{f}_{DL}^{(k)}(\mathbf{S}(t_0);\theta)$, where θ represents the parameters of the learning model. In this workflow $\mathbf{S}$ represents the states of temperature and each chemical species. Indeed, for such systems the temperature might be directly estimated from species mass fractions and enthalpy. In an attempt to make the model more general and not solely tailored for constant pressure low Mach problems, the temperature is included as an input of the DNN in the context of this research. In this section, a detailed explanation of how the dataset is processed to train the neural networks for predicting the thermochemical states from their values at time t is introduced.

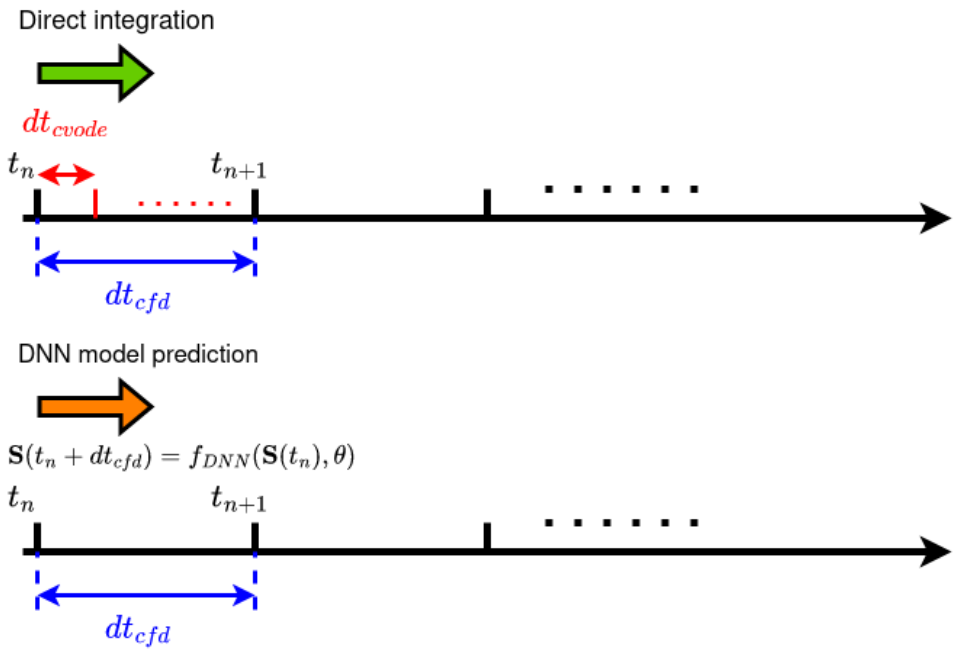


Figure 4: The workflow for the states prediction with time evolution

4.1 Data Pre-processing

To address the issue of chemical species values being in different scales and having extremely small values skewed towards zero, a pre-treatment of the chemical species values is performed using a logarithmic transformation. The transformed state vector is denoted as $\mathbf{S}_l = [T, \ln(Y_j)]$, where $j = 1, ..., N_s$. This transformation ensures that the variables are scaled to similar ranges, which is particularly beneficial for improving the prediction accuracy of minor chemical species. However, before applying the logarithmic transformation, it is necessary to clip the zero values by a threshold. This is because the logarithmic transformation is only applicable to positive values. Empirical thresholds are determined for each dataset, where the thresholds used for the H_2, C_2H_4, and CH_4 cases are 10^{-10}, 10^{-12}, and 10^{-28}, respectively. These thresholds have been chosen to preserve the accuracy of reactor computations. Two aspects seem to have an important impact on the threshold choice:

- The smallest species mass fraction magnitude differ from one chemical mechanism to another. For instance here, the maximum magnitude is around 10^{-8} and the maximum

magnitude is around 10^{-40} for C_2H_4 and CH_4. The integration needs to take these species into account to recover the proper chemical evolution and this sets a higher bound for the threshold.

- The thresholds need to be high enough to suppress numerical artifacts which arise from the CVODE numerical integration for some chemical species with small mass fractions.

A quantitative comparison of training results with different thresholds for C_2H_4 case are provided in to supporting information as an example.

In addition to the logarithmic transformation, a scaling of the data is applied to facilitate the training of the DNN models. A standard normalization technique is employed, which sets the mean of each feature to zero and the variance to one. This scaling helps in ensuring that all features contribute effectively to the learning process.

4.2 Data clustering

A strategy to simplify the learning process is to separate the composition space into several sub-domains. This division is achieved using the K-Means method, which is a non-supervised clustering algorithm. The K-Means algorithm partitions the sampling points into sub-domains by minimizing the average squared distance between K centroids and the sampling points. The K-Means algorithm considers the mean distance in the logarithmic scale to maintain consistency with the data pre-processing used for DNN regression. Each sub-domain corresponds to a separate region of the composition space. In this approach, a distinct DNN model is trained for each sub-domain. This allows for more specialized and focused modeling within each region of the composition space. In this work, the K-Means algorithm is implemented based on K-Means++, which is an improved version of the standard K-Means algorithm. K-Means++ enhances the clustering quality compared to the standard algorithm. More information about this algorithm can be found in appendix A.

4.3 Neural network model

Neural networks, particularly multiple-layer perceptrons (MLP), have been extensively used as surrogates for combustion kinetics solvers. MLPs are popular due to their simplicity and computational efficiency. Numerous studies have demonstrated the accuracy of MLPs in combustion problems[7,8] However, when dealing with large chemical mechanisms, accurate regression using MLPs may require a significant number of parameters. This can lead to optimization challenges, such as the vanishing gradient problem commonly encountered in machine learning applications.[6] The vanishing gradient problem refers to the issue where the gradients during backpropagation diminish as they propagate through many layers, making it difficult for the network to learn effectively. To address this challenge, an alternative approach used in this work is the application of residual layer structures with shortcut connections, known as ResNet.[42] Similar topology of ResNet design has been utilized in various physico-chemistry computations, such as fluid flash computations[43] and flamelet progress variables tabulation.[26] The ResNet architecture has been shown to mitigate the vanishing gradient problem and enable efficient learning for complex problems. Figure 5 provides an illustration of the ResNet architecture. The ResNet architecture in our research consists of basic units called resblocks, which include two standard hidden layers. The key principle of ResNet is the inclusion of shortcut connections that bypass one or more layers, allowing each layer to predict an increment rather than a direct value. This enables the network to learn residual information, which facilitates the training process. If x is the input of a resblock, the output y is given by:

$$\mathbf{y} = \sigma(\mathcal{F}(\mathbf{x}) + \mathbf{x}) \tag{6}$$

where $\mathcal{F}(\mathbf{x})$ is the feed-forward neural network composed of the two hidden layers and σ represents the non-linear activation function. The number of residual blocks is denoted as n_r, and the neuron number in each hidden layer is represented as n_e. The swish activation

function[44] will be used throughout this work:

$$\text{swish}(x) = \frac{x}{1 + e^{-\beta x}} \tag{7}$$

In this equation, β is set by 1 by default as indicated in[44]. In practical applications, particularly for regression problems, the use of the *swish* activation function has been found to be more efficient for the optimization process compared to commonly used activation functions such as *ReLU*, *sigmoid*, or *tanh*.[29,43] This observation has been verified through various experiments conducted in this study. A summary of the DNN-based model architecture is provided in Fig. 6.

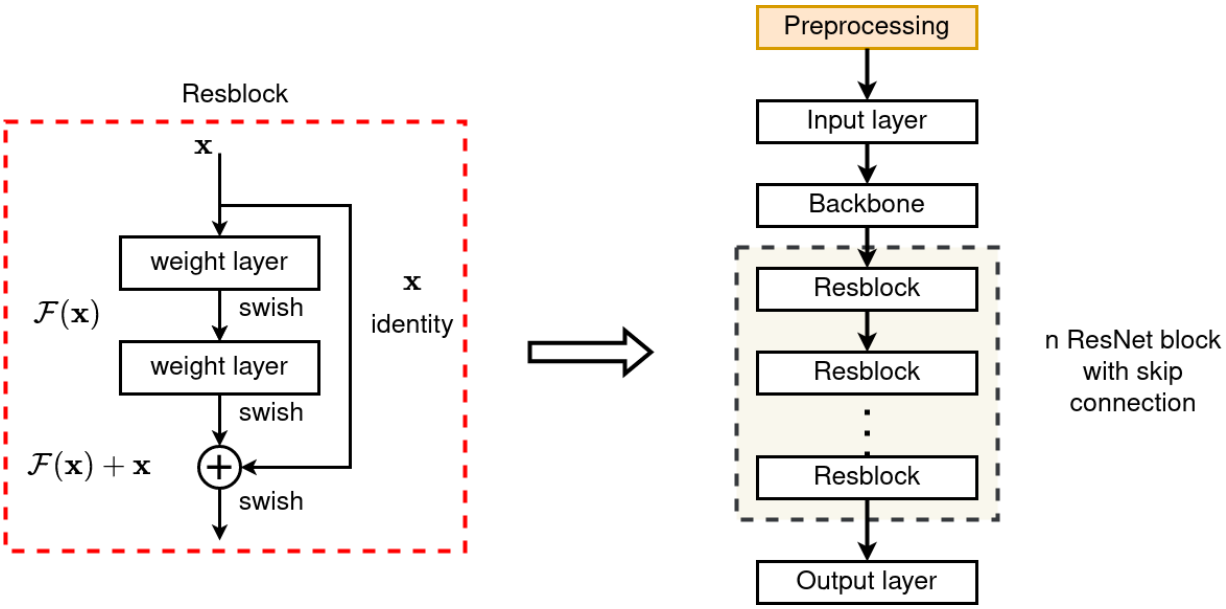


Figure 5: Neural network learning model structure design

4.4 Model training

In the present work, the Mean Squared Error (MSE) is used as the loss function for model training. The DNN models are trained using the Tensorflow 2.10.1 framework.[45] For optimization, the Adam algorithm is chosen.[46] The model parameters are initialized using the Glorot Uniform method, and an exponential decay strategy is applied to the initial learning

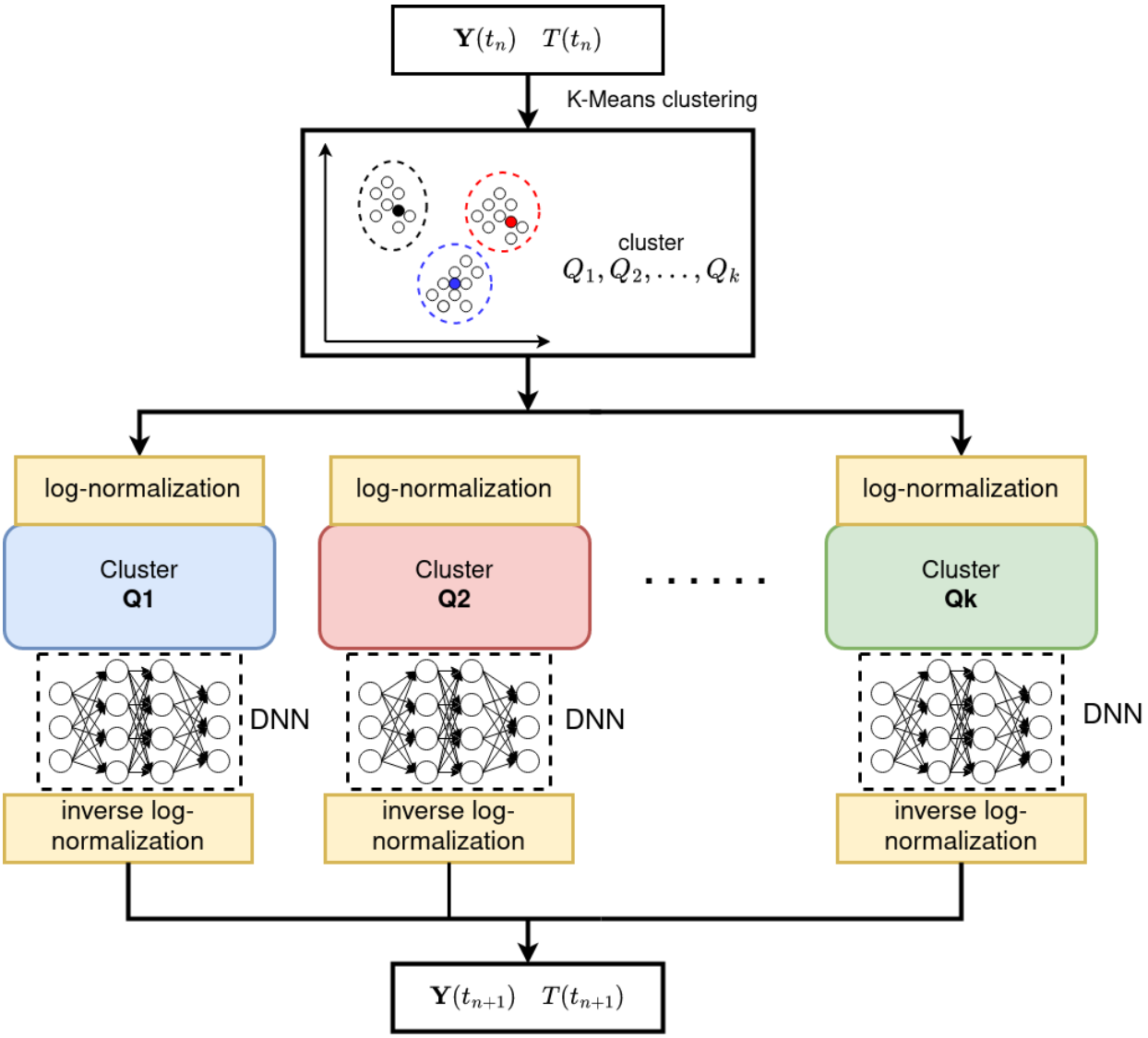


Figure 6: The global learning algorithm for the states prediction with the separation of composition space

rate during the training epochs. In this research, the initial learning rate for all three cases is set to 0.005. A learning decay rate is employed to aid the optimization process,[47] and it is defined as $r(k) = r_0 \times \eta^{\frac{k}{N}}$, where r_0, η, k, and N represent the initial learning rate, the decay rate, the k^{th} step, and the total number of decay steps, respectively. In this study, the decay rate is empirically set to 0.92 and the number of decay steps to 650.

4.5 Model performance evaluation

The DNN model serves as an operator for numerical time-stepping, denoted as $\mathbf{f}_{DL}$. By utilizing this learned operator, we can simulate the trajectory of a chemical reactor with a fixed initial condition by iteratively computing $\mathbf{S}(t_0 + k\Delta t) = \mathbf{f}_{DL}^{(k)}(\mathbf{S}(t_0 + (k-1)\Delta t); \theta)$. During the time evolution, the thermochemical states are computed iteratively in each time step resolution until the end of the simulation. Therefore, it is insufficient to evaluate the prediction performance for a single input-output pair of the models. The error may accumulate and lead to divergence after multiple iterations. Hence, it is crucial to assess the overall inference performance over multiple iterations for the entire simulation.

To compare the results with reference numerical simulations, a global accumulative logarithmic mean average percentage error $\mathcal{M}_i$ is used to evaluate the performance of chemical species trajectories under different initial conditions. Additionally, the normal mean average percentage error $\mathcal{M}_0$ is employed to evaluate the temperature prediction. It is important to note that since the models are trained using data in logarithmic space for chemical species, the logarithmic error metric is used for consistency. These introduced errors are denoted mathematically by:

$$\begin{aligned}
\mathcal{M}_0(T_0, p_0, \phi) &= \frac{1}{N_{iter}} \sum_{k=1}^{N_{iter}} \left| \frac{T^{pred}(kdt) - T(kdt)}{T(kdt)} \right| \\
\mathcal{M}_i(T_0, p_0, \phi) &= \frac{1}{N_{iter}} \sum_{k=1}^{N_{iter}} \left| \frac{\ln(Y_i^{pred}(kdt)) - \ln(Y_i(kdt))}{\ln(Y_i(kdt))} \right| \quad i = 1, ..., N_s
\end{aligned} \tag{8}$$

where i represents each state component (temperature and chemical species), and N_{iter} is the number of total iterations until the final time step prediction which is specific for each simulation with different initial conditions. Each $\mathcal{M}_i$ is computed for one trajectory with one initial condition, for one state component. The overall average errors $\mathcal{M}$ are computed by averaging values of all dimensions, which is written as:

$$\mathcal{M}(T_0, p_0, \phi) = \frac{\mathcal{M}_0(T_0, p_0, \phi) + \sum_{i=1}^{N_s} \mathcal{M}_i(T_0, p_0, \phi)}{N_s + 1} \quad i = 1, ..., N_s \tag{9}$$

The global errors $\mathcal{M}$ and errors for each state component $\mathcal{M}_i$ are statistically evaluated for all 100 initial conditions in the test set. This metric accounts for the potential error accumulation that may arise when repeatedly using a neural network to predict the evolution of chemical states. By taking into account the complete range of initial conditions, the overall performance and reliability of the neural network model can be evaluated in accurately capturing the dynamics of the chemical reactions.

5 Results

This section presents the results of the DNN model training performance and evaluates the simulation results achieved using the trained models. Primarily, a parametric assessment of the K-means clustering is conducted in Section 5.1. Then, the investigation of the performance of the DNN surrogate model on 0-D reactor simulations is carried out in Section 5.2. Finally, a particular study about the influence of the database generation method, comparing regular and CVODE-based sampling is introduced in Section 5.3.

5.1 Analysis of data clustering

The number of K-means clusters needs to be selected beforehand as there is no method for systematic selection. A practical approach to evaluate the clustering efficiency is to compute the distortion $\mathcal{D}$, which corresponds to the squared Euclidean distance between the data points and the centroids:

$$\mathcal{D} = \sum_{i=1}^{K} \sum_{\mathbf{S} \in \Omega_i} \left\| \hat{\mathbf{S}}_l - \mathbf{d}_i \right\|^2 \tag{10}$$

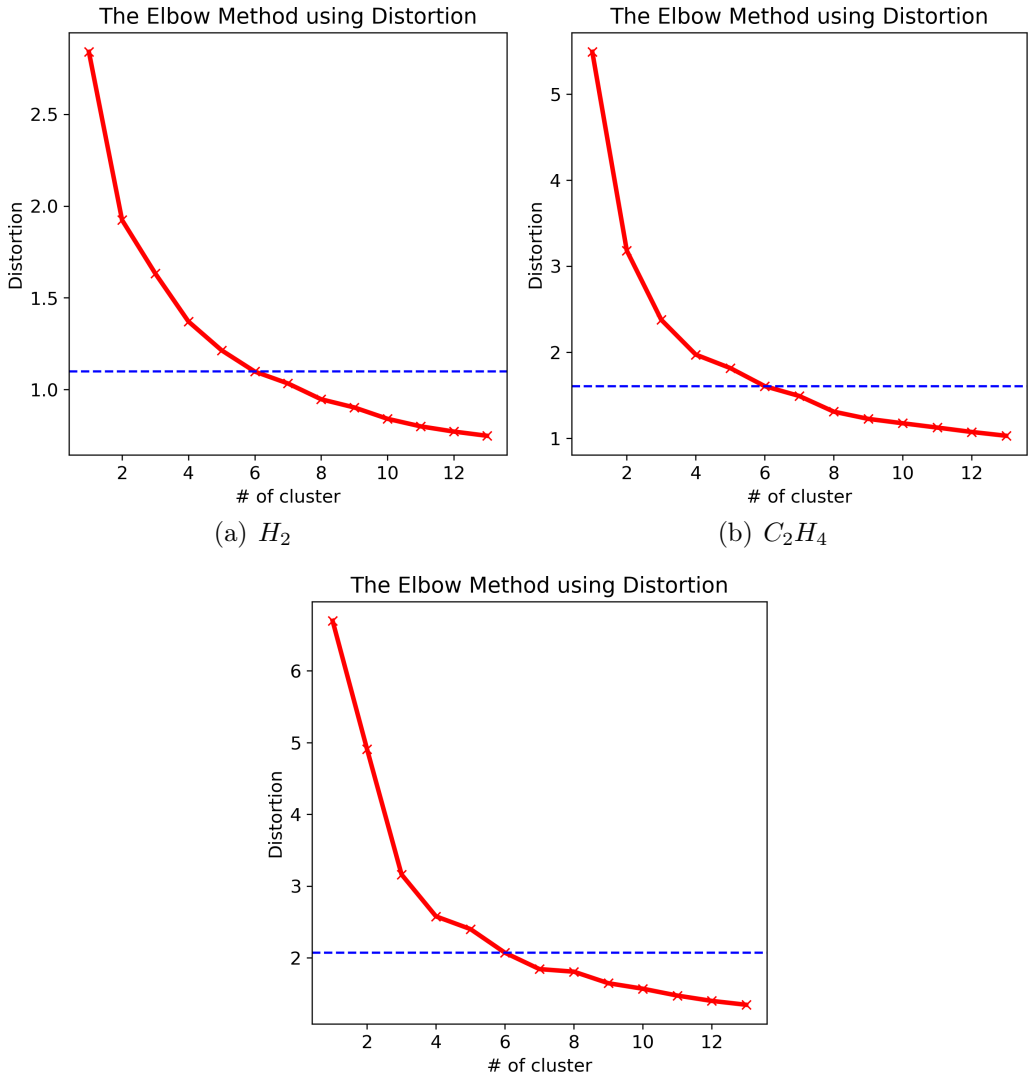
where $\mathbf{S}_l$ is the logarithm of the state vector and $\hat{\mathbf{S}}_l$ its normalized counter-part. $\mathbf{d}_i$ represent the coordinates of the K centroids. The evolution of the distortion with the number of clusters is shown in Figure 7 for H_2, C_2H_4 and CH_4. It can be observed that as the number of clusters increases, the distortion decreases monotonically, with a steeper slope initially and

a smoother evolution thereafter. In this study, the optimal number of clusters for each fuel is determined empirically within a predefined range by evaluating inference simulations. It should be noted that while the data is separated into subdomains with a converged distortion value, the local data distribution may not be optimal for model training, which can lead to inference instabilities during iterative predictions. Models are evaluated with up to six clusters, as the distortion value does not significantly decrease for larger numbers of clusters, indicating that all sampling points have already been optimally partitioned around local centroids. Moreover, with an increasing number of clusters after six, some subdomains may have insufficient data for effective local model training. After partitioning the dataset, individual learning models are trained and evaluated for each subdomain.

5.2 Model training evaluations

Several model training strategies are crucial for achieving optimal performance in this research. Firstly, trajectory learning under the logarithmic scale is necessary to address the prediction of extremely small values. After the data preprocessing, the rescaled data for each subdomain model in different cases are shown in Figure 8. The logarithmic transformation helps to transform the original data distribution into a more homogeneous distribution and removes extreme values skewed towards zero.[48] Data standardization, on the other hand, rescales the original data to have a mean of zero and a unit variance.

Training with a deeper neural network structure, including residual shortcuts, is more efficient compared to traditional multiple-layer perceptrons (MLP). This is demonstrated by comparing a standard MLP with the same number of hidden layers and neurons in each layer to our optimal trained ResNet models. Additionally, the activation function *swish* is expected to improve the optimization process compared to the commonly used *ReLU* activation function in previous research papers[15,34]. The evolution of the loss function over training epochs for models in a cluster containing strongly reactive states is depicted in Figure 9. It can be observed that the *swish* activation function significantly promotes

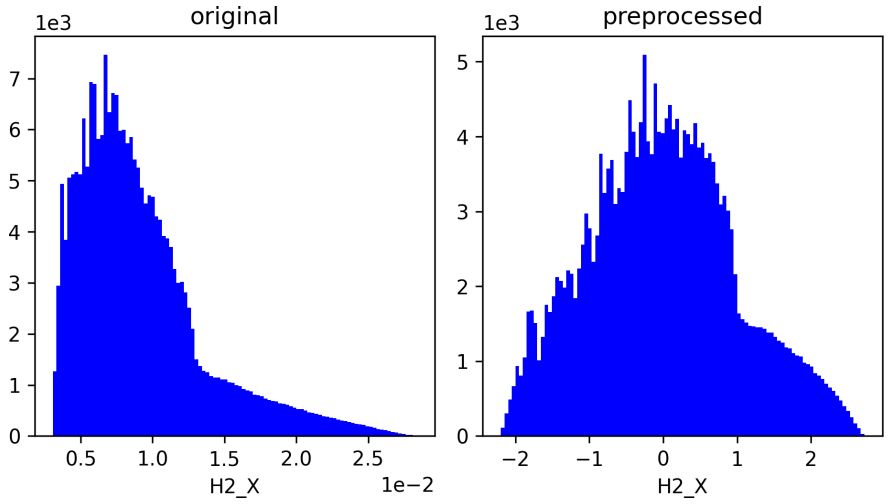


(a) H_2 (b) C_2H_4

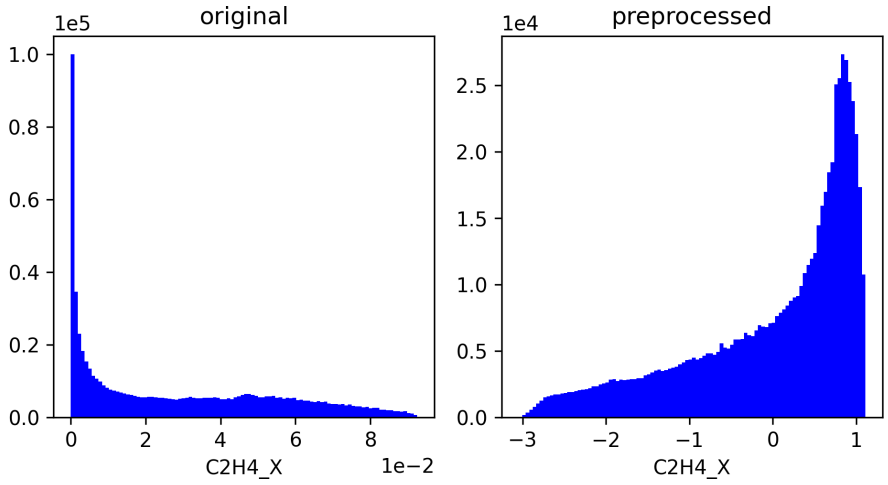
(c) CH_4

Figure 7: distortion values over cluster numbers for (a) H_2/air, (b) C_2H_4/air, and (c) CH_4/air cases

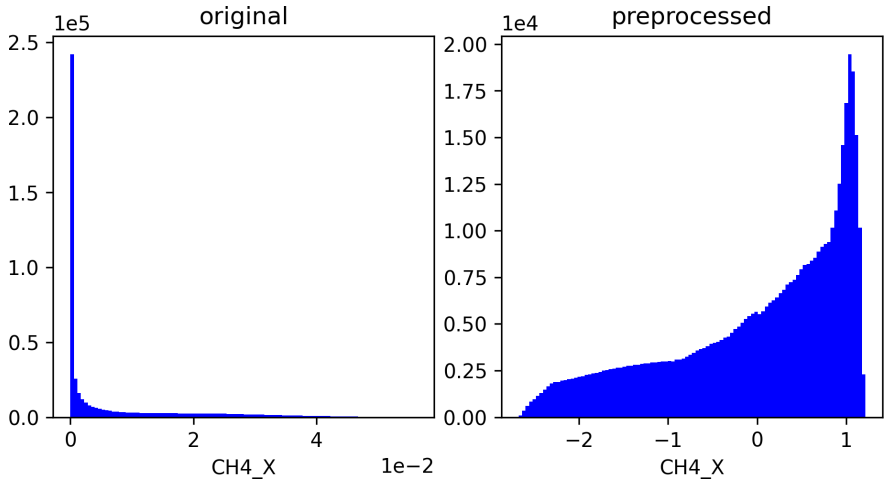
the optimization process, leading to smaller loss function values compared to the *ReLU* activation function. Furthermore, the residual networks with skip-connection structures also improve the optimization results for reactive zones. By employing the selected strategy in this work, which is a residual neural network model with the *swish* activation function, we achieve the lowest loss values during the optimization compared to other strategies. Additionally, there is no significant difference between training and validation losses, indicating that there is no overfitting during the training process. Similar conclusions regarding the absence of



(a)

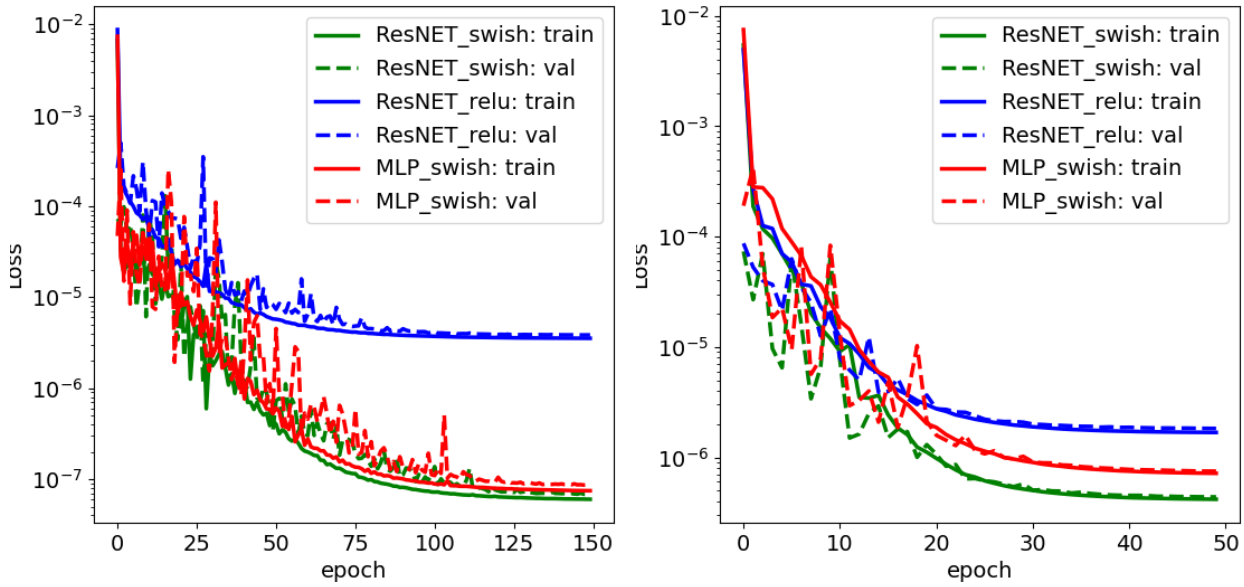


(b)



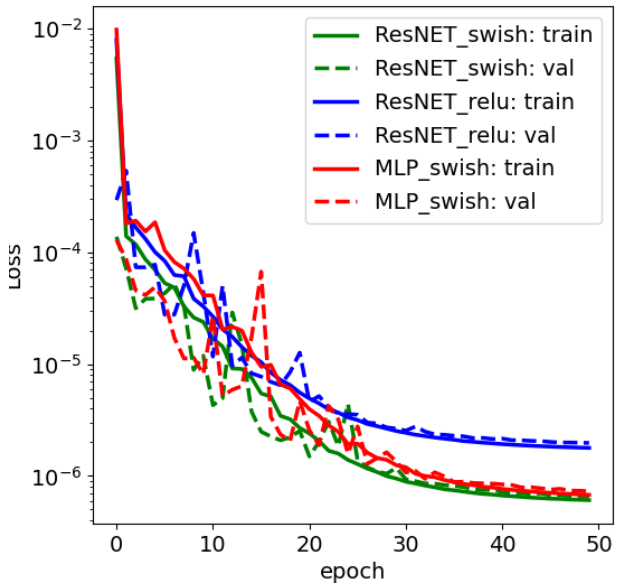
(c)

Figure 8: Distribution of training data from reactive subdomains before and after the preprocessing with nonlinear logarithmic transformation: (a)H_2, (b)C_2H_4, and (c)CH_4.



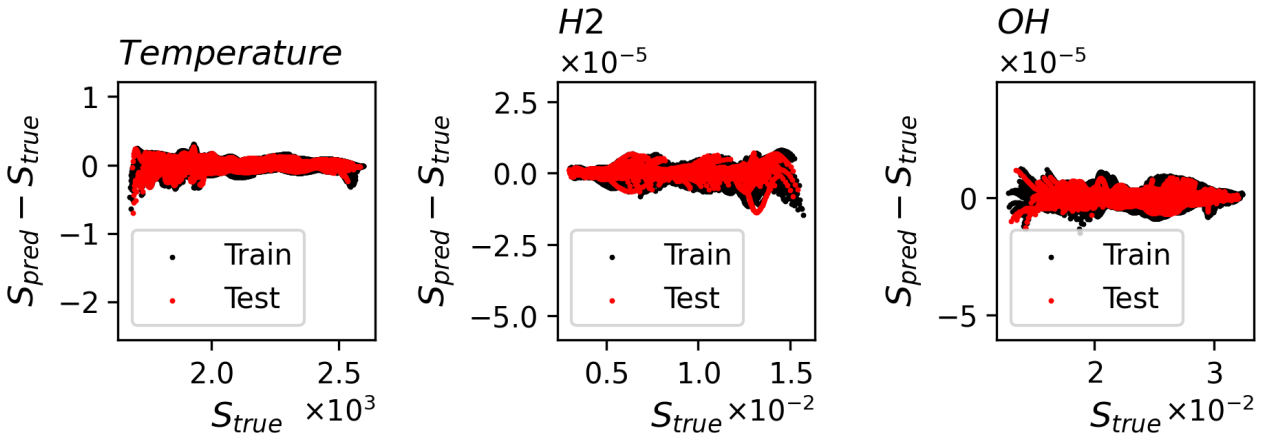
(a)

(b)

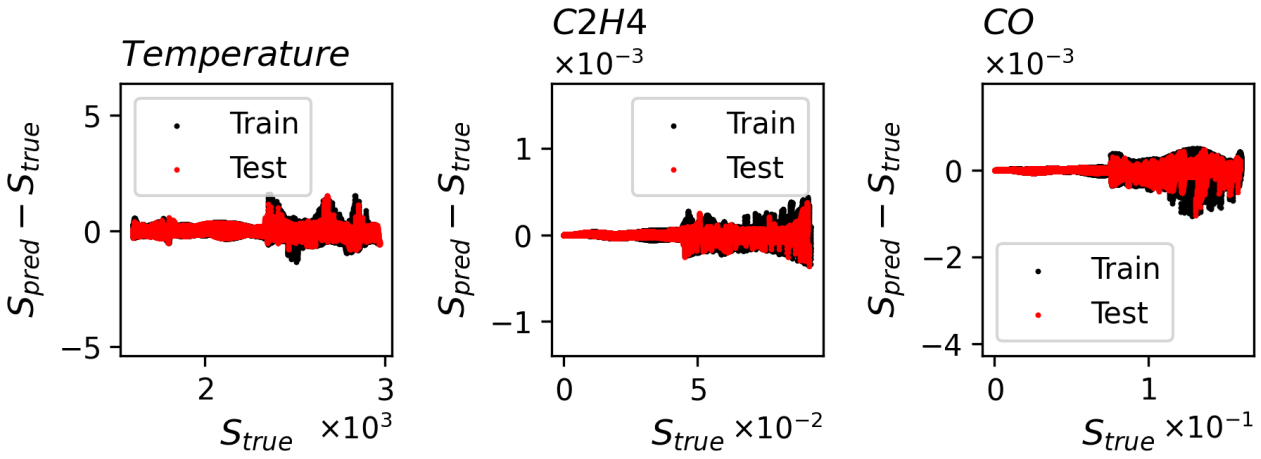


(c)

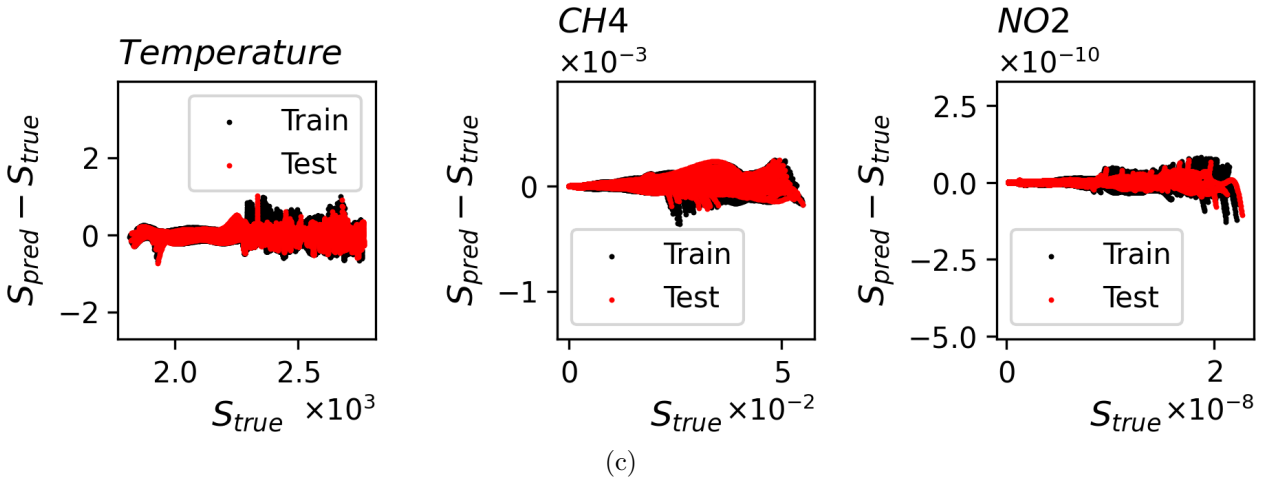
Figure 9: Training and validation loss function values(MSE) over epoch number for models of reactive subdomains for 3 cases (a)H_2/air, (b)C_2H_4/air, and (c)CH_4/air.



(a)



(b)



(c)

Figure 10: Parity plots of true output values and predicted errors for (a)H_2/air, (b)C_2H_4/air, and (c)CH_4/air. The chosen plot elements here are temperature, the fuel and a radical minor species, which are predicted by models of a reactive subdomain.

overfitting can also be drawn from Figure 10, which presents the parity plots of the true output values $\mathbf{S}_{true}$ and the errors between the true and predicted values $\mathbf{S}_{pred} - \mathbf{S}_{true}$. Most of error values between the true and predicted states are much smaller than the state values under physical scales for each dimension.

Tables 2, 3, and 4 provide statistics of $\mathcal{M}$, including the average, minimum, and maximum values, for inference simulations of 100 initial conditions in the test set. The total sizes of the trained models are 330 kB, 3.2 MB, and 3.8 MB for the H_2, C_2H_4, and CH_4 cases, respectively. Box plots in Figures 11, 12, and 13 illustrate the statistical distribution of temperature and species mass fractions. The boxplots represent the distribution of the errors $\mathcal{M}_i$ for each chemical species and of $\mathcal{M}_0$ for temperature. The errors are computed for 100 test simulations and the y axis denotes the error values. The inner box extends from the first quartile $Q1$ to the third quartile $Q3$ of the distribution, while the whiskers extend from $Q1\check{}1.5 * IQR$ to $Q3 + 1.5 * IQR$ where the interquartile range IQR equals $Q3 - Q1$. for the black dots are determined to be outliers which represent the highest and lowest values of the error distribution. From the overall statistical results, it is observed that methods with multiple models are more efficient compared to the baseline method with only one cluster. However, in some cases where the mean error $\mathcal{M}$ is low, there are still a few extreme error values. This observation highlights the issue of trajectory divergence from the correct path during the simulation, which can be attributed to the limited robustness of the local models. The optimal number of clusters leading to the best performance, with the lowest $\mathcal{M}$, in the predefined workflow is found to be 6, 4, and 5 for the H_2, C_2H_4, and CH_4 cases, respectively. The data is divided into different subdomains, including preheated mixing zones with extreme values skewed towards zero, stiff reactive zones, and burn-up zones. The best models for each fuel will be considered in the subsequent analysis. Figure 14 demonstrates the partitioned clusters represented by different colors on the manifold of mass fractions of O_2 and H_2O in logarithmic scale over the progress variable defined by $c = \frac{T-T_0}{T_{\text{eq}}-T_0}$. It is evident that the total manifolds are piecewise separated into subdomains.

Each subdomain exhibits reduced complexity, enabling the construction of learning models with simpler structures and fewer parameters for all subdomains.

Table 2: Statistics for experiments of H_2/air case

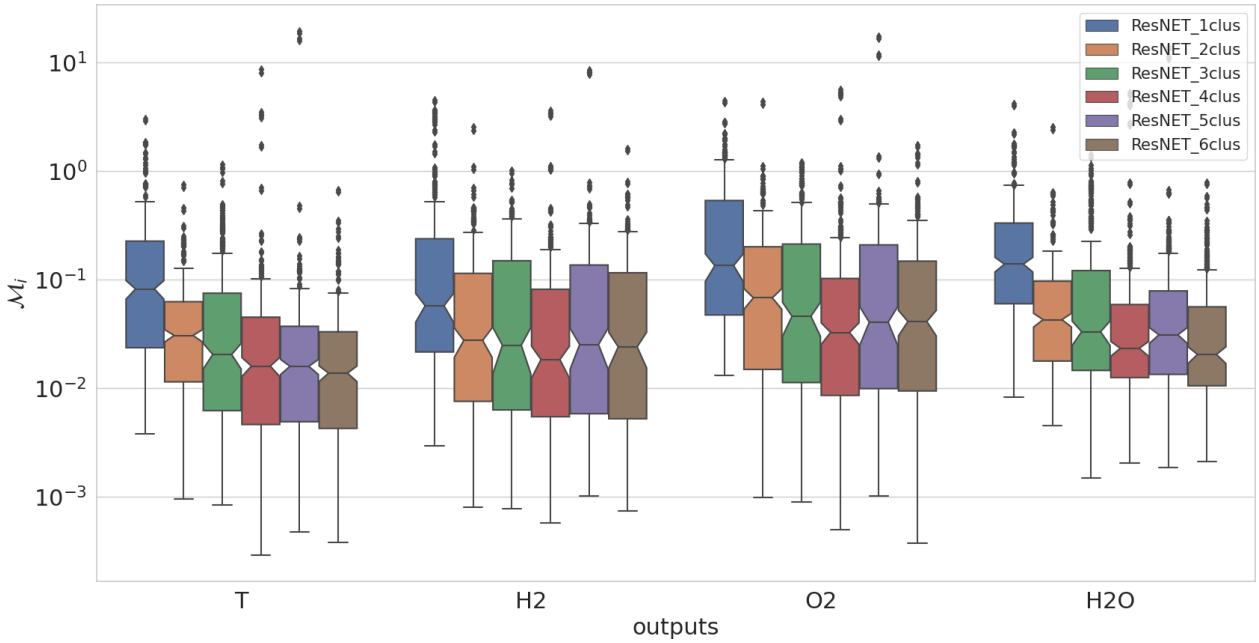
Cluster number	n_r	n_e	mean $\mathcal{M}(\%)$	minimum $\mathcal{M}(\%)$	maximum $\mathcal{M}\%)$
1	1	120	0.301	0.043	2.913
2	1	85	0.097	0.016	1.446
3	1	70	0.117	0.020	1.056
4	1	60	0.068	0.010	3.571
5	1	54	0.081	0.023	11.87
6	1	49	0.068	0.012	0.643

Table 3: Statistics for experiments of C_2H_4/air case

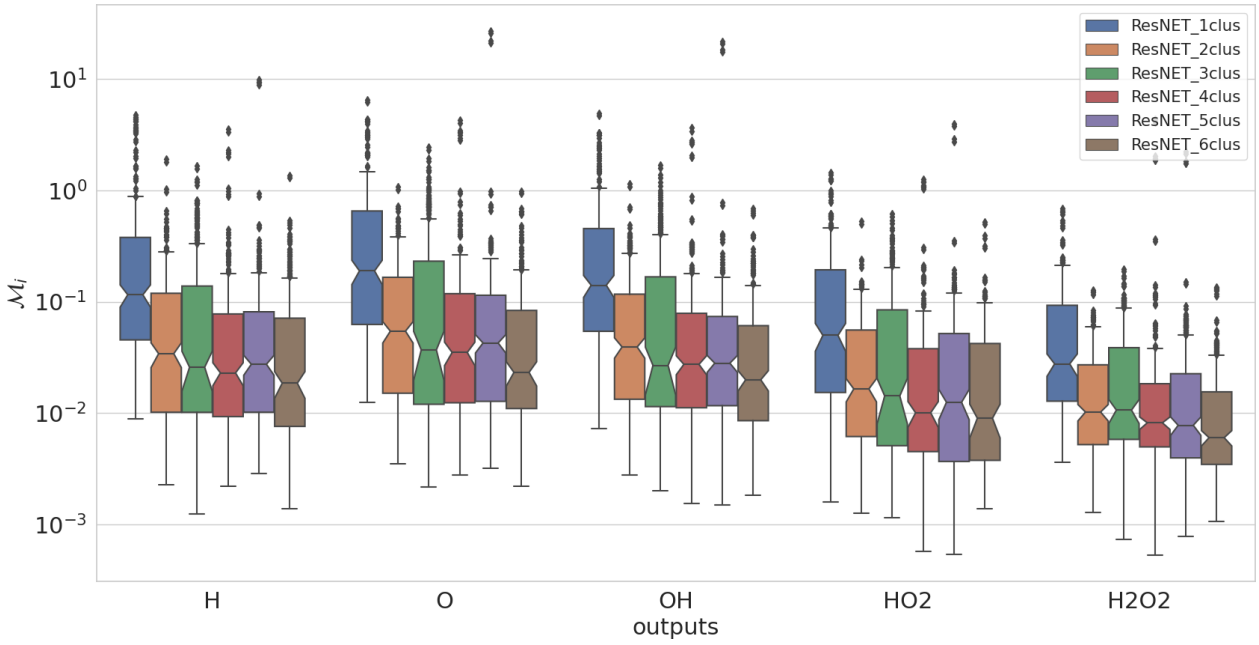
Cluster number	n_r	n_e	mean $\mathcal{M}(\%)$	minimum $\mathcal{M}(\%)$	maximum $\mathcal{M}(\%)$
1	2	300	0.971	0.047	45.30
2	2	212	0.366	0.070	3.571
3	2	174	0.039	0.009	187.94
4	2	150	0.033	0.010	0.249
5	2	135	0.040	0.010	0.312
6	2	123	0.043	0.010	0.179

Table 4: Statistics for experiments of CH_4/air case

Cluster number	n_r	n_e	mean $\mathcal{M}(\%)$	minimum $\mathcal{M}(\%)$	maximum $\mathcal{M}(\%)$
1	2	350	0.947	0.199	6.907
2	2	248	0.882	0.138	9.391
3	2	202	0.332	0.036	4.421
4	2	175	0.153	0.042	2.159
5	2	157	0.152	0.033	1.102
6	2	143	0.173	0.030	1.583



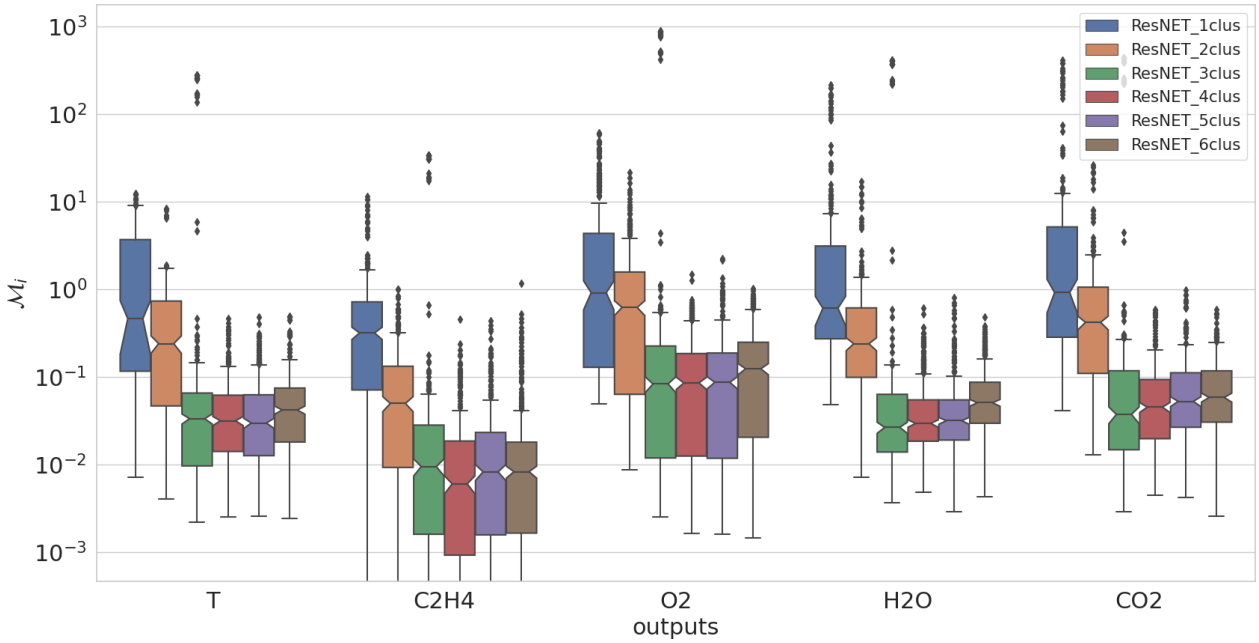
(a)



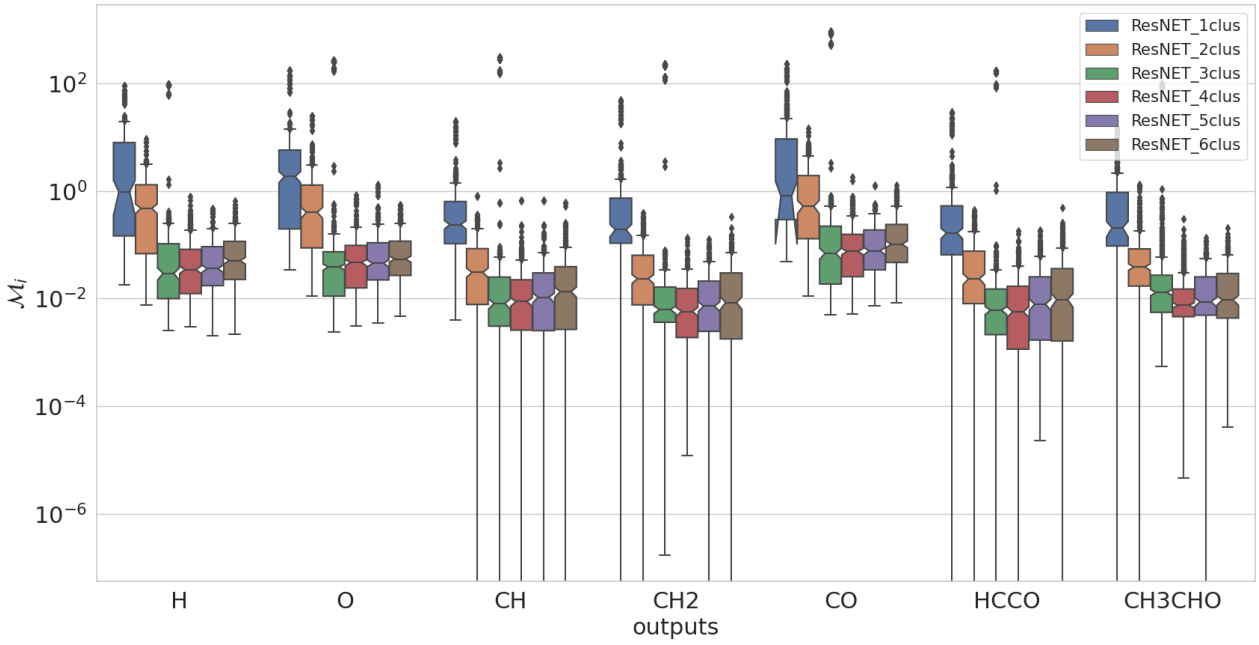
(b)

Figure 11: Box plot of statistical logMAPE errors for 100 a posteriori test simulations of H_2/air case with (a)temperature and major chemical species and (b)several minor chemical species

Furthermore, additional evaluations of iterative predictions are conducted by increasing the number of simulations within the predefined range of initial conditions. In this case,



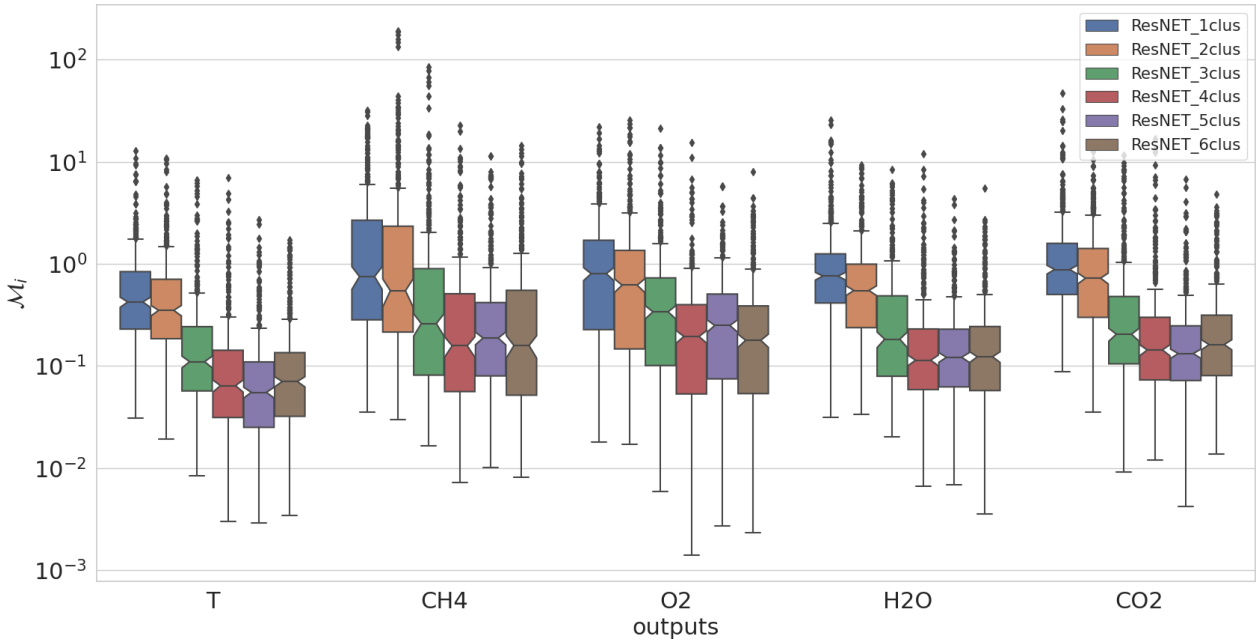
(a)



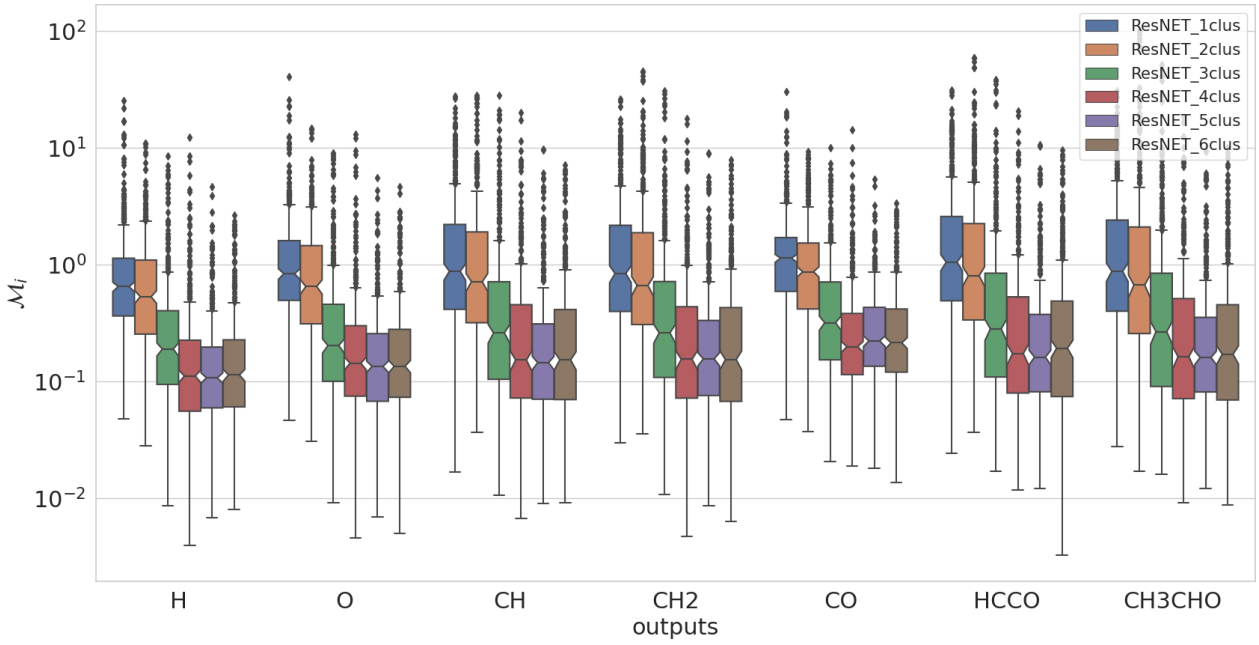
(b)

Figure 12: Box plot of statistical logMAPE errors for 100 a posteriori test simulations of C_2H_4/air case with (a)temperature and major chemical species and (b)several minor chemical species

448 1680 simulations based on model predictions are generated, and the overall accumulative
449 logarithmic Mean Average Percentage Error (logMAPE) (9) is computed for each simulation.



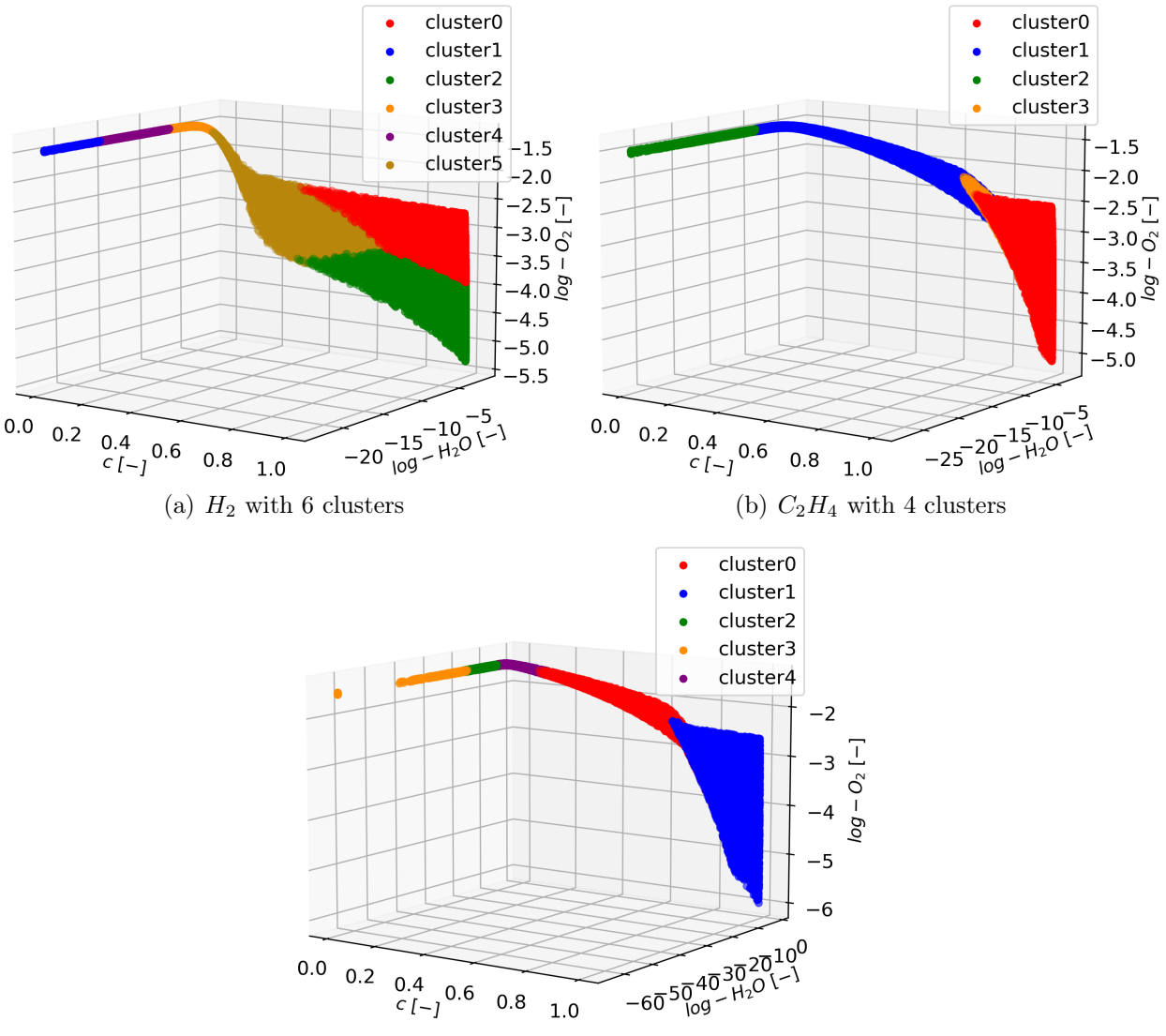
(a)



(b)

Figure 13: Box plot of statistical logMAPE errors for 100 a posteriori test simulations of CH_4/air case with (a)temperature and major chemical species and (b)several minor chemical species

Cubic interpolation is utilized to generalize the 2D error distribution functions. The design of the experiment with a large number of initial conditions is fixed within the same predefined



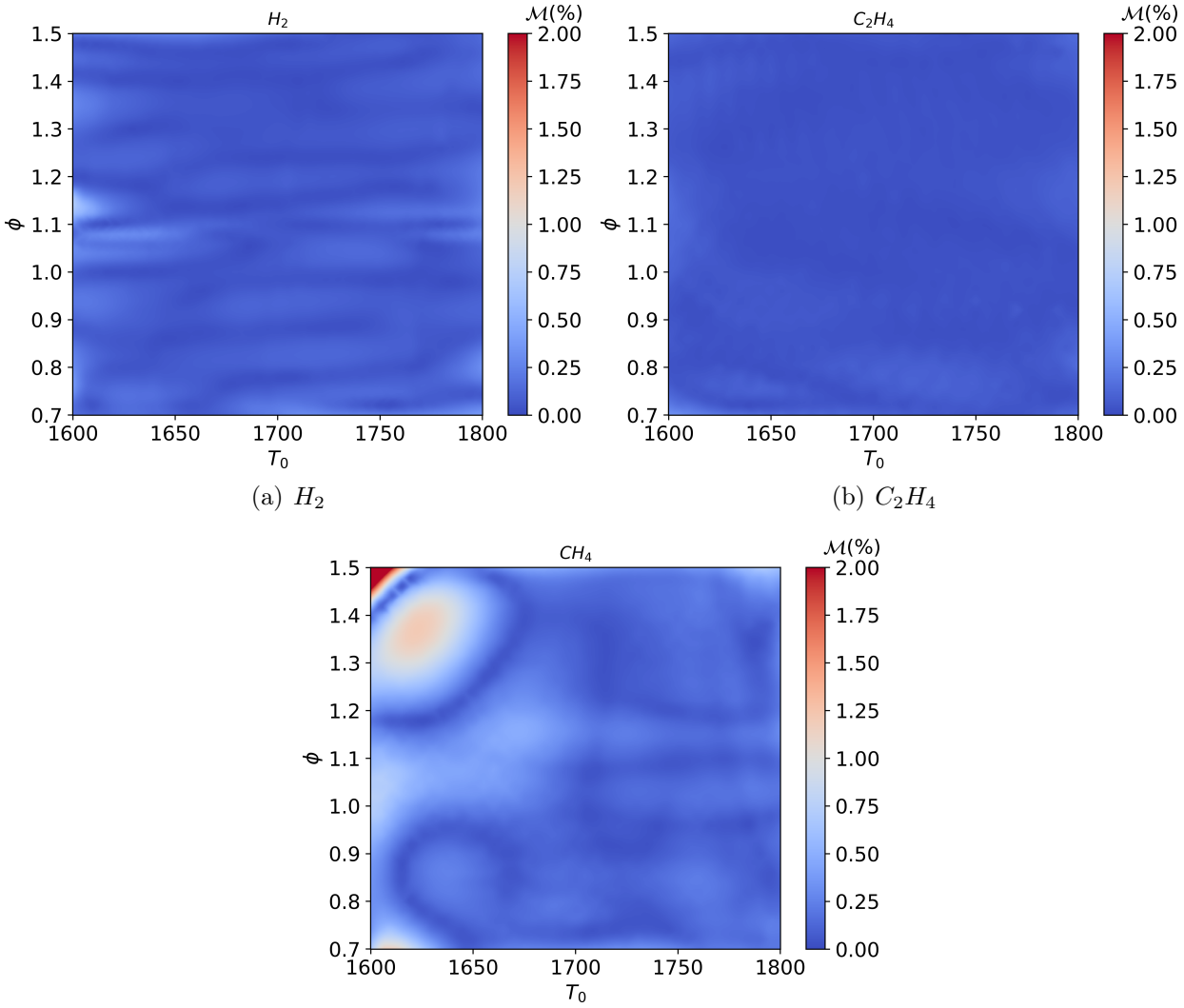
(a) H_2 with 6 clusters

(b) C_2H_4 with 4 clusters

(c) CH_4 with 5 clusters

Figure 14: 3D clustering plots of manifolds for mass fractions of O_2 and H_2O over the evolution of progress variable: (a)H_2/air, (b)C_2H_4/air, and (c)CH_4/air.

range that was used for training the models, with regular samplings of T_0 and ϕ at intervals of $\Delta T_0 = 5K$ and $\Delta\phi = 0.01$. The distribution of overall average MAPE errors is depicted in Figure 15. It can be observed that simulations based on the best models exhibit overall average MAPE errors lower than 1%, with a slight decrease in accuracy near the boundaries of the domain. Due to its higher complexity, the CH_4/air case exhibits larger overall logMAPE errors, but they are still within an acceptable range.



(a) H_2

(b) C_2H_4

(c) CH_4

Figure 15: 2D distribution of mean relative errors of all dimensions for (a)H_2/air, (b)C_2H_4/air, and (c)CH_4/air cases within designed initial condition zone, using refined initial conditions sampling and cubic interpolation

5.3 Comparison of sampling strategies

The results of comparing predictions using regular sampling and the CVODE-based sampling method are shown in Table 5. The chosen clusters are based on the optimal clusters for each case, which is 6, 4 and 5 for H_2, C_2H_4 and CH_4. The results demonstrate that the CVODE-based sampling method improves the overall prediction performance, as indicated by smaller mean, minimum, and maximum errors ($\mathcal{M}$) for all three cases. Regular sampling

Table 5: Statistics for experiments of H_2/air, C_2H_4/air and CH_4/air case using regular sampling method and CVODE sampling method

	H_2	C_2H_4	CH_4
Regular sampling			
mean $\mathcal{M}(\%)$	0.170	0.051	0.332
maximum $\mathcal{M}(\%)$	1103.2	0.434	1.880
minimum $\mathcal{M}(\%)$	0.018	0.019	0.063
CVODE sampling			
mean $\mathcal{M}(\%)$	0.069	0.033	0.152
maximum $\mathcal{M}(\%)$	0.643	0.249	1.102
minimum $\mathcal{M}(\%)$	0.012	0.010	0.034

can lead to unbalanced clustering and uneven data distribution, resulting in unstable iterative predictions, as observed in Table 5 for the H_2 case. By employing adaptive time steps, the CVODE-based method generates more data points from fast reaction regions, leading to a more robust unsupervised clustering model with a balanced distribution of data points.

5.4 Simulation of 0D reactors using DNN

To illustrate the prediction of specific trajectories, two simulations with different initial conditions near the boundaries of the dataset were performed. One simulation was conducted with a low temperature of $1620.0K$ and an equivalence ratio of 0.75, while the other simulation had a high temperature of $1790.0K$ and an equivalence ratio of 1.45. The results for temperature are shown in Figure 16 for the three fuels. It can be observed that the predictions generated by the models closely match the results of the direct numerical simulations. Additional inference results which show the predictions for a subset of the chemical species, and results which display the predictions in logarithmic predictive space, with clustering zones marked by different colors can be found in supplementary materials. The DNN results exhibit good agreement with the direct numerical simulations in both the auto-ignition zones and equilibrium zones, compared to the exact numerical simulations. The cluster index of states is predicted after unsupervised classification by the clustering algorithm, resulting in

states belonging to different subdomains at different time steps. The predictions in logarithmic space are accurate within each partitioned subdomain of states, and the state values skewed toward zero are accurately predicted at the start of the simulations.

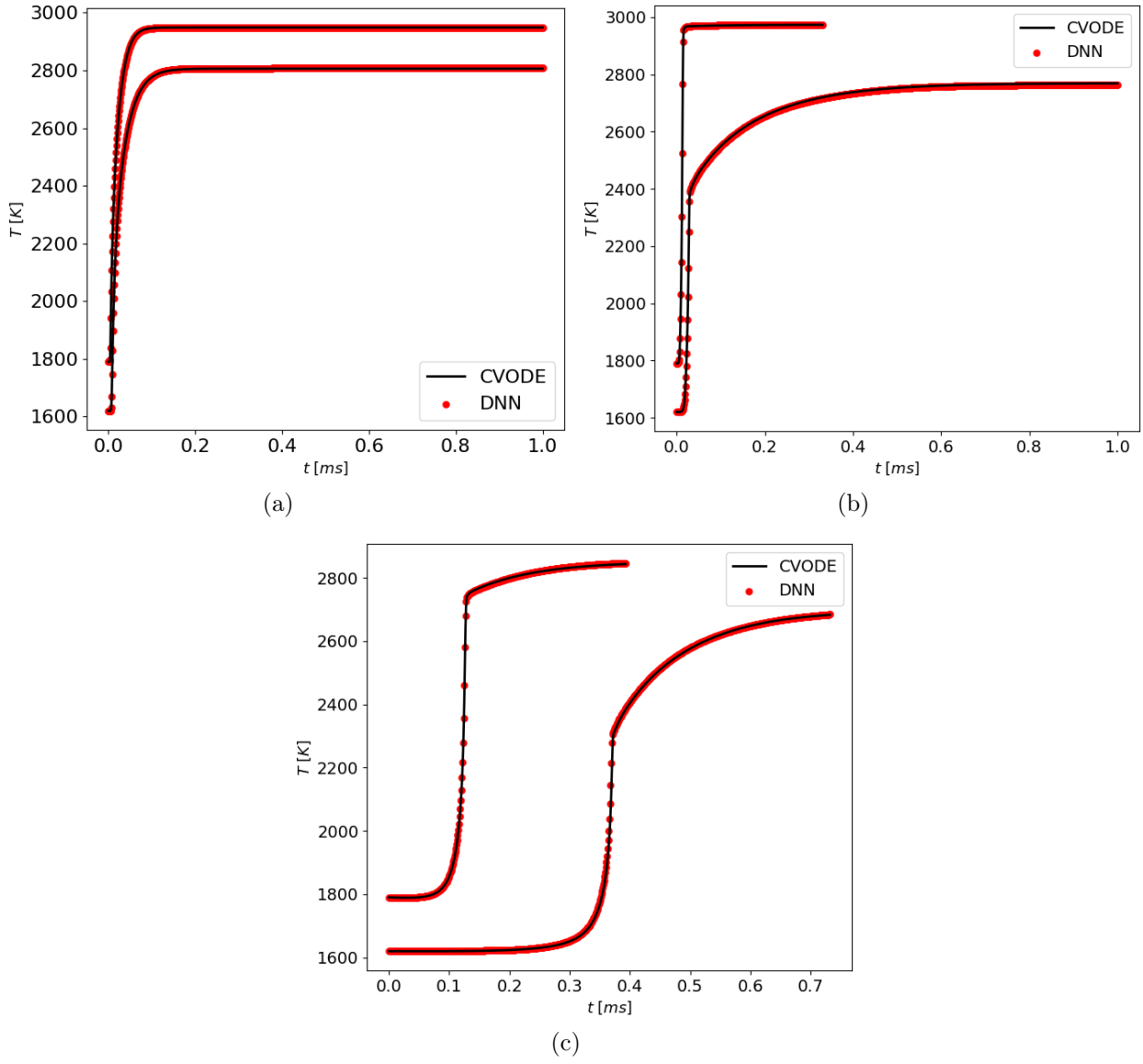


Figure 16: Continuous inference simulation of (a)H_2/air, (b)C_2H_4/air, and (c)CH_4/air cases for temperature with 2 trajectories: $T_{01} = 1620.0K$,$\phi_1 = 0.75$ and $T_{02} = 1790.0K$,$\phi_2 = 1.45$.

5.5 Computing performance

The performance of the DNNs in terms of computational cost is assessed in this section. A single 0-D reactor is computed and the cost of the simulation is analyzed. The Python framework used in the above DNN study is not adapted to performance analysis as the Tensorflow DNN inference involves overhead costs. Tests on computational speed are therefore performed using the NNICE[49] library, which is an in-house C++ code providing lightweight DNN inference capabilities without relying on Machine Learning libraries C++ APIs. The NNICE library is easily coupled to any CFD code and is relevant for estimating gains which can be expected by replacing a direct integrator with DNNs.

The CVODE and DNN integrations are timed at each time step. We focus here on the C_2H_4 case to demonstrate the decrease in computational cost using DNNS. The selected initial condition is $T_0 = 1700.0K$ and $\phi = 1.0$. We use here 4 clusters, as it has been shown to lead to the best results for C_2H_4. Several DNN architectures, summarized in Table 6, are tested to illustrate the impact of DNN size on acceleration. We verified that all networks lead to satisfactory results. Figure 17 shows the acceleration ratio t_{cvode}/t_{DNN} for each iteration during the simulation, where t_{cvode} and t_{DNN} are the time spent in CVODE and DNN integration, respectively.

The results clearly demonstrate that the chemistry integration performed by the DNN is 5 to 30 times faster than CVODE. More specifically, DNN predictions for both stiff (from $0.0s$ to approx. $4 \times 10^{-5}s$) and non-stiff regions exhibit the same computing efficiency, whereas the CVODE solver incurs higher costs due to the use of implicit schemes involving Jacobian matrix computations. This leads to a significant gain in computational cost for the most reactive states, as seen in Figure 17 in the first instants of the simulation, corresponding to the ignition phase. Moreover, the computation acceleration is enhanced when the size of the network decreases. This suggests that for an optimal computing efficiency, a hyper-parameter search on the network size could bring benefits.

Table 6: Models information for DNN performance testing

DNN model	n_r	n_e	model size for each cluster
DNN_1	1	150	462.7 kB
DNN_2	2	120	554.7 kB
DNN_3	2	150	830.9 kB

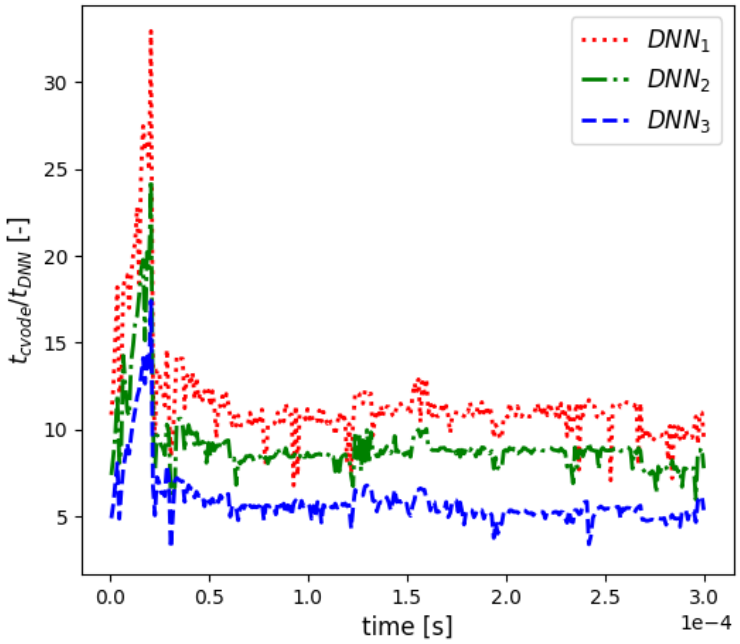


Figure 17: The acceleration ratio t_{cvode}/t_{DNN} between CVODE resolution and DNN prediction resolution for each time step, curves for three models with different sizes.

6 Conclusion and perspective

In this research, a deep learning surrogate model has been successfully trained and applied to the predictions of 0D combustion simulations using different fuels of increasing complexity. Several key strategies have been employed, including a new sampling method based on adaptive time steps of the CVODE solver to achieve a balanced data distribution, the application of a non-supervised K-Means algorithm to separate the dataset into subdomains in logarithmic space, and the use of residual networks to improve optimization during the training process. The analysis of the inference results demonstrates that by partitioning the sampling points into subdomains, the overall complexity of the model prediction process

can be reduced while achieving optimal results. Proper selection of hyperparameters plays a crucial role in obtaining accurate iterative predictions.

As an extension of this work, the proposed sampling method based on implicit numerical stiff solvers can be further applied to sample thermochemical data points in the context of 2D/3D combustion processes involving convection and diffusion terms. This would allow the learning workflow to be extended to more complex problems that are coupled with convection and diffusion in computational fluid dynamics.

A K-Means algorithm

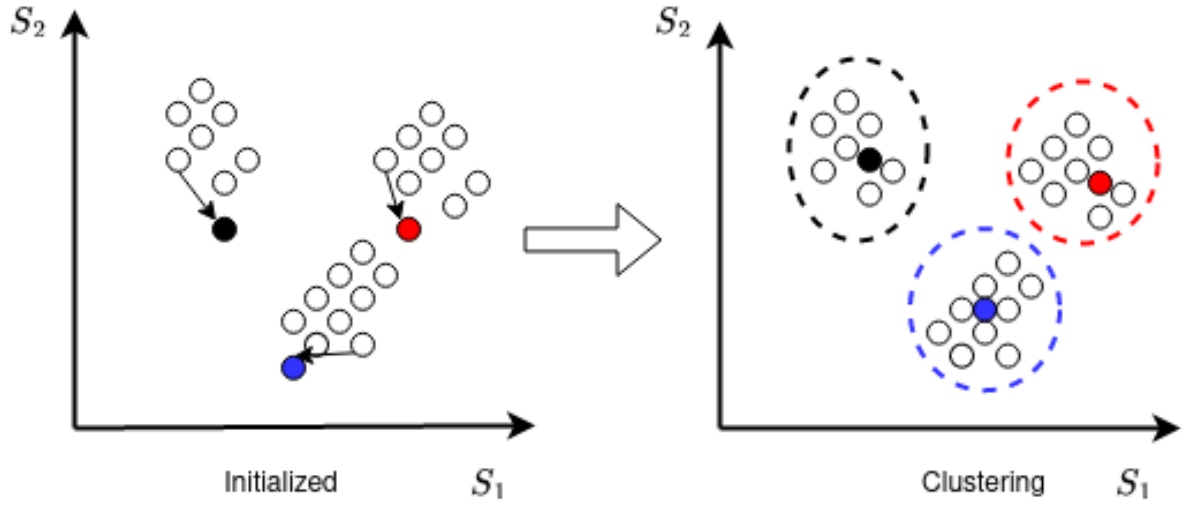


Figure 18: Clustering by K-Means algorithm

The K-Means algorithm, as shown in Figure 18, is a clustering algorithm that partitions a set of N sampling points based on their similarities, aiming to minimize the average squared distance between K centroids and the sampling points. Given a set of N observed sampling points $\mathbf{S}_1, \mathbf{S}_2, ..., \mathbf{S}_N$, the algorithm iteratively minimizes the within-cluster sum of squares (WCSS) as the loss function. The WCSS is defined as the sum of the squared Euclidean

distances between each sampling point and its assigned centroid, which is denoted as:

$$\mathcal{L}_{kmeans} = \sum_{i=1}^{K} \sum_{\mathbf{S} \in \Omega_i} \left\| \hat{\mathbf{S}}_l - \mathbf{d}_i \right\|_2 \tag{11}$$

where the $\mathbf{d}_i$ denotes the centroid of each cluster, $\hat{\mathbf{S}}$ represents the data normalization. Notice that the clustering step is performed in the transformed coordinates after the normalized logarithmic transformation for chemical species Y_j with $\mathbf{S}_l = [T, \ln(Y_j)], j = 1, ..., N_s$.

The K-Means++ algorithm[50] is an improved version of the standard K-Means algorithm that addresses the issue of poor clusterings that can occur with the standard approach. It introduces a more effective initialization step for selecting the initial centroids. The steps of the centroids initialization by the K-Means++ algorithm are as follows:

- Initialize the first centroid by randomly selecting one data point from the dataset using Latin Hypercube Sampling(LHS).
- For each remaining data point, compute its distance to the nearest centroid.
- Select the next centroid from the remaining data points with a probability proportional to the square of the distance to the nearest centroid. This ensures that points further away from existing centroids are more likely to be selected as new centroids.
- Repeat steps 2 and 3 until K centroids are selected.

By using the K-Means++ algorithm for initialization, the K-Means clustering process starts with more representative initial centroids, leading to better overall clusterings. This helps to avoid situations where the algorithm gets stuck in sub-optimal solutions or produces unbalanced clusters.

Supporting Information

Detailed quantitative analysis of thresholds and additional 0D simulation results for several chemical species based on optimal DNN models (PDF)

References

(1) Cohen, S. D.; Hindmarsh, A. C.; Dubois, P. F. CVODE, a stiff/nonstiff ODE solver in C. *Computers in physics* **1996**, *10*, 138–143.

(2) Peters, N.; Rogg, B. *Reduced kinetic mechanisms for applications in combustion systems*; Springer Science & Business Media, 2008; Vol. 15.

(3) Lu, T.; Law, C. K. A directed relation graph method for mechanism reduction. *Proceedings of the Combustion Institute* **2005**, *30*, 1333–1341.

(4) Chen, J.-Y.; Kollmann, W.; Dibble, R. Pdf modeling of turbulent nonpremixed methane jet flames. *Combustion Science and Technology* **1989**, *64*, 315–346.

(5) Pope, S. Computationally efficient implementation of combustion chemistry using in situ adaptive tabulation. *Combustion Theory and Modelling* **1997**, *1*, 41–63.

(6) Goodfellow, I.; Bengio, Y.; Courville, A. *Deep learning*; MIT press, 2016.

(7) Ihme, M.; Chung, W. T.; Mishra, A. A. Combustion machine learning: Principles, progress and prospects. *Progress in Energy and Combustion Science* **2022**, *91*, 101010.

(8) Zhou, L.; Song, Y.; Ji, W.; Wei, H. Machine learning for combustion. *Energy and AI* **2022**, *7*, 100128.

(9) Cybenko, G. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems* **1989**, *2*, 303–314.

(10) Hornik, K.; Stinchcombe, M.; White, H. Multilayer feedforward networks are universal approximators. *Neural networks* **1989**, *2*, 359–366.

(11) Christo, F.; Masri, A.; Nebot, E. Artificial neural network implementation of chemistry with PDF simulation of H2/CO2 flames. *Combustion and Flame* **1996**, *106*, 406–427.

(12) Blasco, J.; Fueyo, N.; Dopazo, C.; Ballester, J. Modelling the temporal evolution of a reduced combustion chemical system with an artificial neural network. *Combustion and Flame* **1998**, *113*, 38–52.

(13) Ihme, M.; Schmitt, C.; Pitsch, H. Optimal artificial neural networks and tabulation methods for chemistry representation in LES of a bluff-body swirl-stabilized flame. *Proceedings of the Combustion Institute* **2009**, *32*, 1527–1535.

(14) Sen, B. A.; Hawkes, E. R.; Menon, S. Large eddy simulation of extinction and reignition with artificial neural networks based chemical kinetics. *Combustion and Flame* **2010**, *157*, 566–578.

(15) 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*, 119–129.

(16) Ranade, R.; Alqahtani, S.; Farooq, A.; Echekki, T. An ANN based hybrid chemistry framework for complex fuels. *Fuel* **2019**, *241*, 625–636.

(17) Ding, T.; Readshaw, T.; Rigopoulos, S.; Jones, W. Machine learning tabulation of thermochemistry in turbulent combustion: An approach based on hybrid flamelet/random data and multiple multilayer perceptrons. *Combustion and Flame* **2021**, *231*, 111493.

(18) An, J.; He, G.; Luo, K.; Qin, F.; Liu, B. Artificial neural network based chemical mech-

anisms for computationally efficient modeling of hydrogen/carbon monoxide/kerosene combustion. *International Journal of Hydrogen Energy* **2020**, *45*, 29594–29605.

(19) Ji, W.; Qiu, W.; Shi, Z.; Pan, S.; Deng, S. Stiff-pinn: Physics-informed neural network for stiff chemical kinetics. *The Journal of Physical Chemistry A* **2021**, *125*, 8098–8106.

(20) Sutherland, J. C.; Parente, A. Combustion modeling using principal component analysis. *Proceedings of the Combustion Institute* **2009**, *32*, 1563–1570.

(21) Isaac, B. J.; Coussement, A.; Gicquel, O.; Smith, P. J.; Parente, A. Reduced-order PCA models for chemical reacting flows. *Combustion and flame* **2014**, *161*, 2785–2800.

(22) Malik, M. R.; Isaac, B. J.; Coussement, A.; Smith, P. J.; Parente, A. Principal component analysis coupled with nonlinear regression for chemistry reduction. *Combustion and Flame* **2018**, *187*, 30–41.

(23) Zdybał, K.; Sutherland, J. C.; Parente, A. Manifold-informed state vector subset for reduced-order modeling. *Proceedings of the Combustion Institute* **2023**, *39*, 5145–5154.

(24) Zhang, Y.; Xu, S.; Zhong, S.; Bai, X.-S.; Wang, H.; Yao, M. Large eddy simulation of spray combustion using flamelet generated manifolds combined with artificial neural networks. *Energy and AI* **2020**, *2*, 100021.

(25) Prieler, R.; Moser, M.; Eckart, S.; Krause, H.; Hochenauer, C. Machine learning techniques to predict the flame state, temperature and species concentrations in counterflow diffusion flames operated with CH4/CO/H2-air mixtures. *Fuel* **2022**, *326*, 124915.

(26) Hansinger, M.; Ge, Y.; Pfitzner, M. Deep residual networks for flamelet/progress variable tabulation with application to a piloted flame with inhomogeneous inlet. *Combustion Science and Technology* **2022**, *194*, 1587–1613.

(27) Li, K.; Rahnama, P.; Novella, R.; Somers, B. Combining flamelet-generated manifold

and machine learning models in simulation of a non-premixed diffusion flame. *Energy and AI* **2023**, *14*, 100266.

(28) Chi, C.; Janiga, G.; Thévenin, D. On-the-fly artificial neural network for chemical kinetics in direct numerical simulations of premixed combustion. *Combustion and Flame* **2021**, *226*, 467–477.

(29) Han, X.; Jia, M.; Chang, Y.; Li, Y. An improved approach towards more robust deep learning models for chemical kinetics. *Combustion and Flame* **2022**, *238*, 111934.

(30) Blasco, J. A.; Fueyo, N.; Dopazo, C.; Chen, J. A self-organizing-map approach to chemistry representation in combustion applications. *Combustion Theory and Modelling* **2000**, *4*, 61–76.

(31) Franke, L. L.; Chatzopoulos, A. K.; Rigopoulos, S. Tabulation of combustion chemistry via Artificial Neural Networks (ANNs): Methodology and application to LES-PDF simulation of Sydney flame L. *Combustion and Flame* **2017**, *185*, 245–260.

(32) Perini, F. High-dimensional, unsupervised cell clustering for computationally efficient engine simulations with detailed combustion chemistry. *Fuel* **2013**, *106*, 344–356.

(33) Barwey, S.; Prakash, S.; Hassanaly, M.; Raman, V. Data-driven classification and modeling of combustion regimes in detonation waves. *Flow, Turbulence and Combustion* **2021**, *106*, 1065–1089.

(34) Nguyen, H.-T.; Domingo, P.; Vervisch, L.; Nguyen, P.-D. Machine learning for integrating combustion chemistry in numerical simulations. *Energy and AI* **2021**, *5*, 100082.

(35) Zhang, T.; Yi, Y.; Xu, Y.; Chen, Z. X.; Zhang, Y.; Weinan, E.; Xu, Z.-Q. J. A multi-scale sampling method for accurate and robust deep neural network to predict combustion chemical kinetics. *Combustion and Flame* **2022**, *245*, 112319.

(36) Lanser, D.; Verwer, J. G. Analysis of operator splitting for advection–diffusion–reaction problems from air pollution modelling. *Journal of computational and applied mathematics* **1999**, *111*, 201–216.

(37) MacNamara, S.; Strang, G. *Splitting methods in communication, imaging, science, and engineering*; Springer, 2016; pp 95–114.

(38) Goodwin, D. G. Cantera c++ user's guide. *California Institute of Technology* **2002**,

(39) Ó Conaire, M.; Curran, H. J.; Simmie, J. M.; Pitz, W. J.; Westbrook, C. K. A comprehensive modeling study of hydrogen oxidation. *International journal of chemical kinetics* **2004**, *36*, 603–622.

(40) Luo, Z.; Yoo, C. S.; Richardson, E. S.; Chen, J. H.; Law, C. K.; Lu, T. Chemical explosive mode analysis for a turbulent lifted ethylene jet flame in highly-heated coflow. *Combustion and Flame* **2012**, *159*, 265–274.

(41) Smith, G. GRI-Mech.-An Optimized Detailed Chemical Reaction Mechanism for Methane Combustion. *http://www. me. berkeley. edu/gri_ mech* **1999**,

(42) He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. Proceedings of the IEEE conference on computer vision and pattern recognition. 2016; pp 770–778.

(43) Qu, J.; Faney, T.; de Hemptinne, J.-C.; Yousef, S.; Gallinari, P. PTFlash : A vectorized and parallel deep learning framework for two-phase flash calculation. *Fuel* **2023**, *331*, 125603.

(44) Ramachandran, P.; Zoph, B.; Le, Q. V. Searching for activation functions. *arXiv preprint arXiv:1710.05941* **2017**,

(45) Abadi, M. et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. 2015; `https://www.tensorflow.org/`, Software available from tensorflow.org.

(46) Kingma, D. P.; Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* **2014**,

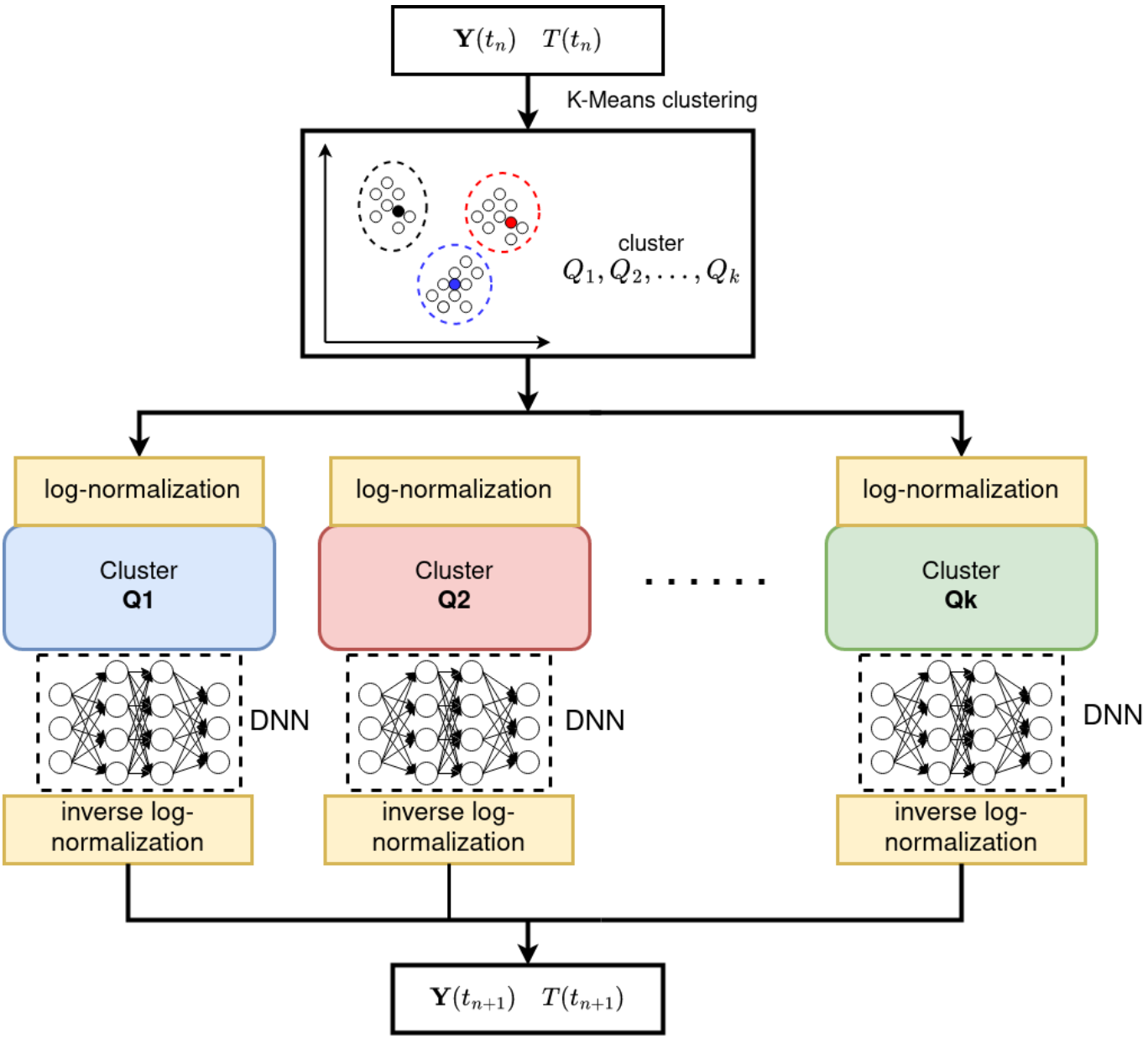
(47) You, K.; Long, M.; Wang, J.; Jordan, M. I. How does learning rate decay help modern neural networks? *arXiv preprint arXiv:1908.01878* **2019**,

(48) Sharma, A. J.; Johnson, R. F.; Kessler, D. A.; Moses, A. Deep learning for scalable chemical kinetics. AIAA scitech 2020 forum. 2020; p 0181.

(49) Aubagnac-Karkar, D.; Mehl, C. NNICE: Neural Network Inference in C made Easy. 2023; `https://doi.org/10.5281/zenodo.7645515`.

(50) Arthur, D.; Vassilvitskii, S. K-means++ the advantages of careful seeding. Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. 2007; pp 1027–1035.

TOC graphic


$\mathbf{Y}(t_n)$ $T(t_n)$
K-Means clustering
cluster
$Q_1, Q_2, \ldots, Q_k$
log-normalization
log-normalization
log-normalization
Cluster
Q1
Cluster
Q2
Cluster
Qk
DNN
DNN
DNN
inverse log-normalization
inverse log-normalization
inverse log-normalization
$\mathbf{Y}(t_{n+1})$ $T(t_{n+1})$