Combustion and Flame 220 (2020) 133–143

Contents lists available at ScienceDirect

Combustion and Flame

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

# Data-driven selection of stiff chemistry ODE solver in operator-splitting schemes

Simon Lapointe*, Sudeepta Mondal, Russell A. Whitesides

*Lawrence Livermore National Laboratory, Livermore, CA 94550, USA*

ARTICLE INFO

*Article history:*
Received 8 February 2020
Revised 22 June 2020
Accepted 23 June 2020
Available online 8 July 2020

*Keywords:*
Ordinary differential equation
Chemical kinetics
Operator-splitting
Computational fluid dynamics

ABSTRACT

Most computational fluid dynamics simulations of practical combustion applications employ operator-splitting schemes, where chemistry and transport are separated and integrated with distinct numerical methods. The changes in composition due to chemistry are evaluated by solving ordinary differential equations (ODE) in each cell of the computational domain, which typically dominates the computational cost when detailed chemistry is considered. In this work, a data-driven approach for the selection of chemistry ODE solvers in operator-splitting schemes is presented. Neural networks are used to predict the ODE solvers CPU times and errors for a given thermochemical state. This allows the selection of an optimal ODE solver on a cell-by-cell, timestep-by-timestep basis. The models are trained using a wide set of thermochemical states generated through partially-stirred reactors and flames simulations. The methodology is validated by quantifying the prediction errors, the classification accuracy, and the computational speedup. The model predicts the optimal ODE solver for 70 to 95% of the validation cases and decreases the computional cost by a factor of 3 or more. The generalizability of the methodology to different chemical mechanisms and different fuels is assessed and it is shown that the model's performance is only slightly degraded and its applicability is significantly enhanced if the inputs to the neural networks are restricted to a small set of thermochemical state variables present in most chemical mechanisms. The models are used in an homogeneous reactor case and a multi-dimensional CFD simulation of a diesel spray at high pressure where a speedup of more than 3 is achieved.



## 1. Introduction

There is a need to accomodate realistic chemistry in engine simulations to accurately predict ignition behavior and pollutant formation of practical transportation fuels and promising biofuels [1]. However, reactive flow simulations with detailed chemistry can be prohibitively expensive due to the size of detailed reaction mechanisms and the numerical stiffness caused by the broad range of chemical time scales. Kinetic mechanisms for transportation fuels can contain thousands of species and reactions [2–4] and the fastest time scales are typically nanoseconds or smaller.

In many reacting flow codes, especially those used for engine simulations, an operator-splitting strategy [5–9] is used where chemistry and transport are separated and integrated with distinct numerical methods. The chemistry contribution results in a system of ordinary differential equations at each cell of the computational domain which is integrated using a stiff ODE solver.

The evaluation of the chemical evolution typically dominates the computational cost of engine simulations and improving the integration of the stiff chemical kinetic ODEs has been the subject of numerous research efforts [10–14]. Implicit backward differentiation formula (BDF) methods [15] using sparse Jacobian matrices [12,16] as preconditioners are commonly used but other approaches such as extrapolation methods [13,17] and exponential methods [14,18] have also been employed.

Determining the optimal ODE solver for combustion simulations is not a trivial task. Different ODE solvers can be slower or faster depending on the local thermochemical state, flow timescales, and user-specified solver settings (such as error tolerances). For example, Imren and Haworth [13] showed that BDF methods perform well for long integration times (i.e., large CFD timesteps in operator-splitting methods) but are outperformed by extrapolation-based methods (such as SEULEX [19]) for short integration times (i.e., small CFD timesteps). Similarly, implicit methods are well suited for stiff ODEs but are slower than explicit methods for nonstiff problems.

In practice, users typically use heuristics to choose a single ODE solver with a single set of solver settings to perform a CFD

* Corresponding author.
*E-mail address:* lapointe2@llnl.gov (S. Lapointe).

https://doi.org/10.1016/j.combustflame.2020.06.033
0010-2180/

simulation. However, it could be beneficial to use different ODE solvers with different settings depending on the local thermochemical state and CFD timestep. For example, regions of low chemical activity such as fully burnt and unburnt gases may be integrated more efficiently with a different integrator than chemically active regions such as autoigniting mixtures and flame reaction zones.

Previous studies on ODE integrator selection for chemical kinetics [20–22] used stiffness metrics to switch between explicit and implicit solvers. The stiffness metrics specific to chemical kinetics rely on the species chemical source terms [20] or the eigenvalues of the system Jacobian [23,24]. However, a recent study comparing different stiffness metrics and ODE integrators computational costs [22] found that stiffness metrics generally do not provide enough information to predict the computational cost of integrating a specific thermochemical state. The integrators' accuracy requirements (*i.e.* error control/tolerances) impact the computational cost significantly and vary depending on the integrator. This suggests that integrator types and tolerances need to be considered in addition to chemical stiffness for optimal integrator selection. A data-driven approach where a large number of thermochemical states are advanced with different ODE integrators (using different tolerances) has the potential to provide more accurate computational cost predictions than empirical stiffness metrics. To the extent of the authors' knowledge, data-driven approaches for the selection of chemical kinetics ODE integrators have not been previously published.

The goal of this work is to describe a data-driven approach to choose the optimal ODE solver on a cell-by-cell, timestep-by-timestep basis. The optimal integrator is defined here as that which provides the lowest CPU time, subject to a user-selected error threshold. A large set of thermochemical states is generated by sampling partially-stirred reactors and laminar flames. Different ODE integrators are considered and neural networks are trained to predict the optimal integrator for a given thermochemical state. The performance of the neural networks is assessed by quantifying the classification accuracy, the types of misclassifications, and the speedpup achieved. The generalizability of the present approach to different mechanisms and fuels is assessed and, finally, the concept is demonstrated in a CFD simulation of a high-pressure reacting diesel spray.

The paper is organized as follows. Section 2 presents the methodology used to generate the training dataset, the stiff ODE solvers considered, the neural networks characteristics, and the conditions and methodology used for the CFD simulations. The results are presented in Section 3 which is divided into three subsections: first, the performance of the neural networks is quantified on a validation dataset; second, the generalizability of the methodology to different chemical mechanisms and fuels is assessed; finally, the on-the-fly ODE solver selector is used in simulations of a 0-D homogeneous reactor and a 3-D diesel spray flame. Conclusions and future work are summarized in Section 4.

## 2. Methodology

### 2.1. Thermochemical states dataset

A dataset of thermochemical states, consisting of species mass fractions, temperature, and pressure $[Y_1, Y_2, \ldots, Y_n, T, p]$, is generated using zero-dimensional constant pressure partially stirred reactor (PaSR) simulations [25,26] and one-dimensional laminar premixed and diffusion flame simulations. The conditions were chosen to span a large range of practical combustion applications, from typical experimental conditions at ambient temperature and pressure to internal combustion engine and gas turbine conditions at elevated temperatures and pressures. The PaSR cases are performed over a range of temperatures, pressures, and residence times for both premixed and diffusion combustion modes. The residence time is varied over multiple orders of magnitude to cover both chemistry-controlled and mixing-controlled regimes. Premixed flame simulations are performed over a range of equivalence ratios, inlet temperatures, and pressures. The equivalence ratio is varied to cover both lean and rich conditions. Diffusion flame simulations are performed over a range of scalar dissipation rates, pressures, and inlet and oxizider temperatures. The conditions of the PaSR and flame simulations are summarized in Table 1.

**Table 1**
Summary of the PaSR and flame simulations used to generate the dataset of thermochemical states. $p$ is the pressure, $T$ is the temperarure, $\phi$ is the equivalence ratio, $\tau_{\text{res}}$ is the residence time, $T_u$ is the unburnt temperature, $\chi_{st}$ is the stoichiometric scalar dissipation rate, $T_f$ is the fuel temperature, and $T_o$ is the oxidizer temperature.

| Simulation | Combust. mode | Conditions | Values used |
|---|---|---|---|
| PaSR | Premixed | $p$ (Bar) | 1, 5, 10, 25, 50, 100 |
| | | $T$ (K) | 300, 400, 500, 600, 700, 800 |
| | | $\phi$ | 0.5, 1.0, 2.0 |
| | | $\tau_{\text{res}}$ (μs) | 30, 100, 300, 1000 |
| PaSR | Diffusion | $p$ (Bar) | 1, 5, 10, 25, 50, 100 |
| | | $T$ (K) | 300, 400, 500, 600, 700, 800 |
| | | $\tau_{\text{res}}$ (μs) | 3, 10, 30, 100, 300, 1000, 3000, 10000 |
| Flame | Premixed | $p$ (Bar) | 1, 5, 10, 25, 50, 100 |
| | | $T_u$ (K) | 300, 400, 500, 600, 700, 800 |
| | | $\phi$ | 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4 |
| Flame | Diffusion | $p$ (Bar) | 1, 5, 10, 25, 50, 100 |
| | | $\chi_{st}$ (1/s) | 0.1, 1, 2, 5, 10, 20, 30 |
| | | $T_f$ (K) | 300, 400, 500, 600, 700, 800 |
| | | $T_o$ (K) | 300, 400, 500, 600, 700, 800 |

The simulations used to generate the initial training and validation dataset are performed with a single fuel composition (TPRF mixture of iso-octane, *n*-heptane, and toluene at 39, 17, and 44% by mole, respectively) representative of fuels used in gasoline internal combustion engines and a single chemical mechanism (269 species gasoline surrogate mechanism reduced from a comprehensive 2,878 species mechanism [27]). Extensions to different fuels and chemical mechanisms are presented in Sections 3.2 and 3.3. In total, the methodology is used with four different chemical mechanisms and similar performance is observed in all cases. The Zero-RK software package [12,28] is used to perform the 0-D simulations and flame simulations are conducted with the solvers described in [29].

To avoid an unbalanced distribution of thermochemical states, only states whose $l^2$ norm (where $T$ and $p$ are normalized by 1000 K and 10 bar, respectively) differ by more than a threshold of $10^{-4}$ are kept in the dataset. Omitting this step leads to an over-representation of the initial/unburnt and equilibrium conditions in the dataset. The final dataset contains 2 million thermochemical states, split evenly between PaSR and flames.

### 2.2. ODE solvers

The methodology introduced in this work assumes that operator-splitting is used to isolate the changes in composition due to chemical reactions from the changes due to convection and diffusion. Operator-splitting is computationally attractive since it is generally faster to compute the solution of the split terms separately than to compute the solution when they are treated together. However, it should be noted that this computational advantage comes at the cost of an error introduced by the splitting, commonly referred to as splitting error. Strategies have been devised to control this error and splitting schemes commonly used in combustion simulations are generally stable and exhibit their theoretical order of accuracy [9]. However, it has been reported that

in some cases the splitting errors can grow beyond the schemes' theoretical order of accuracy and sometimes cause unphysical results [30,31]. Users should thus exercise caution and verify that the time integration method used is stable for the simulations considered.

For each cell in the computational domain, the change in composition due to chemistry over a computational timestep $\Delta t_{\text{CFD}}$ is the solution of a system of $n_{\text{species}} + 1$ ODEs for the species mass fraction and temperature. The species ODEs are

$$\frac{dY_i}{dt} = \omega_i, \tag{1}$$

where $Y_i$ and $\omega_i$ are the mass fraction and production rate of species $i$. The temperature equation is

$$\frac{dT}{dt} = -\frac{RT}{\rho c_p} \sum_i \omega_i h_i, \tag{2}$$

for a constant pressure system, or

$$\frac{dT}{dt} = -\frac{RT}{\rho c_v} \sum_i \omega_i \mu_i, \tag{3}$$

for a constant volume system. $R$ is the universal gas constant, $h_i$ and $\mu_i$ are the enthalpy and internal energy of species $i$, respectively, and $c_p$ and $c_v$ are the heat capacities at constant pressure and constant volume, respectively. The ideal gas law is used as the equation of state.

In practical engine applications, the chemistry time scales can range from $10^{-10}$ or smaller to $10^{-3}$ seconds. The timestep used in CFD is typically chosen to resolve transport phenomena and is around $10^{-7}$ – $10^{-5}$ s. Since $\Delta t_{\text{CFD}}$ is orders of magnitude larger than the smallest chemical time scales, implicit ODE solvers are most commonly used to solve Eq. (1).

Two ODE solvers are considered in this work; A BDF method (CVODE [15]) and an extrapolation method (SEULEX [19]). In this work, CVODE is used with an iterative Krylov linear solver and a sparse representation of the chemical Jacobian is used as preconditioner, as detailed by McNenly et al. [12]. The SEULEX implementation is similar to that used by Imren and Haworth [13]. A direct linear solver is used along with the sparse chemical Jacobian. For both CVODE and SEULEX, the SuperLU sparse matrix library [32] is used to factorize the chemical Jacobian and solve the linear system.

For each solver, two sets of tolerances are considered; a default "tight" tolerance setting where the relative and absolute tolerances are set to $10^{-8}$ and $10^{-20}$, and a relaxed tolerance setting where the relative and absolute tolerances are set to $10^{-6}$ and $10^{-12}$. The default tight tolerance setting is chosen to ensure accurate integration in the presence of a large range of scales and a large number of species with very small mass fractions. While the choice of the present solver tolerances is arbitrary, the overall trends and conclusions are not expected to change significantly if different sets of tolerances are considered. Further, while a single set of tolerances could be used, allowing the neural network to choose between different integration methods as well as different tolerances can lead to reduced computational cost without negatively affecting the overall simulation accuracy. One reason for this is that different integration methods can be impacted differently by the relative and absolute error tolerances. Practically, this means that BDF and extrapolation methods using the same error tolerances may not have the same simulation accuracy. This was reported by Imren and Haworth [13]. It is therefore beneficial to consider different tolerances for each solver. Another reason is that the integration of different thermochemical states may be affected by the tolerances differently. For example, it is possible that regions with little chemical activity such as the unburnt and burnt gases can be integrated with relaxed toleranes without negatively affecting the overall solution accuracy. Thus, it is advantageous to consider different tolerances for each solver and use the data-driven approach to determine the appropriate tolerances for a given state.

**Table 2**
Summary of the ODE integrators used.

| Id. | Method | Rel. tol. | Abs. tol |
|---|---|---|---|
| 0 | CVODE | $10^{-8}$ | $10^{-20}$ |
| 1 | CVODE | $10^{-6}$ | $10^{-12}$ |
| 2 | SEULEX | $10^{-8}$ | $10^{-20}$ |
| 3 | SEULEX | $10^{-6}$ | $10^{-12}$ |

In summary, four integrator/tolerance combinations are considered, as listed in Table 2. A small number of ODE solvers and tolerance settings is preferred for simplicity in the present proof of concept. This allows to showcase that the selection methodology can choose between both different methods and different tolerances while keeping the results relatively simple and easy to present and interpret. Furthermore, the inclusion of numerous integration methods presents additional challenges such as optimizing the implementation and any tuning parameters for each integrator. A wider variety of ODE solvers should be considered in future work.

To generate the training dataset, all thermochemical states are advanced by $\Delta t_{\text{CFD}}$ with all integrators. For each state, $\Delta t_{\text{CFD}}$ is sampled randomly from a uniform distribution spanning from $10^{-7}$ to $10^{-4}$ s. The CPU time taken by each integrator to complete the CFD timestep is monitored along with the relative error of each integrator, evaluated by comparing the temperature change ($\Delta T_i$) predicted by integrator $i$ to the temperature change predicted by the reference integrator

$$\epsilon_i = \frac{|\Delta T_i - \Delta T_0|}{|\Delta T_0|}. \tag{4}$$

Integrator 0 (CVODE with tight tolerances) is used as the reference and $\epsilon_0$ is thus zero for all states. It was verified that defining error using the change in the full state vector instead of the change in temperature leads to very similar results.

### 2.3. Neural networks

Neural networks are used to determine the optimal integrator (ODE solver and tolerances) on a cell-by-cell, timestep-by-timestep basis.

Two networks are employed; one network is used to predict the error ($\epsilon_i$) of each integrator, and a second network is used to predict the necessary CPU time of each integrator. The selected integrator is the fastest integrator for which the predicted error is smaller than a user-specified threshold ($\epsilon_{\text{TOL}}$). The user thus only needs to specify $\epsilon_{\text{TOL}}$ and the neural networks will choose the integrator type and tolerances. For both networks, the inputs are the thermochemical states and the CFD timestep [$Y_1$, $Y_2$, …, $Y_n$, $T$, $p$, $\Delta t_{\text{CFD}}$]. The logarithm of the species mass fractions and the logarithm of the CFD timestep are used since these inputs vary over a wide range of scales. The species mass fractions were clipped to $10^{-20}$ prior to computing the logarithm. All inputs are normalized by subtracting their mean value and scaling to unit variance. For the outputs, the logarithm of the error is used since the error varies by multiple orders of magnitude and no transformation is applied to the CPU time.

The networks consist of multiple fully-connected dense layers. The error-predictor network consists of 3 hidden layers with 1024, 512, and 256 nodes, respectively and the CPU time-predictor consists of 5 hidden layers (containing 256, 256, 256, 256, and 128 nodes). The ReLU activation function is used for all hidden layers. The number of hidden layers and hidden nodes for each net-

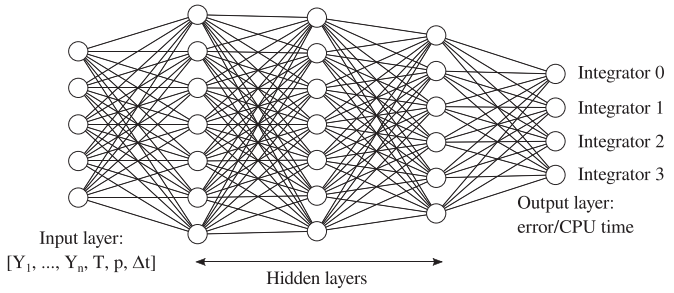



**Fig. 1.** Schematic illustrating the architecture of the neural networks.

**Table 3**
Spray A simulation conditions.

| Fuel | *n*-dodecane |
|---|---|
| Fuel temperature (K) | 363 |
| Nozzle diameter (μm) | 90 |
| Injection duration (ms) | 1.5 |
| Ambient temperature (K) | 900 |
| Ambient density (kg/m$^3$) | 22.8 |
| Ambient $O_2$ (%) | 15 |

work were determined through trial and error with the objective of minimizing the validation error. Larger (both deeper and wider) networks did not significantly reduce the validation error. Figure 1 illustrates the architecture of the networks used in this work.

In addition to the conventional networks described above, a neural network with probabilistic layers [33] is also employed to predict the error. The architecture of the network is similar to Fig. 1 except that the last hidden layer is a dense variational layer and the output layer is a normal distribution (variational posterior distribution) which allows to model the aleatoric and epistemic uncertainties in the data. The dense variational layer implemented through TensorFlow Probability [33] enables learning a distribution over the weights of a layer through maximization of the Evidence Lower Bound (ELBO) [34], which in turn minimizes the Kullback–Leibler divergence [35] between the true posterior distribution and the variational posterior distribution. The loss function minimized in this setting is the negative log-likelihood loss, which is equivalent to minimizing the mean squared error loss when the data likelihood follows a Gaussian distribution. Practically, this network provides the mean and standard deviation of the outputs, which are approximated by a Gaussian distribution. The prediction uncertainty can be useful when deploying the network, as will be shown in Section 3.1.

The models are implemented using the TensorFlow library [36]. The loss function is the mean squared error between the predicted and labeled $\log(\epsilon)$ or CPU time. The models are trained using the Adam optimizer for up to 200 epochs with a batch size of 512. An initial learning rate of $10^{-3}$ is used and the learning rate is decreased by a factor of 5 when the loss plateaus. The models are trained in less than an hour on a single Nvidia P100 GPU.

### 2.4. Reacting spray simulations

The potential of the present approach for practical CFD is demonstrated by performing simulations of a diesel spray flame. The Spray A conditions from the Engine Combustion Network (ECN) database [37], listed in Table 3, are chosen. Spray A has been the subject of many experimental [38–40] and numerical [41–43] studies where detailed descriptions of the experimental setup can be found.

The simulations are performed using CONVERGE [44] with the renormalized group (RNG) RANS turbulence model [45]. A reduced *n*-dodecane chemical mechanism which contains 65 species and 363 reactions is used. The hybrid mechanism was calibrated following the methodology presented in [46] and is available at http://combustion.llnl.gov/. A well-stirred reactor combustion model is used and a multi-zone model [47] with 5 K temperature bins and 0.05 equivalence ratio bins is used to accelerate the chemistry calculations. The use of a multi-zone model allows for a significant computation cost reduction in the present simulations, decreasing the simulation time from over 10 days to around 13 h on 36 CPUs (with integrator 0). The minimum reaction temperature was set to 600 K. This is not expected to affect the simulation results since, with an ambient temperature of 900 K, only a very small number of cells experience temperature below 600 K (due to vaporization). A base mesh size of 2 mm is used with the mesh size refined to 0.25 mm around the nozzle and adaptive mesh refinement (AMR) with a minimum cell size of 0.25 mm used to automatically refine the mesh in areas where the velocity and temperature sub-grid field values exceed 1% and 2.5% of their domain characteristic values, respectively, leading to a peak cell count of around 240,000.

While the Converge code is used in the present work, the integrator selector introduced here does not depend on a specific CFD code or specific operator-splitting scheme. The method can be integrated in any CFD code which uses operator-splitting. While different CFD codes can produce different results for a variety of reasons (*e.g.* differences in spatial or temporal discretization and integration schemes, difference in turbulence model implementations, etc.), it is not expected that the present methodology would amplify these differences. Additionally, it should be noted that the present approach does not predict "breakdowns" of the operator-splitting method as it assumes that chemistry and transport can be accurately integrated separately and only aims to select the most appropriate chemistry integrator. Users should verify that the operator-splitting method used is stable for the simulations considered.

Furthermore, there is no special coupling or dependence between the multi-zone model and the ODE integrator selector. The neural networks takes a state vector as input and return the appropriate integrator for that state. The prediction is done for each zone when using a multi-zone model or each computational cell without multi-zone. Both the neural network evaluation and the chemistry integration are performed on the CPU. The cost of evaluating the neural network is negligible compared to the cost of the chemistry integration. For the 65 species mechanism used in the spray simulations, the computational cost of the neural network evaluation is less than 1% of the chemistry integration cost. Both the neural network evaluation cost and the chemistry integration cost scale roughly linearly with the number of zones (or grid cells without a multi-zone model) so the cost breakdown is not expected to change if the grid is refined or if no multi-zone model is used.

## 3. Results

In the present section, the ODE integrator selection methodology is first validated, then its generalizability to different fuels and mechanisms is assessed, and, finally, the methodology is used in 3-D CFD simulations. The results in Sections 3.1–3.3.1 are from 0-D well-stirred reactor simulations, performed using the Zero-RK software package [12,28]. These simulations do not include transport and thus no operator-splitting is necessary. In constrast, the simulations in Section 3.3.2 use the Converge CFD code [44] with operator-splitting and a multi-zone model.

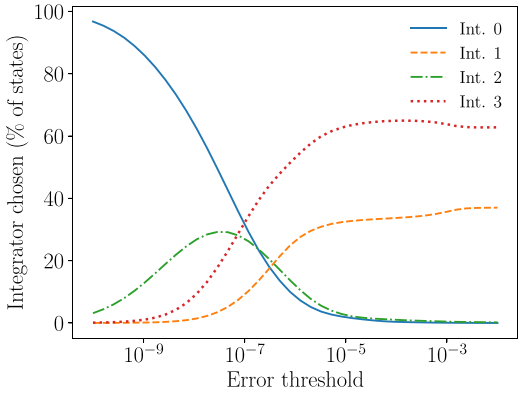



**Fig. 2.** Distribution of the optimal integrators at varying error thresholds.

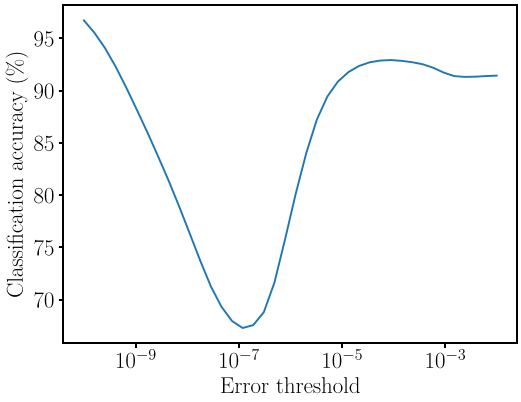



**Fig. 4.** Classification accuracy of the neural network predictor at varying error thresholds.

### 3.1. Model validation

The performance of the ODE integrator "selector" neural networks on the thermochemical states dataset is assessed here. The dataset was randomized and split into training (80%, 1.6M samples) and validation (20%, 0.4M samples) sets.

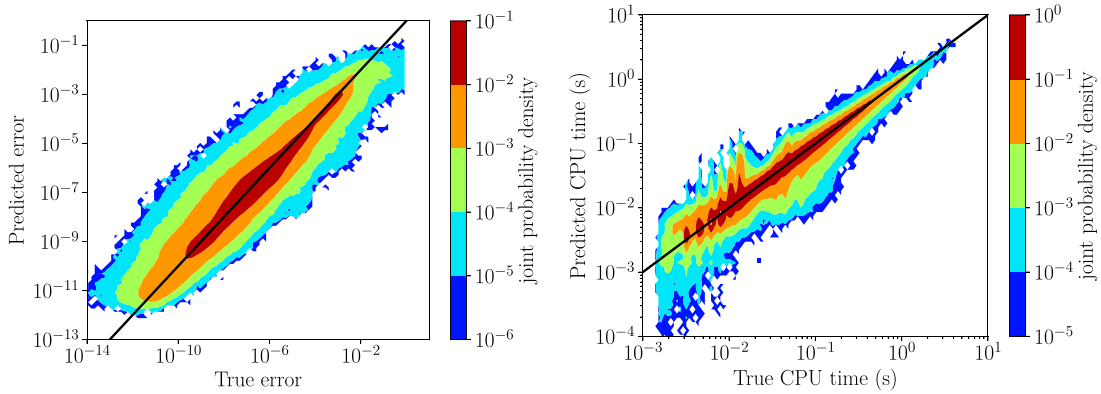
Prior to evaluating the performance of the neural networks, it is insightful to observe the breakdown of optimal integrators at varying error thresholds for all cases in the dataset. Figure 2 shows how often each of the four integrators is the optimal integrator at a given error threshold. As expected, the reference integrator is the optimal integrator in most cases at low error thresholds. As the error threshold is increased, the reference integrator is less likely to be optimal and the integrators with relaxed tolerances are more likely to be optimal. Notably, both CVODE integrators and SEULEX integrators are chosen with high frequency, highlighting that the fastest integration method depends on the thermochemical state and CFD timestep.

The performance of the neural networks is first assessed by comparing the predicted and true values. Figure 3 shows the joint probability density functions of the predicted and true integrator errors and CPU times. The joint probability density function, $\rho_{X,Y}$, is normalized such that the sum over all values of $X$ and $Y$ is one; $\int_x \int_y \rho_{X,Y}(x,y)dydx = 1$. The figure can be interpreted as a scatter plot of the predicted and true values, colored by the local probability density. There is some scatter in the predicted values, but the regions of highest probability are clustered around the diagonal lines indicating perfect predictions. The $R^2$ values for integrator errors and CPU times are 0.77 and 0.97, respectively, illustrating that the error predictor is less accurate than the CPU time predictor. The higher accuracy of the CPU time predictor is likely due to the strong correlation between the CPU time and the $\Delta t_{CFD}$ input.

While quantifying the discrepancies between predicted and true values is insightful, it does not provide direct information on the performance of the ODE integrator selector in simulations. The classification accuracy, the computational speedup, and the occurrence of error underpredictions are more relevant metrics.

The classification accuracy is defined as the percentage of cases where the neural networks model selects the optimal integrator. The accuracy is shown in Fig. 4 at varying error thresholds. The accuracy is close to 100% for very low error thresholds. This is expected from the integrator distribution (Fig. 2) since integrator 0 is almost always the optimal integrator. As the error threshold is increased, the accuracy first decreases down to around 70% and then increases again to above 90% at the highest error thresholds (coinciding with the binary distribution in Fig. 2).

The performance is also assessed by comparing the computational speedup, defined as the ratio between the sum of the CPU times of all validation cases computed with integrator 0 and the sum of the CPU times computed with the selected integrators. The speedup is shown in Fig. 5 for the predicted and optimal integrators at varying error thresholds. The speedup achieved using the predicted integrators is very close to that with the optimal integrators, approaching 3.5 at error thresholds greater than $10^{-5}$. It is interesting to observe this similarity between the "predicted" and "optimal" speedups despite mispredicting the optimal integrator in as many as 30% of the cases (as highlighted by the classification




(a) Integrator error ($R^2$=0.77) (b) Integrator CPU time ($R^2$=0.97)

**Fig. 3.** Joint probability density functions of the predicted and true integrator errors and CPU times.

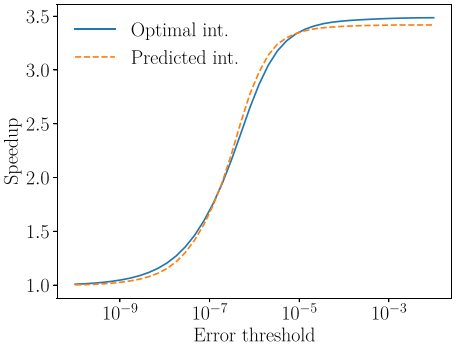



**Fig. 5.** Speedup achieved using the predicted and optimal integrators at varying error thresholds.

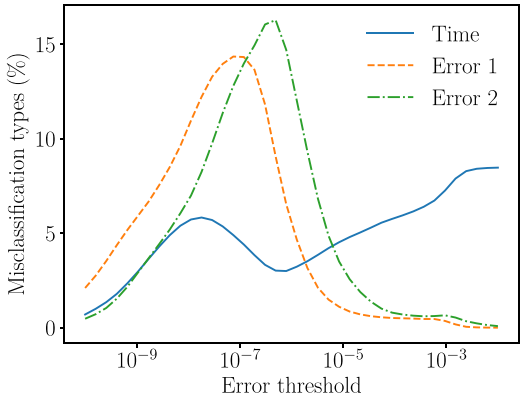



**Fig. 6.** Frequency of the different integrator misclassification types at varying error thresholds.

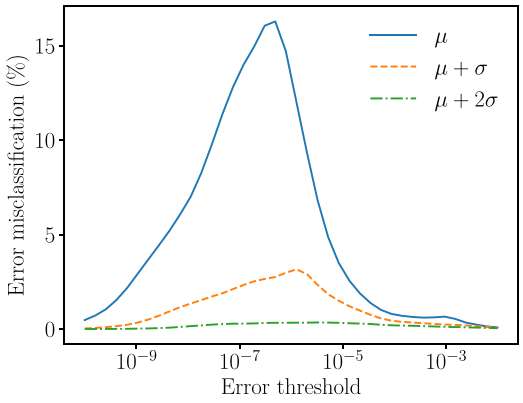



**Fig. 7.** Frequency of error underpredictions (error misclassification type 2) at varying error thresholds with three levels of error uncertainty using probabilistic error prediction.

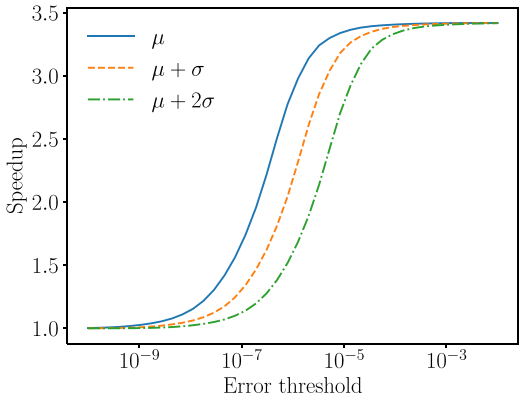



**Fig. 8.** Speedup achieved using the predicted integrators at varying error thresholds with three levels of error uncertainty.

accuracy). This indicates that in most cases where the model selects an integrator which is not optimal, this integrator's CPU time is close to that of the optimal integrator.

Further insight can be gained by characterizing the different types of integrator mispredictions. Given the current integrator selection process (choosing the fastest integrator among those below the error threshold), the mispredictions can be broken down into three distinct cases:

- CPU time misclassification: among the integrators with $\epsilon_i < \epsilon_{\text{TOL}}$, the predicted fastest integrator is not the true fastest integrator.
- Error misclassification type 1: the predicted error of the optimal integrator is above $\epsilon_{\text{TOL}}$ but its true error is below $\epsilon_{\text{TOL}}$.
- Error misclassification type 2: the predicted error of the selected integrator is below $\epsilon_{\text{TOL}}$ but its true error is above $\epsilon_{\text{TOL}}$.

The first two cases lead to the selection of an integrator which is slower than the optimal integrator. In the last case, the selected integrator has an error larger than the specified threshold. This last case is the most undesirable since it can lead to simulation errors exceeding the user tolerance.

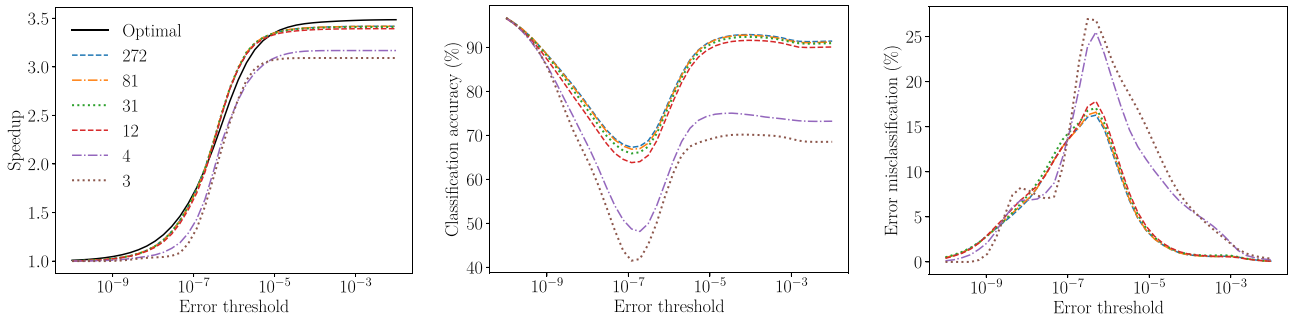
The frequency of the different misclassification types are shown in Fig. 6 at varying error thresholds. The error misclassifications are most common at low error thresholds and decrease as the error threshold is increased. This is expected since integrator CPU time becomes the only decision criterion when all integrators are below the error threshold.

An important observation from Fig. 6 is that error misclassifications of type 2 are fairly common, occurring in up to 15% of cases. A possible avenue to reduce the occurrence of this type of misclassification is to use the variance in the error prediction provided by the probabilistic layer described in Section 2.3. While a (typical) dense output layer provides the predicted error for each integrator, the probabilistic layer provides both the mean predicted error ($\epsilon_i$) and the standard deviation of the predicted error ($\sigma_i$) for each integrator. The standard deviation can be interpreted as a confidence interval for $\epsilon_i$. For example, in our dataset, the true error is within $2\sigma$ of the predicted error for 97% of the samples. The standard deviation can thus be used in the integrator selection to reduce the risk of underpredicting the integrator error by choosing only integrators which satisfy $\epsilon_i + \alpha\sigma_i < \epsilon_{\text{TOL}}$, where $\alpha$ is a user-specified multiplier. As $\alpha$ is increased, the integrator selection becomes more conservative, reducing the likelihood of choosing an integrator whose true error is above the threshold. This is illustrated is Fig. 7 where the frequency of error misclassifications of type 2 is shown for $\alpha = 0$, 1, and 2. The occurrence of error underpredictions is reduced significantly when using the uncertainty in the prediction, occurring in less than 3% of cases with $\alpha = 1$ and less than 0.4% of cases with $\alpha = 2$.

However, the reduction of error underpredictions achieved by using the prediction uncertainty has a negative effect on the simulation time. The more conservative error evaluation leads to selection of slower integrators. This is illustrated in Fig. 8 showing the speedup achieved using the predicted integrators with $\alpha = 0$, 1, and 2. The user thus needs to balance error control and simulation time when choosing values of $\alpha$ and $\epsilon_{\text{TOL}}$.

### 3.2. Generalizability to other mechanisms and fuels

The neural network models presented in previous sections were trained and validated on a dataset generated with a single




(a) Speedup (b) Classification accuracy (c) Error underpredictions

**Fig. 9.** Performance of the ODE solver selector neural networks using different numbers of input features. The "Optimal" lines illustrate the computational speedup achieved if the optimal integrator is used, relative to integrator 0. (For interpretation of the references to color in this figure, the reader is referred to the web version of this article.)

chemical mechanism and using a single fuel composition. It is important to assess the performance of the models for chemical mechanisms or fuels that are not present in the training dataset, as well as the performance when multiple mechanisms or fuels are used to generate the training dataset.

### 3.2.1. Selection of input features

The results presented earlier were obtained using 272 input features; all 269 species mass fractions, temperature, pressure, and the CFD timestep. Using all species mass fractions is an obstacle to generalizability since different chemical mechanisms will contain different chemical species and thus networks trained with data from a given mechanism will be incompatible (or perform poorly) with other mechanisms. Reducing the number of species used as input features can allow to use the same neural network with different mechanisms.

Note that the selection of input features for chemistry integrator selection differs significantly from the selection of input features (progress variables) used in tabulated chemistry approaches since the choice of the appropriate chemistry integrator will likely depend on the stiffness of the ODE system. Common progress variables definitions typically include combustion products (*e.g.* $H_2O$, CO, $CO_2$) to track the progress of the combustion process and such major chemical species are unlikely to be the most useful to predict the stiffness of the ODE system.

The impact of the number of input features on the performance of the ODE solver selector is assessed by considering six different cases using 272, 81, 31, 12, 4, and 3 inputs to the neural networks. The 81 inputs case uses the 81 most important features computed using a tree-based estimator, which include $T$, $p$, and $\Delta t_{\text{CFD}}$. The 31 inputs case uses only $C_0$–$C_3$ species commonly found in hydrocarbon chemical mechanisms along with $T$, $p$, and $\Delta t_{\text{CFD}}$ and is a subset of the 81 inputs case. The 12 inputs case uses the 9 species of the $H_2/O_2$ submechanism (including $N_2$) along with $T$, $p$, and $\Delta t_{\text{CFD}}$. The 4 inputs case uses OH along with $T$, $p$, and $\Delta t_{\text{CFD}}$. Finally, the 3 inputs case uses only $T$, $p$, and $\Delta t_{\text{CFD}}$. A list of the species included for each set of inputs is provided in the supplemental material. The temperature, pressure, and the CFD timestep are present in all six cases since they have a high importance in the tree-based estimator and are available regardless of the chemical mechanism used. The results are shown in Fig. 9 where the speedup, classification accuracy, and occurrence of error underpredictions of the six different cases are compared ($\alpha = 0$ is used in the error prediction). There is very little difference between the 272, 81, 31, and 12 input features cases while a sharp drop in classification accuracy and increase in error underpredictions is observed for the 4 and 3 inputs cases.

Twelve input features can thus be used without any significant performance drawbacks. This is particulary advantageous since the 9 species forming the $H_2/O_2$ submechanism are present in all mechanisms used for simulations of practical transportation fuels.

### 3.2.2. Performance on different mechanisms

The performance of the ODE solver selector neural networks on different chemical mechanisms is assessed by considering two additional mechanisms. The gasoline surrogate mechanisms of Cai and Pitsch [48] (336 species and 1620 reactions, labelled "Cai") and Blanquart et al. [49] (192 species and 1156 reactions, labelled "CM" for CaltechMech) are used. Thermochemical states datasets for these two mechanisms are generated using PaSR and laminar flame calculations as described in Section 2.1 with the same TPRF mixture. The 12 features identified in Section 3.2.1 are used as inputs to the neural networks so that datasets generated with all three mechanisms contain all the input features.

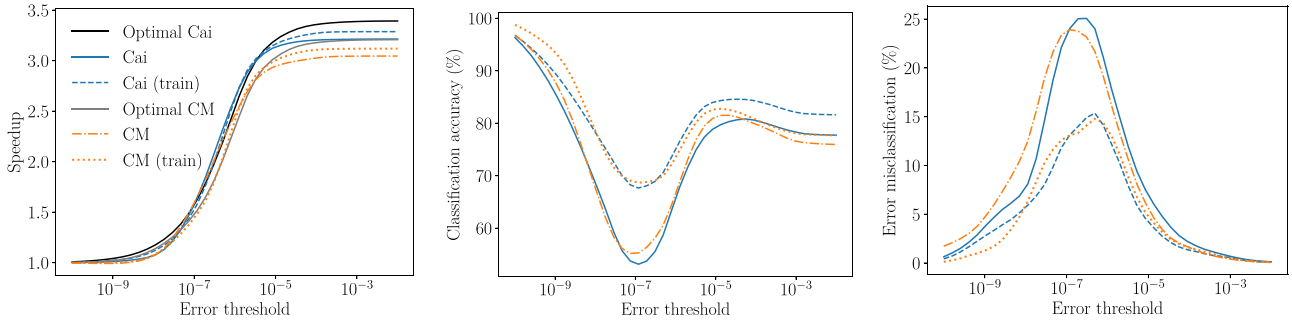
The computational speedup, classification accuracy, and occurrence of error underpredictions for these two mechanisms are first evaluated using the neural networks trained using the 269 species mechanism described in Section 2.1. The two mechanisms tested are thus not used to generate any of the training and validation data. The results are shown in solid and dashdotted lines on Fig. 10. Similar trends are observed for both mechanisms; the speedup is close to the optimal value but the classification accuracy (and error underpredictions) is approximately 10% lower (higher) than for the mechanism that was used to generate the training data.

In a second test, the performance of the "Cai" and "CM" mechanisms is evaluated using neural networks trained on datasets containing thermochemical states computed using two mechanisms; the original 269 species TPRF mechanism and the mechanism tested (either "Cai" or "CM"). The results, labelled with "(train)" are shown in dashed and dotted lines on Fig. 10. A significant improvement in classification accuracy and error underpredictions is observed when the training data contains states generated with the mechanism tested. The performance is similar to that observed when using a single mechanism for training and testing (red line in Fig. 9).

This illustrates that the present ODE solver selector neural network can be used with different chemical mechanisms. A performance reduction is observed if the training data is generated with a single mechanism but it can be avoided by using multiple mechanisms to generate the training dataset.

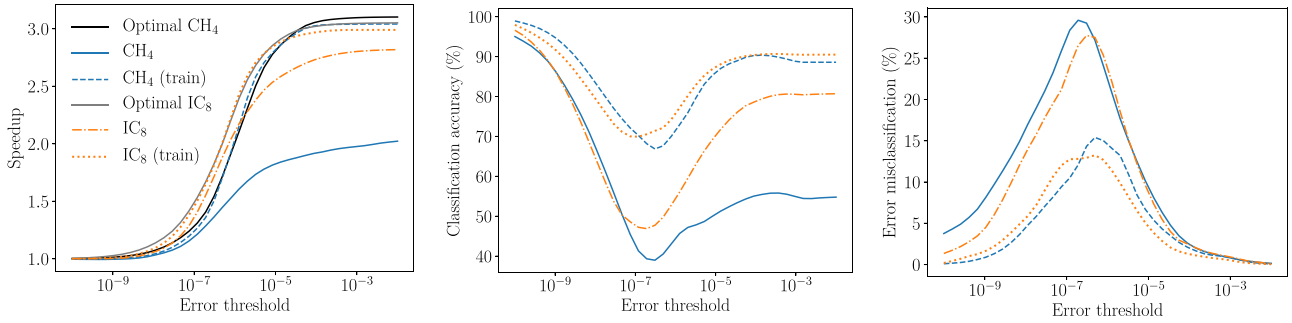
### 3.2.3. Performance on different fuels

The results presented in previous sections used a single fuel composition consisting of a mixture of iso-octane, n-heptane, and toluene. Two additional fuel compositions are now considered to assess the generalizability of the method: pure iso-octane and pure methane. Iso-octane combustion is expected to be relatively sim-




(a) Speedup (b) Classification accuracy (c) Error underpredictions

**Fig. 10.** Performance of the ODE solver selector neural networks for two chemical mechanisms. Solid and dashdotted lines denote cases where the mechanism tested is not used to generate any of the training data. "Train" (dashed and dotted lines) denote cases where the mechanism tested is used to generate some of the training data. The "Optimal" lines illustrate the computational speedup achieved if the optimal integrator is used, relative to integrator 0.




(a) Speedup (b) Classification accuracy (c) Classification accuracy

**Fig. 11.** Performance of the ODE solver selector neural network for two fuels: methane and iso-octane. Solid and dashdotted lines denote cases where the fuel tested is not present in the training data (trained on TPRF fuel data only). "Train" (dashed and dotted lines) denote cases where the fuel tested is present in the training data (trained on TPRF and $CH_4$/$IC_8$ data). The "Optimal" lines illustrate the computational speedup achieved if the optimal integrator is used, relative to integrator 0.

ilar to that of the TPRF mixture used while methane combustion is expected to differ more noticeably. Thermochemical states datasets are generated for these two fuels using the 269 species TPRF mechanism. In addition to the neural networks trained using only TPRF states, two sets of neural networks are trained using TPRF and iso-octane states and TPRF and methane states. The twelve input features selected in Section 3.2.1 are used in all cases.

The performance of the ODE solver selector is first assessed using the neural networks trained only on the TPRF dataset. The fuels tested (pure methane and pure iso-octane) are thus not present in the training data. The computational speedup, classification accuracy, and error underpredictions for both fuels are shown in solid and dashdotted lines on Fig. 11. The speedup for methane is much lower than the optimal value and both fuels exhibit decreased classification accuracies and increased error underpredictions compared to TPRF. Methane results are noticeably worse than iso-octane results, which is expected since its chemistry differs more from that of TPRF.

The performace is significantly improved when neural networks trained on datasets containing the tested fuels are used. This is labelled with "(train)" and shown in dashed and dotted lines on Fig. 11. The speedup is now close to the optimal values for both fuels and the classification accuracy and error underpredictions are similar to those observed when training and testing use a single fuel. This illustrates that the present ODE integrator selector neural network can be used with different fuels. However, all fuels of interest must be present in the training dataset to ensure adequate performance. This suggests that one can build a large dataset containing multiple fuels and train a single neural network to use in all simulations. This is more convenient than training different neural networks for different fuels. However, the performance of neural networks trained using a large number of fuels ( $> 3$) has not been tested here and thus the performance of such a network should be carefully assessed prior to using it in CFD simulations.

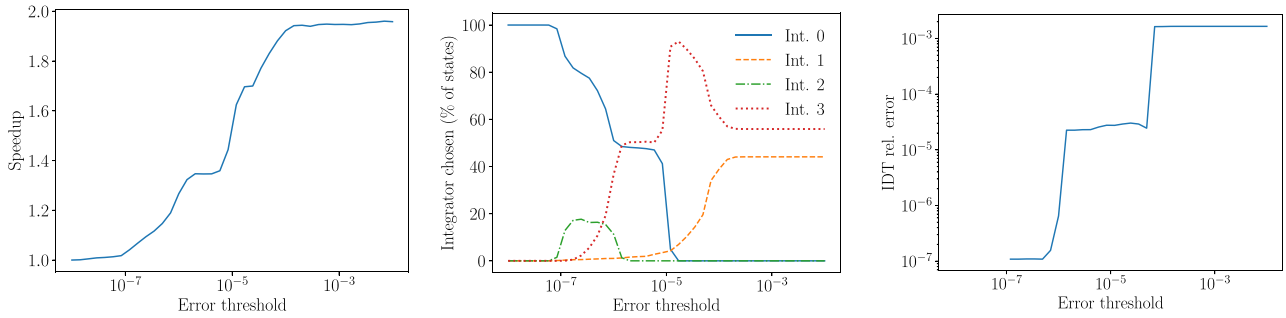
### 3.3. Application to combustion simulations

In this section, the ODE integrator selector neural networks are deployed in combustion simulations. As in previous sections, a dataset of PaSR and flames thermochemical states, generated using the reduced *n*-dodecane mechanism described in Section 2.4, is used to train the neural networks and the same four integrators listed in Table 2 are considered.

#### 3.3.1. 0-D homogeneous reactor

The methodology to select the optimal ODE integrator for each state at each time step is first tested in a 0-D homogeneous reactor. As would be the case in a CFD simulation, the ODE solver is reinitialized at fixed intervals corresponding to the CFD time step. A constant volume adiabatic reactor with a *n*-dodecane/air mixture at 60 bar and 900 K is chosen for its similarity to the diesel spray flame studied in Section 3.3.2. At these conditions, the mixture undergoes a two-state ignition which the first stage occurring after ~ 0.03 ms and the second stage occurring after ~ 0.05 ms. The simulations are conducted for 0.1 ms with a time step of 0.2 µs, similar to the time step used for most of the spray simulation.

Simulations are performed for varying error thresholds from $10^{-8}$ to $10^{-2}$ (with $\alpha = 0$) and the computational speedup (with respect to integrator 0), distribution of selected ODE integrators,




(a) Simulation speedup (b) Integrator distribution (c) IDT relative error

**Fig. 12.** Performance of the ODE solver selector neural networks in homogeneous reactor simulations with $\Delta t_{CFD} = 2 \cdot 10^{-7}$ s. The speedup is defined by comparing the time to solution to that of integrator 0.

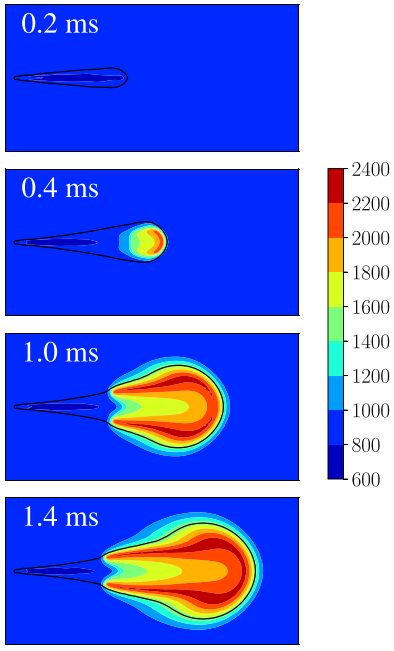



**Fig. 13.** Cross sections of the temperature field along the spray axis at different times. The black line indicates the stoichiometric mixture fraction iso-contour ($Z_{st} = 0.045$).

and relative error on the ignition delay time (IDT) are computed and reported in Fig. 12. The ignition delay time is defined as the time when temperature rises 400 K above the initial temperature.

As was the case with the PaSR/flames training data, the simulation speedup increases as the error threshold is relaxed. The distribution of the selected ODE integrators shows that, as expected, integrators with tight tolerances (0 and 2) are selected for low error thresholds and relaxed tolerance integrators (1 and 3) are preferred for higher error thresholds. Interestingly, integrator 3 is selected with higher frequency for error thresholds around $10^{-5}$ to $10^{-4}$ than for error thresholds $\geq 10^{-3}$, indicating that integrator 1 is faster for these states but integrator 3 has a lower error. In the limit of high error thresholds, integrators 1 and 3 are selected with similar frequency (integrator 1 is preferred up to ignition while integrator 3 is preferred after ignition). Finally, while there is a plateau around $10^{-5}$, the relative error on the ignition delay time exhibits a similar magnitude as the error threshold. This provides confidence that the selector error threshold $\epsilon_{TOL}$ is a good indicator of the overall chemistry integration error.

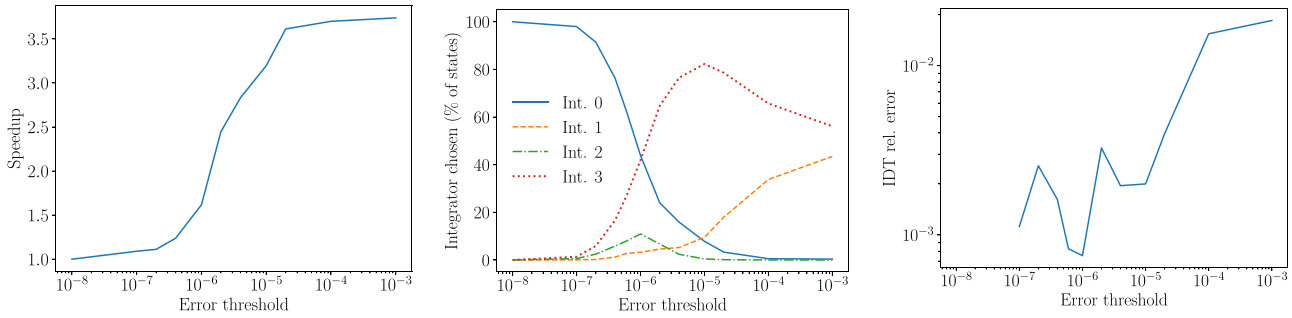
### 3.3.2. 3-D diesel spray

Finally, the integrator selection procedure is used in multi-dimensional CFD simulations of a diesel spray flame. The neural networks are used at each time step in each zone (since a multi-zone model is used) to select the appropriate ODE integrator. The simulation is performed for a total integration time of 2 ms. CONVERGE uses an adaptive time stepping strategy which results in time steps ranging from $10^{-7}$ to $10^{-5}$ s during the course of the present simulations. The time step is approximately $2 \cdot 10^{-7}$ s from around ignition ( ~ 0.4 ms) to the end of injection ( ~ 1.5 ms).

Figure 13 shows the temporal evolution of the temperature field. The black line indicates the iso-contour of stoichiometric mixture fraction ($Z_{st} = 0.045$). The ignition delay time is defined as the time from the start of injection to the time when $dT_{max}/dt$ is maximum (where $T_{max}$ is the maximum temperature in the domain) and is around 0.35 ms in the present simulations. This compares well with the experimentally-measured ignition delay time of approximately 0.4 ms.

Simulations are performed with different error thresholds from $10^{-8}$ to $10^{-3}$. The computational cost reduction (speedup), the distribution of selected ODE integrators, and relative error on the ignition delay time are shown in Fig. 14. The speedup is defined as the simulation time ratio between a reference simulation performed with integrator 0 (CVODE BDF integrator with relative and absolute tolerances of $10^{-8}$ and $10^{-20}$) and the simulation using the ODE solver selector.

Similarly to the PaSR/flames datasets and the homogeneous reactor simulations, the speedup increases monotonically as the error threshold is increased. A speedup of more than 3.5 is achieved for $\epsilon_{TOL} > 2 \cdot 10^{-5}$, noticeably larger than in the 0-D reactor. The distribution of the selected ODE integrators resembles that observed in the 0-D homogeneous reactor with integrator 0 preferred for low error thresholds and integrators 1 and 3 preferred at high error thresholds. The relative error on the ignition delay time increases from $10^{-3}$ at low error thresholds to $10^{-2}$ at the higher thresholds but differs from the roughly monotonic increase from $10^{-7}$ to $10^{-3}$ observed in Fig. 12(c). The effect of the selector error threshold (or, similarly, of the ODE solver tolerances) on the overall simulation error is harder to assess in CFD simulations due to the coupling between reaction and transport and the presence of other sources of error. With that in mind, the user should choose the selector error threshold in a similar fashion as one typically chooses other solver tolerances, compromising between accuracy and computational cost. It is important to keep in mind that the overall accuracy of the present 3-D spray simulations can be affected by many sources of errors, including the chemical mechanism, transport model, multi-zone model, time discretization, spatial discretization, etc. Quantifying the different sources of error in multi-dimensional CFD is beyond the scope of




(a) Simulation speedup (b) Integrator distribution (c) IDT relative error

**Fig. 14.** Performance of the ODE solver selector neural networks in 3-D diesel spray simulations at varying error thresholds. The speedup is defined by comparing the time to solution to that of integrator 0.

the present study. However, the present results suggest that the errors introduced by the ODE integrator selection are small compared to other sources of error.

## 4. Conclusion

A data-driven approach was presented for the selection of ODE solvers in reacting flow simulations using operator-splitting schemes. Neural networks were used as models to predict the CPU time and error of a set of ODE solvers for a given thermochemical state, allowing to choose the optimal ODE solver on a cell-by-cell and timestep-by-timestep basis. The models were trained and validated using a large set of thermochemical states generated through PaSR and laminar flame simulations covering a large array of temperatures, pressures, and mixture compositions. The methodology was first validated by quantifying the prediction errors, the classification accuracy, the types of misclassifications, and the speedup achieved on the validation dataset. The models were shown to predict the optimal ODE solver for 70–95% of the cases in the validation set, depending on the error threshold. The majority of the misclassifications were caused by mispredictions of the integrator error. These error mispredictions were reduced significantly by considering the variance in the predicted error in addition to the mean predicted error. The selection of the optimal ODE integrator for each case led to a speedup of a factor of three over the default ODE integrator.

The generalizability of the present approach to different chemical mechanisms and different fuels was also assessed. It was shown that the performance of the models is only slightly degraded if the inputs to the neural networks are restricted to a small set of thermochemical state variables present in most chemical mechanisms (namely, chemical species present in $H_2/O_2$ reaction mechanisms). This allows to use a single neural network for training or inference with thermochemical states generated using different chemical mechanisms or different fuels. However, a performance reduction (reduced speedup and lower classification accuracy) was observed when the mechanism or fuel used to produce the test dataset were not used to generate any of the training data.

Finally, the models were also employed in a 0-D simulation of an homogeneous reactor and a 3-D CFD simulation of a diesel spray flame. Different ODE integrators were selected for different flow conditions and error thresholds and a speedup of more than 3 was achieved without significant accuracy sacrifices, illustrating that the present methodology is well suited for practical combustion simulations.

In future work, the approach will be extended to include a wider variety of ODE solution methods and solver settings which will likely contribute to further computational cost reductions in multi-dimensional CFD. For example, the prediction and assignment of non-stiff and moderately-stiff chemistry to explicit or linearly-implicit integration techniques could potentially lead to significant cost savings. Another promising avenue for future contributions is online training of the integrator selector neural networks during a CFD simulation. This would reduce the computational burden of the present approach as well as improve its generalizability by avoiding the a priori generation of large PaSR and flames datasets for each fuel and chemical mechanism.

## 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

This work was supported by the U.S. Department of Energy, Office of Energy Efficiency and Renewable Energy, Vehicle Technologies Office and performed under contract DE-AC52-07NA27344.

## Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.combustflame.2020.06.033.

## References

[1] R.D. Reitz, Directions in internal combustion engine research, Combust. Flame 1 (160) (2013) 1–8.
[2] H. Curran, P. Gaffuri, W. Pitz, C. Westbrook, A comprehensive modeling study of n-heptane oxidation, Combust. Flame 114 (1998) 149–177.
[3] H. Curran, P. Gaffuri, W. Pitz, C. Westbrook, A comprehensive modeling study of iso-octane oxidation, Combust. Flame 129 (2002) 253–280.
[4] M. Mehl, W.J. Pitz, C.K. Westbrook, H.J. Curran, Kinetic modeling of gasoline surrogate components and mixtures under engine conditions, Proc. Combust. Inst. 33 (2011) 193–200.
[5] G.I. Marchuk, Some application of splitting-up methods to the solution of mathematical physics problems, Apl. Mat. 13 (2) (1968) 103–132.
[6] G. Strang, On the construction and comparison of difference schemes, SIAM J. Numer. Anal. 5 (3) (1968) 506–517.
[7] O.M. Knio, H.N. Najm, P.S. Wyckoff, A semi-implicit numerical scheme for reacting flow: II. Stiff, operator-split formulation, J. Comput. Phys. 154 (2) (1999) 428–467.
[8] Z. Ren, S.B. Pope, Second-order splitting schemes for a class of reactive systems, J. Comput. Phys. 227 (17) (2008) 8165–8176.
[9] J.F. MacArt, M.E. Mueller, Semi-implicit iterative methods for low mach number turbulent reacting flows: operator splitting versus approximate factorization, J. Comput. Phys. 326 (2016) 569–595.
[10] D.A. Schwer, J.E. Tolsma, W.H. Green Jr, P.I. Barton, On upgrading the numerics in combustion chemistry codes, Combust. Flame 128 (3) (2002) 270–291.
[11] F. Perini, E. Galligani, R.D. Reitz, A study of direct and Krylov iterative sparse solver techniques to approach linear scaling of the integration of chemical kinetics with detailed combustion mechanisms, Combust. Flame 161 (5) (2014) 1180–1195.

[12] M.J. McNenly, R.A. Whitesides, D.L. Flowers, Faster solvers for large kinetic mechanisms using adaptive preconditioners, Proc. Combust. Inst. 35 (2015) 581–587.
[13] A. Imren, D.C. Haworth, On the merits of extrapolation-based stiff ode solvers for combustion CFD, Combust. Flame 174 (2016) 1–15.
[14] N.J. Curtis, K.E. Niemeyer, C.-J. Sung, An investigation of GPU-based stiff chemical kinetics integration methods, Combust. Flame 179 (2017) 312–324.
[15] S.D. Cohen, A.C. Hindmarsh, P.F. Dubois, Cvode, a stiff/nonstiff ode solver in c, Comput. Phys. 10 (2) (1996) 138–143.
[16] F. Perini, E. Galligani, R.D. Reitz, An analytical Jacobian approach to sparse reaction kinetics for computationally efficient combustion modeling with large reaction mechanisms, Energy Fuels 26 (8) (2012) 4804–4822.
[17] P. Deuflhard, Recent progress in extrapolation methods for ordinary differential equations, SIAM Rev. 27 (4) (1985) 505–535.
[18] F. Bisetti, Integration of large chemical kinetic mechanisms via exponential methods with Krylov approximations to jacobian matrix functions, Combust. Theor. Model. 16 (3) (2012) 387–418.
[19] G. Wanner, E. Hairer, Solving Ordinary Differential Equations II, Springer, Berlin Heidelberg, 1996.
[20] Y. Shi, W.H. Green, H.-W. Wong, O.O. Oluwole, Accelerating multi-dimensional combustion simulations using GPU and hybrid explicit/implicit ode integration, Combust. Flame 159 (7) (2012) 2388–2397.
[21] J. Kodavasal, K. Harms, P. Srivastava, S. Som, S. Quan, K. Richards, M. Garcia, Development of a stiffness-based chemistry load balancing scheme, and optimization of input/output and communication, to enable massively parallel high-fidelity internal combustion engine simulations, J. Energy Res. Tech. 138 (5) (2016).
[22] A.T. Alferman, Evaluating Stiffness Metrics for Predicting the Cost of Chemical Kinetics Integration, Oregon State University, 2018.
[23] S. Lam, D. Goussis, Understanding complex chemical kinetics with computational singular perturbation, Symp. (Int.) Combust. 22 (1) (1989) 931–941.
[24] T.F. LU, C.S. YOO, J.H. CHEN, C.K. LAW, Three-dimensional direct numerical simulation of a turbulent lifted hydrogen jet flame in heated coflow: a chemical explosive mode analysis, J. Fluid Mech. 652 (2010) 45–64.
[25] J.-Y. Chen, Stochastic modeling of partially stirred reactors, Combust. Sci. Technol. 122 (1–6) (1997) 63–94.
[26] K.E. Niemeyer, N.J. Curtis, C.-J. Sung, pyjac: Analytical jacobian generator for chemical kinetics, Comput. Phys. Commun. 215 (2017) 188–203.
[27] M. Mehl, K. Zhang, S.W. Wagnon, G. Kukkadapu, C.K. Westbrook, W.J. Pitz, Y. Zhang, H.J. Curran, M.A. Rachidi, N. Atef, S.M. Sarathy, A comprehensive detailed kinetic mechanism for the simulation of transportation fuels, 10th US National Combustion Meeting, 2017. Paper 1A17
[28] M. McNenly, R. Whitesides, Zero-RK: Zero Order Reaction Kinetics. Available at https://github.com/llnl/zero-rk, 2019.
[29] S. Lapointe, R.A. Whitesides, M.J. McNenly, Sparse, iterative simulation methods for one-dimensional laminar flames, Combust. Flame 204 (2019) 23–32.
[30] Y. Gao, Y. Liu, Z. Ren, T. Lu, A dynamic adaptive method for hybrid integration of stiff chemistry, Combust. Flame 162 (2) (2015) 287–295.
[31] Z. Lu, H. Zhou, S. Li, Z. Ren, T. Lu, C.K. Law, Analysis of operator splitting errors for near-limit flame simulations, J. Comput. Phys. 335 (2017) 578–591.
[32] J. Demmel, S. Eisenstat, J. Gilbert, X. Li, J. Liu, A supernodal approach to sparse partial pivoting, SIAM J. Matrix Anal. A 20 (3) (1999) 720–755.
[33] J.V. Dillon, I. Langmore, D. Tran, E. Brevdo, S. Vasudevan, D. Moore, B. Patton, A. Alemi, M. Hoffman, R.A. Saurous, TensorFlow Distributions, arXiv e-prints (2017) arXiv:1711.10604.
[34] D. Blei, A. Kucukelbir, J. McAuliffe, Variational inference: a review for statisticians, J. Am. Stat. Assoc. 112 (2017) 859–877.
[35] S. Kullback, R. Leibler, On information and sufficiency, Ann. Math. Stat. 22 (1951) 79–86.
[36] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al., Tensorflow: a system for large-scale machine learning, 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16) (2016), pp. 265–283.
[37] L. Pickett, G. Bruneaux, R. Payri, Engine Combustion Network, 2020, http://ecn.sandia.gov.
[38] L. Pickett, C. Genzale, G. Bruneaux, L.-M. Malbec, L. Hermant, C. Christiansen, J. Schramm, Comparison of diesel spray combustion in different high-temperature, high-pressure facilities, SAE Int. J. Engines 3 (2010) 156–181.
[39] L. Pickett, J. Manin, C. Genzale, D. Siebers, M. Musculus, C. Idicheria, Relationship between diesel fuel spray vapor penetration/dispersion and local fuel mixture fraction, SAE Int. J. Engines 4 (2011) 764–799.
[40] S. Skeen, J. Manin, L. Pickett, Simultaneous formaldehyde PLIF and high-speed schlieren imaging for ignition visualization in high- pressure spray flames, Proc. Combust. Inst. 35 (2015) 3167–3174.
[41] S. Bhattacharjee, D.C. Haworth, Simulations of transient n-heptane and n-dodecane spray flames under engine-relevant conditions using a transported PDF method, Combust. Flame 160 (10) (2013) 2083–2102.
[42] Y. Pei, E.R. Hawkes, S. Kook, G.M. Goldin, T. Lu, Modelling n-dodecane spray and combustion with the transported probability density function method, Combust. Flame 162 (5) (2015) 2006–2019.
[43] Y. Pei, S. Som, E. Pomraning, P.K. Senecal, S.A. Skeen, J. Manin, L.M. Pickett, Large eddy simulation of a reacting spray flame with multiple realizations under compression ignition engine conditions, Combust. Flame 162 (12) (2015) 4442–4455.
[44] K. Richards, P. Senecal, E. Pomraning, CONVERGE (v2.4), convergent science, madison, wi, 2017,
[45] H. Zhiyu, R. Reitz, Turbulence modeling of internal combustion engines using RNG $k$-$\epsilon$ models, Combust. Sci. Technol. 106 (1995) 267–295.
[46] S. Lapointe, K. Zhang, M.J. McNenly, Reduced chemical model for low and high-temperature oxidation of fuel blends relevant to internal combustion engines, Proc. Combust. Inst. 37 (2019) 789–796.
[47] A. Babajimopoulos, D. Assanis, D. Flowers, S. Aceves, R. Hessel, A fully coupled computational fluid dynamics and multi-zone model with detailed chemical kinetics for the simulation of premixed charge compression ignition engines, Int. J. Engine Res. 6 (2005) 497–512.
[48] L. Cai, H. Pitsch, Optimized chemical mechanism for combustion of gasoline surrogate fuels, Combust. Flame 162 (2015) 1623–1637.
[49] G. Blanquart, P. Pepiot-Desjardins, H. Pitsch, Chemical mechanism for high temperature combustion of engine relevant fuels with emphasis on soot precursors, Combust. Flame 156 (3) (2009) 588–607.