Data-driven framework for input/output lookup tables reduction – with application to hypersonic flows in chemical non-equilibrium

Clément Scherding, Georgios Rigas, Denis Sipp, Peter J. Schmid, Taraneh Sayadi

Data-driven framework for input/output lookup tables reduction - with application to hypersonic flows in chemical non-equilibrium

Clément Scherding,[1,*] Georgios Rigas,[2] Denis Sipp,[3] Peter J. Schmid,[4] and Taraneh Sayadi[1,5]

[1] *Institut Jean le Rond d'Alembert, Sorbonne University, France*
[2] *Department of Aeronautics, Imperial College London, UK*
[3] *DAAA, Onera, France*
[4] *Department of Mechanical Engineering, KAUST, SA*
[5] *Institute for Combustion Technology, Aachen University, Germany*

Hypersonic flows are of great interest in a wide range of aerospace applications and are a critical component of many technological advances. Accurate simulations of these flows in thermodynamic (non)-equilibrium (accounting for high temperature effects) rely on detailed thermochemical gas models. While accurately capturing the underlying aerothermochemistry, these models dramatically increase the cost of such calculations. In this paper, we present a novel model-agnostic machine-learning technique to extract a reduced thermochemical model of a gas mixture from a library. A first simulation gathers all relevant thermodynamic states and the corresponding gas properties via a given model. The states are embedded in a low-dimensional space and clustered to identify regions with different levels of thermochemical (non)-equilibrium. Then, a surrogate surface from the reduced cluster-space to the output space is generated using radial-basis-function networks. The method is validated and benchmarked on a simulation of a hypersonic flat-plate boundary layer with finite-rate chemistry. The gas properties of the reactive air mixture are initially modeled using the open-source Mutation++ library [1]. Substituting Mutation++ with the light-weight, machine-learned alternative improves the performance of the solver by 50% while maintaining overall accuracy.

I. INTRODUCTION

Non-equilibrium effects have been shown to play an important role in the accurate simulation of flows at hypersonic conditions and in the computation of design characteristics, such as transition location or thermal loading [2–5]. Recent studies have identified these effects as causes of order-one changes in growth rates, response behavior or sensitivities, even though the corresponding variations in first-order flow statistics have been modest. These findings have in turn prompted a significant endeavor of augmenting existing flow solvers with non-equilibrium modules to account for finite-rate aerothermochemical features.

Simulations in this parameter regime introduce and track a range of species in their inert or ionized forms [6–8]. Complementing the hydrodynamic state vector by chemical components is a well-established technique, for example in combustion or atmospheric simulations, but the required modeling of the inter-species interactions, such as dissociation, reaction and recombination [9], for hypersonic applications poses great challenges.

Much of this modeling is accomplished by lookup libraries, which act as repositories of tabulated chemical reactions encountered for a given flow state [1]. When passing state-vector components to the library, amplitudes and time-scales for various forcing terms are returned, appearing as exogeneous inputs to the momentum, energy and species transport equations.

Much effort has gone into these libraries, and for aerothermochemical non-equilibrium effects in hypersonic flows, the Mutation++ library (MUlticomponent Thermodynamic And Transport properties for IONized gases in C++), developed and maintained at the von Karman Institute (VKI), has become the standard for high-fidelity simulations of high-speed and high-enthalpy flows [1]. This library can be coupled to existing flow solvers and is capable of modeling a range of partially ionized gas effects, together with non-equilibrium features, energy exchange processes and gas-surface interactions. The flexibility and scope of the library comes at the expense of a computational bottleneck that slows down a typical large-scale simulation by a large factor, as shown in Fig. 1, where typical simulation times for calorically and thermally perfect gases are juxtaposed with results for non-equilibrium chemical reactions. A wide margin can be observed. For this reason, non-equilibrium computations range among the most inefficient and laborious calculations in fundamental hypersonic research.

* Corresponding author: clement.scherding@dalembert.upmc.fr

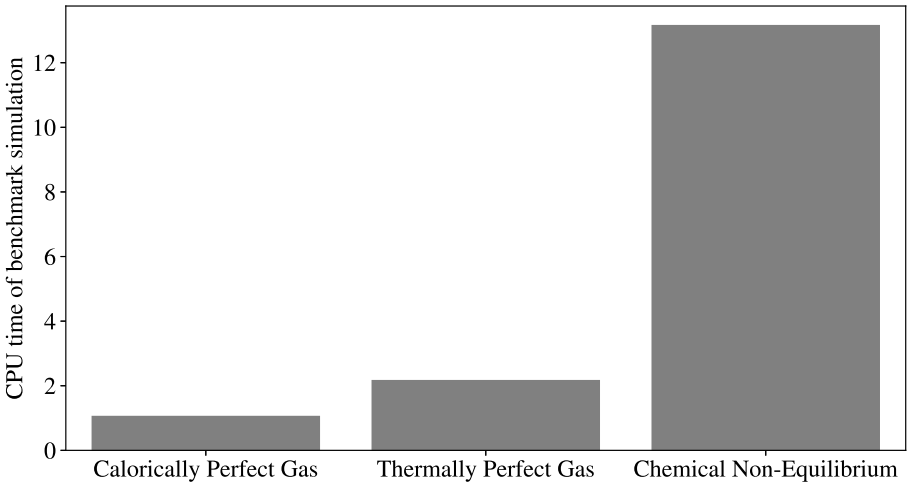

FIG. 1. CPU time of a benchmark simulation, run with different aerothermochemical models. Including non-equilibrium effects in the simulation causes a significant increase in computational time.

More generally, many engineering applications need to evaluate an expensive function $\tilde{f}(\mathbf{x})$ many times. Therefore, it is of great interest to alleviate the CPU burden of these applications by finding an efficient approximation of such functional forms. One of the oldest and most common approach to approximate $\tilde{f}$ is to use structured tabulation. In a pre-processing step, values of $\tilde{f}$ are tabulated for a hypercube in the input space. Then, during the simulation, values of $\tilde{f}$ are linearly interpolated in the table. The Look-up Table method (LuT) was proven successful in many applications, such as tabulated chemistry for spray combustion [10], design of energy devices using organic Rankine Cycles [11] or simulations of hypersonic boundary layers in chemical equilibrium [12]. However, building and storing the table, together with the look-up procedure during the simulation, become computationally more intensive as the number of dimensions D of the input space increases. This demonstrates the well-known curse of dimensionality, where the volume of sample points needed to construct an accurate table increases exponentially with the number of dimensions of the input space. Similarly, linear interpolation in high dimensions is a tedious task. This latter point even prevents the application of this LuT-methodology in the case considered in this study where the input space dimension is $D = 6$. Pope developed the ISAT algorithm (*in situ adaptive tabulation*) to overcome this deficiency in high dimensions with a storage/retrieval approach and demonstrate the concept on applications in the combustion field [13].

Recently, more general methods that can tackle higher dimensional problems have also been proposed and saw considerable success in a variety of applications, particularly in the active research field known as surrogate modeling. Underlying this effort is the universal approximation theorem [14] which proves that deep neural networks, with at least one hidden layer and non-linear activation functions, formally proposed by LeCun [15], can approximate any non-linear function of any dimension. For example, Liu *et al.* [16] used neural-networks for surrogate-model-based optimization in aeronautics. However, the training cost of the network by back propagation becomes prohibitive as the number of neurons and layers increase – a necessity which might arise in complex high-dimensional problems. Radial basis function (RBF) networks, a special case of three layers neural networks [17, 18], can also be used for nonlinear function approximation in any dimension. Their training is easier and cheaper than classical neural networks as the optimal weights can be found by solving a linear system of equations. RBFs have been widely used for surrogate modeling in many fields such as aerodynamic shape optimization [19, 20] and meteorology [21], to name but two. Statistical surrogate modeling techniques have also found great success as they directly include an estimation of the error in the model. The method of kriging, originally developed for two-dimensional geostatistics problems [22], has been extended to approximate input/ouput problems of any dimension by Sacks *et al.* [23]; see the review by Kleijnen on the use of kriging for surrogate modeling [24]. Finally, Polynomial Chaos Expansion (PCE) is another technique that can generate surrogate models well suited for uncertainty quantification [25].

Despite some success, the often brute-force nature of these algorithms may not always yield a satisfactory surrogate model in terms of accuracy and computational cost. Bouhlel *et al.* [26] pointed out several performance issues when performing kriging in high dimensions ($D = 100$). This number of dimensions is common in reactive flow simulations where hundreds of species are tracked, even with reduced chemical mechanisms [27–29]. Moreover, one common assumption in surrogate modeling relates to the smoothness of the approximated relation. This is not always true,

especially in hypersonic applications where shocks and temperature discontinuities are amongst the typical features of such flows. Nonetheless, clever pre-processing steps can greatly improve the model's performance in these cases. For example, Bouhlel et al. [26] coupled kriging with partial least-squares (PLS) methods to reduce the high-dimensional ($D = 100$) input space. In [30], principal component analysis (PCA) has been used as a pre-processing step before applying polynomial chaos expansions on the PCA basis [30]. When dealing with discontinuous functions, Bettebghor *et al.* [31] proposed to cluster the input basis into different regions (to avoid a discontinuity within a cluster) and build a surrogate model on each of these regions. All models are then combined together and form a mixture of experts (MoE), as described in the literature [32]. Yang [33], however, pointed out that combining surrogate surfaces does not necessarily outperform a single model fitted over the entire input space. Hence, special care has to be taken in combining these steps.

The objective of this work is therefore to develop an effective pre-processing technique, allowing the construction of a low-dimensional surrogate model capable of replacing the computationally expensive library and the memory-intensive look-up tables when modeling inter-species interactions in simulations of chemically reactive flows.

The paper is organized as follows. In Section II, the generic algorithm is presented in detail. It combines techniques from nonlinear model reduction, network clustering and surrogate modeling to efficiently extract a surrogate, light-weight model of the full library. In Section III, the governing equations for hypersonic flows in chemical non-equilibrium as well as the thermo-chemical model for such flows are recalled. Finally, in Section IV, the algorithm is tested on the simulation of an adiabatic Mach-10 boundary layer in chemical non-equilibrium, initially studied by Marxen *et al* [2]. The gas properties of the reactive air mixture are initially modeled using the open-source Mutation++ library [1]. Replacing the library by the surrogate model then overcomes the computational bottleneck alluded to above.

While focusing on the non-equilibrium gas-dynamic library Mutation++, we stress that the employed techniques are agnostic about the particular library they are applied to, and can just as readily be employed to other libraries or lookup tables attached to simulations. Applications in combustion, phase-change simulations or particle-laden flows stand to benefit from this accelerating methodology at the interface between flow solvers and material-property libraries.

II. DESCRIPTION OF THE ALGORITHM

In this section, the general algorithm to extract a reduced library for the thermochemical properties of multi-component mixtures in chemical non-equilibrium is described.

Compressible flow simulations in chemical non-equilibrium require transport, thermodynamic and chemical reactions' properties, $\tilde{\mathbf{z}} \in \mathbb{R}^{D_Z}$, (*e.g.* viscosity, conductivity, enthalpies, and chemical source terms) to close the governing equations. These properties are modeled as functions of the local thermodynamic state and mixture composition, concatenated into the local thermodynamic vector $\tilde{\mathbf{q}}_{\text{th}} \in \mathbb{R}^D$, usually computed using tabulation or external libraries, and can be considered as an input/output problem:

$$\tilde{\mathbf{z}} = \tilde{f}(\tilde{\mathbf{q}}_{\text{th}}), \tag{1}$$

where the function $\tilde{f}$ represents the library of interest, and D and D_Z represent the dimensions of the input and output spaces, respectively. This function then needs to be evaluated at each grid point and at each time-step. While accurate, the extensive calls to the library come at a substantial performance loss for the solver, together with a significant time penalty (see Fig. 1).

While these function calls cannot be entirely avoided, existing features of the flow inspire strategies to seek a less expensive method to evaluate the required properties. (i) Flows have history. In other words, several calls to the library may be redundant since some thermodynamic states are seen multiple times throughout the simulation. (ii) Any flow of interest contains only a subset of all possible thermodynamic states, given its nature and freestream conditions. Hence, only a small subset of the input space of function $\tilde{f}$ needs to be accessed. (iii) While data-driven method requires a lot of data for training, some (rare) flows of interest, such as hypersonic boundary layers in chemical non-equilibrium, exhibit elegant, locally self-similar solutions [34] that can be used for training instead of an expensive direct numerical simulation (DNS). This final point will be explored in more detail in future work.

The proposed algorithm leverages these features by creating a surrogate model of the function $\tilde{f}$ only on a subset of input states relevant to the simulation, which is commonly represented as a low-dimensional manifold in $\mathbb{R}^D$. This allows us to first perform dimensionality reduction of the input space (see [26, 30]). Next, following a similar approach as in [31], regions with different dynamics and/or discontinuities between them are clustered into a low-dimensional representation. Finally, surrogate models are constructed on each cluster in this low-dimensional space. Hence, the

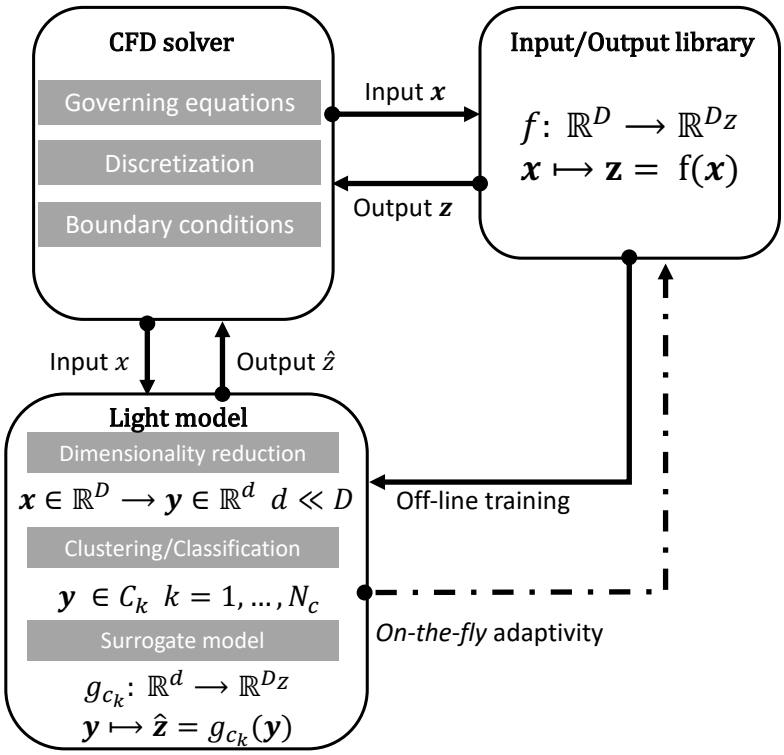

FIG. 2. General schematic of the model training and coupling to replace any expensive Input/Output library.

training of the algorithm is performed in three steps: (i) dimensionality reduction, (ii) community clustering, and (iii) surrogate model construction. Once trained, the model replaces the look-up library already in place to predict the thermochemical properties of the mixture within the flow solver. We stress that the lighter version of the library will perform correctly only on the range of conditions seen during the simulation. A general schematic of the training process and the coupling with the flow solver is presented in Fig. 2.

In the future, on-the-fly adaptivity will be added to the algorithm to allow the model to learn new states as the simulation is advanced in time. This capability will help tackle more challenging and unsteady flow problems, while alleviating the need for a complete training set which may not always be available.

A. Training

The algorithm is trained using the simulation of an adiabatic Mach-10 boundary layer in chemical non-equilibrium, thoroughly described in Section III D. The gas considered is a five-species air model $\mathcal{S} = N_2, O_2, NO, N, O$ with five reactions. The computational code solves the compressible reactive Navier-Stokes equations, where the kinetic parameters are computed by coupling with Mutation++ [1]. The input thermodynamic state vector $\tilde{\mathbf{q}}_{\mathrm{th}}$ is composed of density ρ, internal energy ρe and the mixture partial densities $\rho_s, s \in \mathcal{S}$. The outputs of the library fall within three categories: thermodynamic properties (pressure p, temperature T, species specific enthalpies $h_s, s \in \mathcal{S}$), transport properties (viscosity μ, thermal conductivity κ, diffusion coefficients $D_s, s \in \mathcal{S}$) and chemical kinetics source term $\dot{\omega}_s, s \in \mathcal{S}$. More details concerning the governing equations and the thermo-chemical model can be found in Section III.

1. *Data collection*

To train the model, N thermodynamic state vectors $\tilde{\mathbf{q}}_{\mathrm{th}}$ are randomly sampled on the grid of a previously converged simulation in chemical non-equilibrium and concatenated into the input vector $\tilde{\mathbf{X}} \in \mathbb{R}^{N \times D}$. The corresponding outputs from the library are collected and concatenated into the output vector $\tilde{\mathbf{Z}} \in \mathbb{R}^{N \times Dz}$. Fig. 3 shows the numerical range

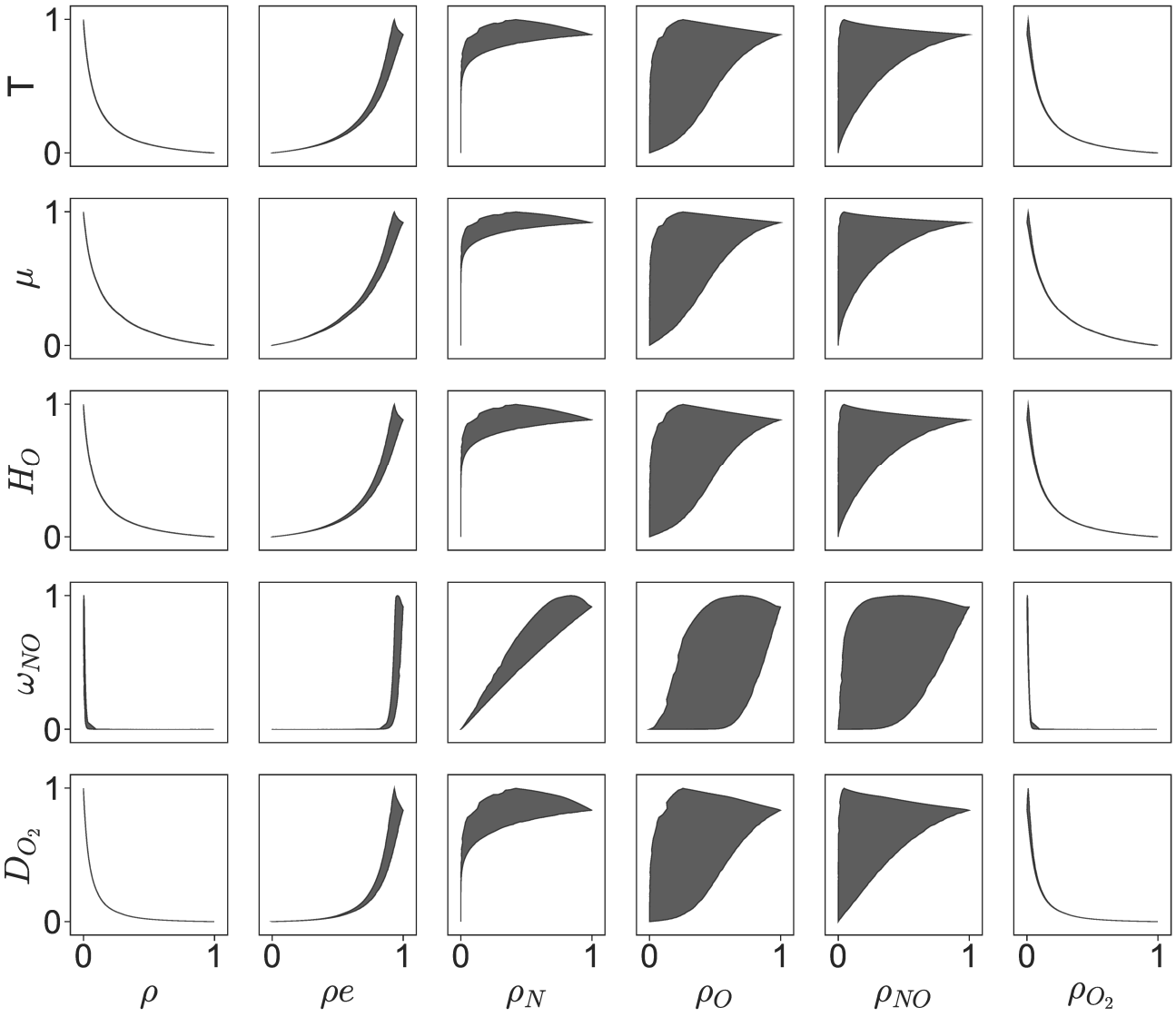

FIG. 3. Numerical range of selected Mutation++ outputs $\mathbf{Z}$ (vertical) with respect to the inputs $\mathbf{X}$ (horizontal).

of selected output variables along each input, normalized between 0 and 1 with a minimum-maximum scaling

$$\mathbf{X} = \frac{\tilde{\mathbf{X}} - \tilde{\mathbf{X}}_{min}}{\tilde{\mathbf{X}}_{max} - \tilde{\mathbf{X}}_{min}}, \qquad \mathbf{Z} = \frac{\tilde{\mathbf{Z}} - \tilde{\mathbf{Z}}_{min}}{\tilde{\mathbf{Z}}_{max} - \tilde{\mathbf{Z}}_{min}}. \tag{2}$$

Taking dynamic viscosity μ, for example, it shows a strong dependency on density ρ and internal energy ρe but a low variation with respect to the radicals' partial densities ρ_O, ρ_N and ρ_{NO}. The same observations can be made for all other outputs. Hence, the variation of the function $\tilde{f}$ with respect to the inputs can be accurately represented on a low-dimensional subspace of the inputs. This motivates the first step of the algorithm, namely, dimensionality reduction.

2. Dimensionality reduction

The goal of this section is to find an effective algorithm for dimensionality reduction of the input space in order to construct a mapping between its reduced-order representation and the output of the library

$$\hat{\mathbf{Z}} = g(\mathbf{Y}), \tag{3}$$

where g is the approximation of the scaled library f in the low-dimensional subspace of the inputs, $\mathbf{Y}$ is the reduced-order representation of an input $\mathbf{X}$ and $\hat{\mathbf{Z}}$ the prediction of the model. The benefit of this first pre-processing step is to maintain high accuracy of the surrogate model, while decreasing the overall cost of construction and evaluation. In fact, constructing a response surface faces the well-known curse of dimensionality; as the number of input dimensions increases, the cost of constructing an accurate surface increases exponentially. This approach was proven successful in [26] where PLS was used in tandem with kriging to reduce the dimension of the input space.

a. Principal component analysis The most common technique for dimensionality reduction of a dataset $\mathbf{X} \in \mathbb{R}^{N\times D}$ in high dimensions is principal component analysis PCA (see eg. [35]). The principal components of $\mathbf{X}$ are found through the eigenvalue decomposition of the covariance matrix of the data. The dataset $\mathbf{X}$ is then projected onto $d < D$ leading eigenvectors (or principal components) of the covariance matrix, resulting in a low-dimensional representation $\mathbf{Y} \in \mathbb{R}^{N\times d}$ of the original dataset. However, depending on the shape of the manifold, the variations of the output variables with respect to the low-dimensional sub-space may not be properly preserved, which is the case presented in Fig. 4a with points of high and low temperature projected onto similar locations. This example illustrates the limitations of PCA for dimensionality reduction of a dataset constrained to a nonlinear manifold.

b. Auto-encoders Nonlinear dimensionality reduction via auto-encoders (AE) typically have a higher compression rate than linear techniques. An auto-encoder is a parametric model (i.e. a deep neural network with an activation function f_{ac}) that embeds the input dataset $\mathbf{X} \in \mathbb{R}^{N\times D}$ into a low-dimensional representation $\mathbf{Y} \in \mathbb{R}^{N\times d}$ through an encoder function $\mathbf{E}$. The low-dimensional representation is then decoded back to the input space with the decoder function $\mathbf{D}$, producing a reconstruction of the input $\widehat{\mathbf{X}} \in \mathbb{R}^{N\times D}$.

$$\begin{aligned} \mathbf{Y} &= \mathbf{E}(\mathbf{X}) \\ \hat{\mathbf{X}} &= \mathbf{D}(\mathbf{Y}) \end{aligned} \tag{4}$$

The weights of the two networks $\mathbf{E}$ and $\mathbf{D}$ can be trained using back-propagation of the L_2 error $\|\mathbf{X} - \widehat{\mathbf{X}}\|_2$ through the network. If the activation function is selected as the identity (i.e $f_{ac}(\mathbf{x}) = \mathbf{x}$), the auto-encoder is linear and unbiased

$$\begin{aligned} \mathbf{E} &= W_E \\ \mathbf{D} &= W_D \end{aligned} \tag{5}$$

where W_E and W_D are the weights matrices of the encoder and decoder, respectively. These optimal weights can be found through PCA. In fact, the linear latent space of dimension d of the encoder will span the same sub-space as the top d PCA singular vectors. The equivalence between the two techniques was first shown by Baldi & Hornik [36]. Correspondingly, a two-layered nonlinear auto-encoder can be mathematically described as follows

$$\begin{aligned} \mathbf{Y} &= W_{E,2} f_{ac}(W_{E,1}\mathbf{X} + \mathbf{b_{E,1}}) + \mathbf{b_{E,2}} \\ \hat{\mathbf{X}} &= W_{D,2} f_{ac}(W_{D,1}\mathbf{Y} + \mathbf{b_{D,1}}) + \mathbf{b_{D,2}} \end{aligned} \tag{6}$$

where f_{ac} is a nonlinear activation function, $W_{E,1} \in \mathbb{R}^{H\times D}, W_{E,2} \in \mathbb{R}^{d\times H}$ are the weights matrices of the first and second layer of the encoder with respective biases $\mathbf{b_{E,1}} \in \mathbb{R}^H$ and $\mathbf{b_{E,2}} \in \mathbb{R}^d$. H denotes the dimension of the hidden layer. The matrices and bias vectors of the decoder have transposed dimensions. This corresponds to the minimal architecture (i.e. with one hidden nonlinear layer and an output linear layer) requested by the universal approximation theorem [14]. However, more hidden layers can be considered. Fig. 4b shows the manifold unrolled in two dimensions with an auto-encoder, colored by the magnitude of the temperature, an output of the library. The AE outperforms PCA by preserving the local structure and preventing points at different thermodynamic states (i.e different temperatures) to be projected onto the same location. However, the highest temperature zone is concentrated in a thin layer adjacent to the lower temperature area. This will result in strong and unphysical gradients of the surrogate model within this region.

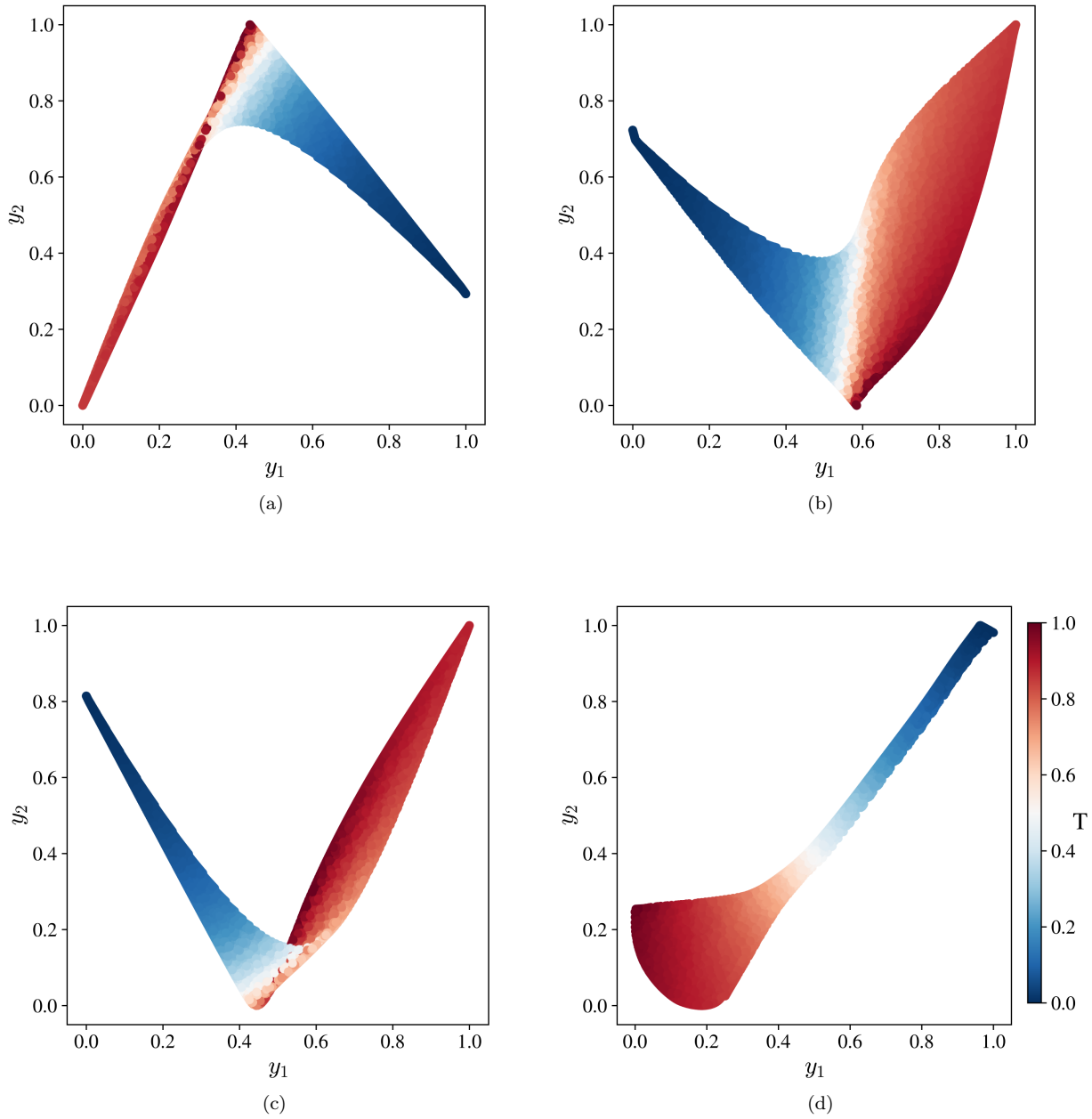

FIG. 4. Low-dimensional representation $Y \in \mathbb{R}^{N\times d}$ ($d = 2$) of $X \in \mathbb{R}^{N\times D}$, colored by temperature T. Obtained with (a) PCA (b) AE (c) PLS (d) IO-E.

c. Partial least-squares Since our interest lies in reducing the dimensionality of the input to construct a reduced-order surrogate model of the input/output relations, it is useful to entangle the input into a low-dimensional space that best reconstructs the outputs. In analogy to PCA finding dependencies between the inputs, partial least-squares (PLS) finds a basis of the input space that optimally accounts for features in the output space. It has been used to construct surrogate models aimed at reducing the dimensions of the input space (see [26]). Different variants of PLS now exist, using either a singular value decomposition (PLS-SVD) or iterative algorithms (such as PLS-W2A in [37]). While it has been shown that in cases where the dimension of the latent space is strictly greater than one, PLS-SVD differs from PLS-W2A and its variant PLS2, no major differences in the resulting latent space were observed. The results of PLS-SVD are presented here to highlight the similarity to PCA. Given the input $\mathbf{X}$ and output vectors $\mathbf{Z}$,

the PLS-SVD algorithm [37] determines

$$\mathbf{X}^T\mathbf{Z} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T. \tag{7}$$

Similar to PCA, the projection of the input is then obtained by projecting onto the $d < D$ top left singular vectors

$$\mathbf{Y} = \mathbf{X}\overline{\mathbf{U}} \tag{8}$$

where $\overline{\mathbf{U}} \in \mathbb{R}^{D\times d}$ is the truncated left-singular matrix. Fig. 4c shows the training set projected onto the two-dimensional basis generated with PLS. As expected, adding the output information in the computation improves the output's representation in the reduced-order input basis compared to PCA. However, an artificial discontinuity is created near the high-gradient region that was not present in Fig. 4b. This highlights the lower compression rate of linear techniques compared to nonlinear ones.

d. Input/output-encoders The strategy adopted here is therefore a nonlinear dimensionality reduction of the input data using input/ouput-encoders (IO-E), a modified version of auto-encoders (AE) adapted to input/ouput relations. To that end, the decoder architecture is modified to best reconstruct the output of the library $\mathbf{Z} \in \mathbb{R}^{N\times D_Z}$ as

$$\widehat{\mathbf{Z}} = W_{D,2} f_{ac}(W_{D,1}\mathbf{Y} + \mathbf{b_{D,1}}) + \mathbf{b_{D,2}} \tag{9}$$

where the weight matrices are now $W_{D,1} \in \mathbb{R}^{H_D\times d}, W_{D,2} \in \mathbb{R}^{D_Z\times H_D}$ with respective biases $\mathbf{b_{D,1}} \in \mathbb{R}^{H_D}$ and $\mathbf{b_{D,2}} \in \mathbb{R}^{D_Z}$. H_D now represents the hidden dimension of the decoder. The resulting network is then trained via back-propagation of the reconstruction error based on the output $\|\mathbf{Z} - \widehat{\mathbf{Z}}\|_2$. The input/output-encoder (IO-E) architecture is illustrated in Fig. 5.

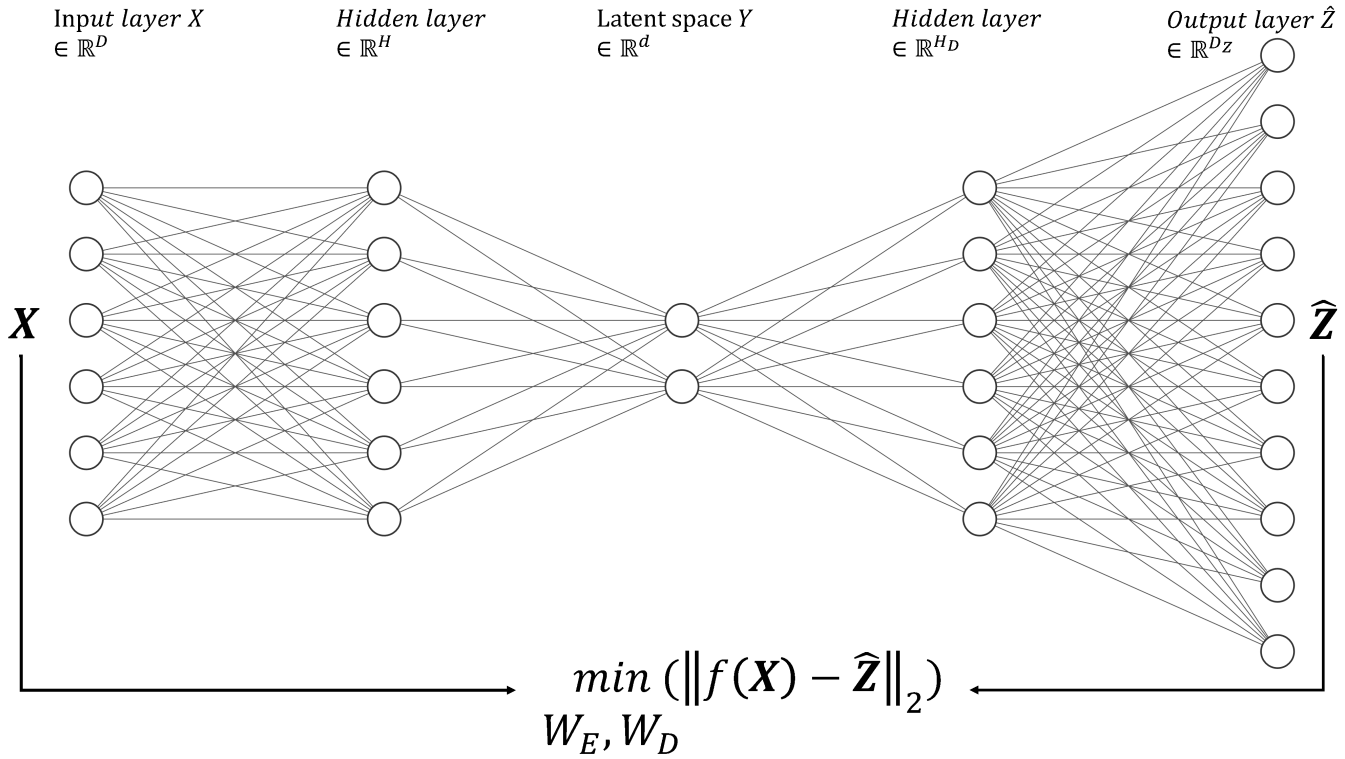

FIG. 5. Schematic of the input/output-encoder architecture proposed. An input sample $\mathbf{X}$ is passed through the encoder network to generate a low-dimensional representation $\mathbf{Y}$. $\mathbf{Y}$ is decoded back to predict the library output $\widehat{\mathbf{Z}}$. At each training step, the L_2 loss $\|\mathbf{Z} - \widehat{\mathbf{Z}}\|_2$ between the library and the network prediction is calculcated and backpropagated until convergence of the encoder and decoder weights, W_E and W_D, respectively, is reached.

Fig. 4d shows the equivalent manifold using the IO-E architecture, with the same number of hidden dimensions. The IO-E architecture outperforms all other techniques as the high-temperature region is now projected onto its own

properly defined zone with smooth variations. This feature will have a marked impact on the performance of the library when linked to the solver. It is worth noting that the decoder part of the IO-E could be used directly to predict the output of the library. However, we will see in Section IV B that the accuracy of the full IO-E would not optimal in this case. This motivates the use of clustering and radial basis functions, as described next.

3. *Community clustering*

In the second step of the algorithm, we seek to discover clusters within our data. In our present context, a cluster represents a subset of data that shares similar thermodynamic features. These feature classification will then be useful in constructing a dedicated surrogate surface of the low-dimensional input manifold. To this end, Newman's spectral algorithm for community detection in a network [38] is used. A clear advantage of Newman's algorithm is that the number of clusters is not defined *a priori*, in contrast to more common clustering techniques (e.g. k-means). The number of clusters is instead the result of a maximization procedure performed on the modularity Q of the network. In other words, the number of thermodynamic clusters in the flow are determined only from the data and is not based on *a priori* knowledge of the user. This knowledge might be even impossible to come by in complex, unsteady hypersonic flows subjected to shocks.

Following this approach, the Euclidean distance matrix Δ of the dataset in low-dimensional space is computed

$$\Delta_{ij} = \|\mathbf{Y}_i - \mathbf{Y}_j\|^2 \quad \text{for} \quad (\mathbf{Y}_i, \mathbf{Y}_j) \in \mathbf{Y}^2. \tag{10}$$

The dataset is subsequently recast into an undirected network with a binary adjacency matrix A, constructed as

$$A_{ij} = \begin{cases} 1 & \text{if } \Delta_{ij} < \epsilon, \\ 0 & \text{otherwise.} \end{cases} \tag{11}$$

Two points are connected with an edge if their Euclidean distance is below a certain threshold ϵ. This threshold is usually chosen as a fraction of the mean of the distance matrix Δ. The influence of the threshold ϵ on the number of clusters will be investigated in Section IV A 2.

Finally, the dataset is progressively split into two communities until the modularity, Q, is maximized. The modularity is defined as the proportion of edges contained within a community over the same proportion for a random reference network. Let k_i be the number of edges pointing towards the data point numbered i, and m the total number of connections within the network. Then, the probability of having an edge between i and j in the random reference network is $k_i k_j / m$. Hence, the modularity Q is defined as,

$$Q = \frac{1}{m} \sum_{ij} (A_{ij} - \frac{k_i k_j}{m}) \delta_{c_i, c_j}, \tag{12}$$

where δ_{c_i,c_j} is the Kronecker delta for the communities of i and j. For instance, $\delta_{c_i,c_j} = 1$ only when i and j belong to the same community. By defining a vector $\mathbf{s}$ where $s_i = 1$ if vertex i belongs to the first group, and $s_i = -1$ otherwise, we can reformulate

$$Q = \frac{1}{2m} \sum_{ij} (A_{ij} - \frac{k_i k_j}{m})(s_i s_j + 1) = \frac{1}{2m} \mathbf{s}^{\mathbf{T}} B \mathbf{s}, \tag{13}$$

where the modularity matrix B is given as,

$$B_{ij} = A_{ij} - \frac{k_i k_j}{m}. \tag{14}$$

Since the graph is undirected, the modularity matrix B is symmetric and the modularity Q represents a Rayleigh quotient for matrix B. In order to maximize Q, we need to choose a vector s that is parallel to the principal eigenvector (corresponding to the largest eigenvalue) of B, $\mathbf{v}$, which can be achieved by setting $s_i = 1$ if $v_i > 0$ and $s_i = -1$ if $v_i < 0$. In order to partition the graph into more than two communities, this algorithm is repeated until the modularity of each subgraph can no longer be increased. A thorough descripion of the full algorithm can be found in [38].

Fig. 6a shows the application of the clustering algorithm to the boundary layer data. The distance matrix Δ is constructed on $\mathbf{Y}$. After running the algorithm on the subsequent adjacency matrix A, two distinct clusters are identified, highlighted by the low distance between the points within a cluster in Fig. 6b.

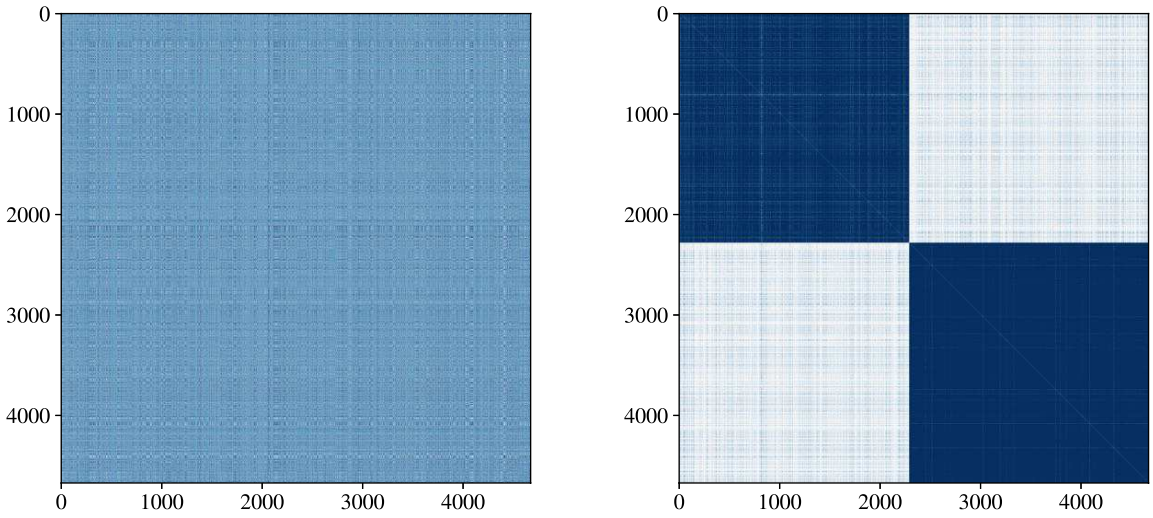

FIG. 6. Demonstration of the Newman algorithm. (a) Original distance matrix Δ of dataset $\mathbf{Y}$. (b) Restored communities from running Newman's algorithm on A, showing two distinct clusters

4. Surrogate model construction

Finally, a surrogate surface is computed on the scattered low-dimensional points of each cluster. Many algorithms can be used to that end, such as kriging [24, 26], artificial neural networks [39], or radial basis functions [17, 18, 40]. In the application of interest to this work, radial basis functions (RBF) provided the best trade-off between performance and accuracy, as well as an easy training step. In fact, the optimal weights of a RBF can be obtained through the solution of a linear system. However, it should be noted that our choice is not definitive and can be easily changed.

Given a set of N_R input points $\mathbf{x}_1, ..., \mathbf{x}_{N_R} \in \mathbb{R}^d$ and the function value at these points $f(\mathbf{x}_1), ..., f(\mathbf{x}_{N_R})$ the radial basis function (RBF) interpolant g is given by

$$g(\phi, x) = \sum_{i=1}^{N_R} \lambda_i \phi(\|\mathbf{x} - \mathbf{x}_i\|) \tag{15}$$

where ϕ is the kernel function whose value depends on the distance $r = \|\mathbf{x} - \mathbf{x}_i\|$ between the evaluation point $\mathbf{x}$ and the center $\mathbf{x}_i$ of the RBF. In this study, the thin-plate spline kernel [41] was used, i.e.,

$$\phi(r) = r^2 log(r). \tag{16}$$

With this particular kernel, the kernel matrix is only conditionally positive definite. To ensure a unique solution for the interpolation weights, the system is augmented by a polynomial $p \in \Pi_m^d$ (space of polynomials of d variables and degree up to m) to the right hand side of Eq. (15) [18], resulting in

$$g(x) = \sum_{i=1}^{N_R} \lambda_i \phi(\|x - x_i\|) + p(x). \tag{17}$$

The extra degrees of freedom are accounted for by enforcing orthogonality of the coefficients with respect to the polynomial space as

$$\sum_{i=1}^{N_R} \lambda_i r(\mathbf{x}) = 0, r \in \Pi_m^d. \tag{18}$$

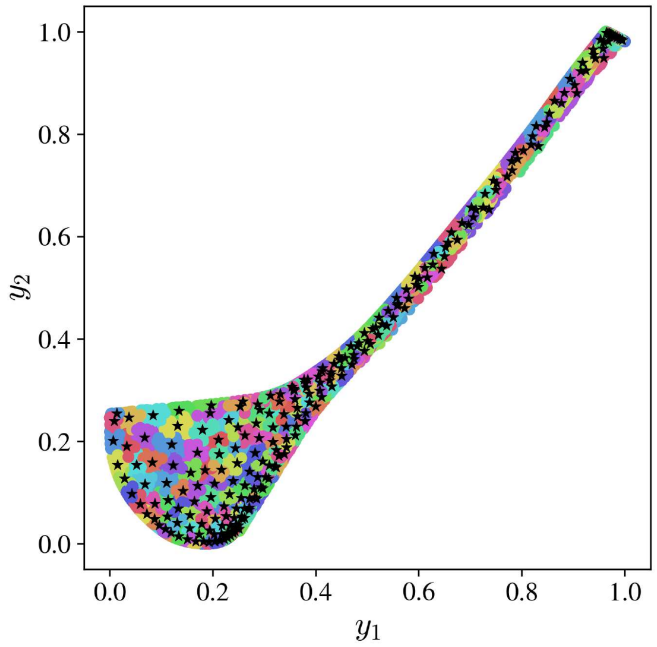

FIG. 7. Tesselation of the low-dimensional space after applying the *k-means* algorithm with $N_R = 250$ on the concatenation of the input $\mathbf{X} \in \mathbb{R}^{N\times D}$ and output $\mathbf{Z} \in \mathbb{R}^{N\times D_Z}$ vectors. Black stars represent the cluster centroids' low-dimensional representation $\mathbf{y}^c_1, ..., \mathbf{y}^c_{N_R}$.

Finally, the polynomial coefficients $\mathbf{c} = [c_1, ..., c_{d+1}]^T$ and the RBF coefficients $\mathbf{\Lambda} = [\lambda_1, ..., \lambda_{N_R}]^T$ are found through the solution of the following linear system

$$\begin{bmatrix} \mathbf{\Phi} & \mathbf{P} \\ \mathbf{P}^T & 0 \end{bmatrix} \begin{bmatrix} \mathbf{\Lambda} \\ \mathbf{c} \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ 0^{d+1} \end{bmatrix} \tag{19}$$

where $\mathbf{P}_i = [1, x_i, ..., x_i^d]$ for $i \in [1, N_R]$ is the polynomial matrix, and $\mathbf{f} = [f(\mathbf{x}_1), ..., f(\mathbf{x}_{N_R})]$ denotes the vector containing the function values at the RBF centers.
Due to the large size of the training set, using one cluster center per training point (i.e. $N_R = N = O(10^5)$) will likely result in overfitting and prohibitive computational cost [42]. Following the recommendations in [42], the interpolant is constructed in two steps. First, the *k-means* algorithm with $N_R \ll N$ clusters is applied on the concatenation of the input and output vector $(\mathbf{X}, \mathbf{Z}) \in \mathbb{R}^{N\times(D+D_Z)}$. The addition of the output vector results in a low within-cluster variance of the outputs and will ultimately improve the surrogate model. The N_R centroids obtained with *k-means*, $\mathbf{x}^c$, are sent to the library to compute the function value vector $\mathbf{f}$. Simultaneously, they are encoded in the low-dimensional space to obtain the N_R cluster centers that will be used to train the RBF $\mathbf{y}^c_1, ..., \mathbf{y}^c_{N_R} \in \mathbb{R}^d$. Fig. 7 shows the resulting tesselation in the embedded space after applying the *k-means* algorithm with $N_R = 250$. The influence of the number of RBF centers on the quality of the surrogate model will be assessed in Section IV A 3.

Following this approach, a single interpolant, g_{c_k}, is constructed for each cluster, c_k; in other words, g_{c_k} is the approximation of the scaled library function $\mathbf{f}$ on the low-dimensional subspace corresponding to cluster c_k. The advantage of having one interpolant per cluster is twofold. First, it allows the surrogate model to best fit a region with a given dynamics of the high-dimensional function, especially in the presence of discontinuities, similar to the approach of Bettebghor et al. [31]. Secondly, as the surrogate model spans a smaller range of input parameters, less centers are required to capture the given dynamics accurately, resulting in a lighter model with faster evaluation time. To enforce continuity of the surrogate surface near the cluster boundaries, the nearest centroids that do not belong to the considered cluster are added to its training set.

B. Coupling to the solver

Once the model is trained, we can replace the calls of the solver to the look-up library with the new lighter model. New points have to go through three steps: (i) Out-of-sample dimensionality reduction, (ii) classification, and (iii) interpolation. These three steps will be described in the following. Let $\mathbf{X}^t \in \mathbb{R}^{N_t \times D}$ denote the stack of all new points.

1. *Out-of-sample encoding*

The low-dimensional representation of $\mathbf{X}^t$ (after proper scaling of $\tilde{\mathbf{X}}^t$) is straightforward. Indeed, the point is simply fed to the encoder portion of the input/output-encoder

$$\mathbf{Y}^t = W_{E,2} f_{ac}(W_{E,1}\mathbf{X}^t + \mathbf{b_{E,1}}) + \mathbf{b_{E,2}}. \tag{20}$$

This results in a fast and inexpensive encoding of the new out-of-sample points. In fact, the time complexity of the encoding step is $O(H \times C_{ac} \times L \times N_t)$, where H is the maximum number of neurons in a layer, C_{ac} is the complexity of the activation function and L is the number of layers in the encoder step of the input/output-encoder.

2. *Classification*

The next step is to determine to which community the new state $\mathbf{Y}^t$ belongs. To this end, after applying Newman's algorithm, a random forest classifier is trained on the resulting clusters. A random forest classifier, formally proposed by Breiman [43], is a collection of n_{tree} tree-based classifiers, $h(\mathbf{x}, \Phi_k), k = 1, ..., n_{tree}$, where Φ_k are identically distributed random vectors. Each tree votes for the most likely class of input vector $\mathbf{x}$, and the majority wins. Fig. 8 shows the confusion matrix C of the classifier trained on the two clusters obtained above with $n_{tree} = 20$. The clusters of the embedded new points $\mathbf{Y}^t$, not seen during the training-phase of the classifier, are predicted and compared to the true clusters (given by Newman's algorithm). The off-diagonal values count the number of points that are assigned to the wrong cluster. All off-diagonal values are zero, demonstrating the ability of the classifier to correctly predict the community of an out-of-sample point.

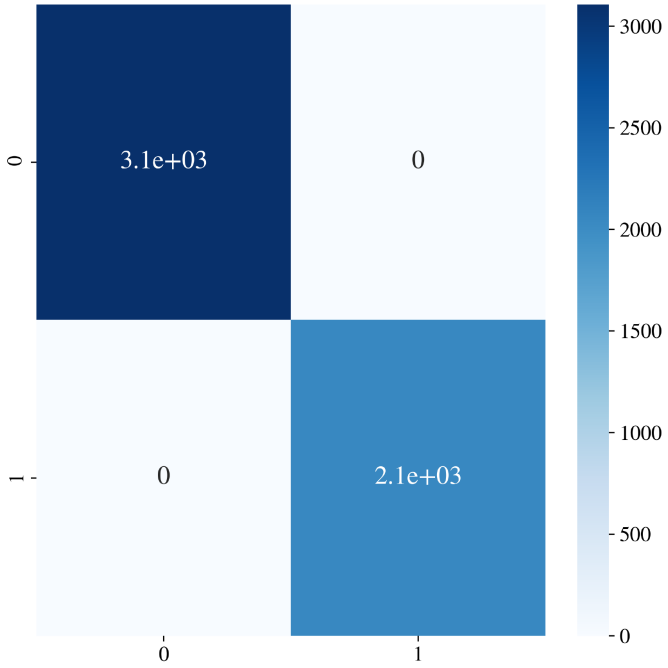

FIG. 8. Confusion matrix C of the random forest classifier ($n_{tree} = 20$), tested on unseen data points during training. C_{ij} is the number of points belonging to cluster c_j predicted to be in cluster c_i.

The time complexity of the induction time of the classifier is $O(n_{tree} \times N_t log(N_t))$ (see the book by Witten, Frank and Hall [44] for a demonstration), where n_{tree} is the number of decision trees in the random forest. The logarithmic term accounts for the worst case scenario for the maximum depth of each tree. However, the maximum depth is usually set to a smaller value, resulting in a complexity of $O(n_{tree} \times depth \times N_t)$, which results in relatively fast classification times.

3. *Interpolation*

Finally, once $\mathbf{Y}^t$ has been found to belong to cluster c_k, the corresponding RBF g_{c_k} is called to evaluate the thermochemical properties $\widehat{\mathbf{Z}}^t$ of the mixture at that state $\mathbf{X}^t$,

$$\widehat{\mathbf{Z}}^t = g_{c_k}(\mathbf{Y}^t) = \sum_{i=1}^{N_R} \lambda_i \phi(\mathbf{Y}^t - \mathbf{Y}_i^c). \tag{21}$$

The hat denotes the predicted value by the reduced library, as opposed to the true value that would have been given by the target library, $\mathbf{Z}^t$. Finally, these properties are re-scaled according to

$$\tilde{\widehat{\mathbf{Z}}}^t = \widehat{\mathbf{Z}}^t * (\tilde{\mathbf{Z}}_{max} - \tilde{\mathbf{Z}}_{min}) + \tilde{\mathbf{Z}}_{min} \tag{22}$$

and passed back to the flow solver.

The time complexity of each surrogate model is given by $O(C_{RBF} \times N_t \times N_R \times d)$, where C_{RBF} is the complexity of the kernel function. The time limiting part of the RBF interpolation is the calculation of the distance matrix that scales with $O(N_t \times N_R \times d)$. Hence, using a relatively small number N_R of RBF centers, compared to N_t, will greatly improve the performance of the surrogate model. It should be noted that the dimensionality reduction directly contributes to the added performance of the surrogate model as $d < D$.

4. *Global performance*

The time complexity of the whole algorithm can be recovered by adding the time complexity of each of the three steps. Hence, the total time complexity of the algorithm can be written as $O(C_{ML}N)$, where $C_{ML} = O(HC_{ac}L + n_{tree}depth + dN_RC_{RBF})$.

III. APPLICATION TO HYPERSONIC FLOWS IN CHEMICAL NON-EQUILIBRIUM

In this section, the general equations governing hypersonic flows in chemical non-equilibrium are first recalled, followed by a brief description of the numerical framework used to solve these equations. Then, the benchmark case, a Mach-10 adiabatic laminar boundary layer in chemical non-equilibrium, initially studied by Marxen et *al* [2, 3], is presented.

A. Governing conservation equations

The nondimensional Navier-Stokes equations for a mixture of multiple species $\mathcal{S}$ are presented in Eqs. (23–26).

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \tag{23}$$

$$\left\{ \frac{\partial \rho_s}{\partial t} + \nabla \cdot (\rho_s \mathbf{u} + \rho_s \mathbf{V}_s) = \dot{\omega}_s \right\}, \ \forall\, s \in \mathcal{S} \tag{24}$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u}) = -\nabla p + \nabla \cdot \boldsymbol{\tau} \tag{25}$$

$$\frac{\partial \rho e_0}{\partial t} + \nabla \cdot (\rho h_0 \mathbf{u}) = \nabla \cdot (\boldsymbol{\tau} \cdot \mathbf{u}) - \nabla \cdot \mathbf{q} \tag{26}$$

Equation (23) is the continuity equation, describing mass conservation in the system. Equation (24) corresponds to the set of mass conservation equations for each species, with the net production rate terms, $\dot{\omega}_s$, appearing on their

right-hand side. In order to ensure global mass conservation, in the case of a finite-rate reacting mixture with a varying composition, Eq. (23) needs to be solved together with Eq. (24) for all but one species. The omitted species is selected based on numerical considerations, avoiding species with the smallest concentrations.

The nondimensional quantities are the time, t, the density, ρ, the velocity, $\mathbf{u}$, the pressure, p, the stress tensor, $\boldsymbol{\tau}$, the total energy, e_0, the total enthalpy, h_0, the heat flux, $\mathbf{q}$, as well as, the partial density, ρ_s, the net mass production rate, $\dot{\omega}_\mathrm{s}$, and the diffusion velocity, $\mathbf{V}_\mathrm{s}$, for the species s. More details regarding the derivation of the equations and the validity of our invoked assumptions are provided in [9, 45, 46]. There are $N_\mathrm{s}-1$ conservation equations for the species, so the total problem involves $N_\mathrm{s}+4$ equations. The stress tensor, $\boldsymbol{\tau}$, and the heat flux, $\mathbf{q}$, are computed as

$$\boldsymbol{\tau} = \frac{\mu}{Re_\infty}\left(\nabla\mathbf{u} + (\nabla\mathbf{u})^\mathrm{T} - (\nabla\cdot\mathbf{u})\,\mathrm{I}\right), \tag{27}$$

$$\mathbf{q} = -\frac{\kappa}{Re_\infty Pr_\infty Ec_\infty}\nabla T + \sum_{\mathrm{s}\in\mathcal{S}} \rho_\mathrm{s} h_\mathrm{s} \mathbf{V}_\mathrm{s}. \tag{28}$$

The nondimensional Reynolds number, Re_∞, and Prandtl number, Pr_∞, are defined in the free stream, at the domain inlet. The Eckert number, Ec_∞, is also computed at the free stream and is equal to one by design. These nondimensional quantities are defined with dimensional quantities (distinguishable by the tilde) as

$$Re_\infty = \frac{\tilde{\rho}_\infty \tilde{c}_\infty \tilde{L}_\mathrm{ref}}{\tilde{\mu}_\infty}, \quad Pr_\infty = \frac{\tilde{\mu}_\infty \tilde{c_p}_\infty}{\tilde{\kappa}_\infty}, \quad Ec_\infty = \frac{\tilde{c}_\infty^2}{\tilde{c_p}_\infty \tilde{T}_\mathrm{ref}} \tag{29}$$

where $\tilde{T}_\mathrm{ref} = (\tilde{\gamma}_\infty - 1)\tilde{T}_\infty$, and c stands for the speed of sound. μ denotes the dynamic viscosity, κ the frozen thermal conductivity, and c_p the specific heat at constant pressure.

B. Thermodynamic and chemical models for a mixture in chemical non-equilibrium

In the following, the closure of the governing equations when assuming a mixture in chemical non-equilibrium is detailed. In general, a reacting gas in a high-enthalpy flow is considered as a multi-component mixture that consists of a set of species $\mathcal{S}$ interacting through a defined network of reactions. The presence of the species makes the chemical reaction and diffusion terms in the governing equations significant, which in turn require to be modeled using various assumptions.

For a multi-component gas mixture, the global mixture properties are derived from the species properties based on

$$\rho = \sum_{\mathrm{s}\in\mathcal{S}} \rho_\mathrm{s}, \quad e = \sum_{\mathrm{s}\in\mathcal{S}} Y_\mathrm{s} e_\mathrm{s}, \quad h = \sum_{\mathrm{s}\in\mathcal{S}} Y_\mathrm{s} h_\mathrm{s} \tag{30}$$

where the species mass fraction is $Y_\mathrm{s} = \frac{\rho_\mathrm{s}}{\rho}$, with $\sum_{\mathrm{s}\in\mathcal{S}} Y_\mathrm{s} = 1$. The mixture thermodynamic and transport properties depend, generally, on their composition, and any two thermodynamic properties, for example the temperature and the pressure, which define the thermodynamic state of the mixture. The composition is commonly taken as an independent variable. It can potentially become dependent on other thermodynamic quantities for the special cases of frozen chemistry or chemical equilibrium. However, these cases are not described in this study. The individual species properties are accurately computed by kinetic theory and statistical mechanics [47, 48]. Hence the composition and the two thermodynamic properties can be concatenated into the thermodynamic state vector $\tilde{\mathbf{q}}_{th}$. This defines the relation $\tilde{\mathbf{Z}} = \tilde{f}(\tilde{\mathbf{q}}_{th})$ on which the algorithm is based.

When finite-rate chemistry is not neglected, the species mass production rate, $\dot{\omega}_\mathrm{s}$, and the species diffusion velocities, $\mathbf{V}_\mathrm{s}$, need to be modeled. For a general case, a set of reactions, $\mathcal{R}$, is considered depending on the mixture in question. Each reaction r is characterized by a reaction rate, R_r, which is computed by the forward rate, k_{f_r}, and the backward rate, k_{b_r}. These rates, in turn, are obtained according to experimentally or theoretically calibrated Arrhenius formulas in the form $k_{f_\mathrm{r}} = C_\mathrm{r} T^{n_\mathrm{r}} \exp\left(T_{a_\mathrm{r}}/T\right)$, and $k_{b_\mathrm{r}} = k_{f_\mathrm{r}}/K_{\mathrm{eq}_\mathrm{r}}(T)$, where $K_{\mathrm{eq}_\mathrm{r}}$ is the reaction equilibrium constant at the specific conditions. This description is given in Eq. (31) for a generic reaction.

$$\text{Reaction (r):} \qquad \sum_{\mathrm{s}\in\mathcal{S}} \nu'_{\mathrm{r,s}} S_\mathrm{s} \underset{k_{b_\mathrm{r}}}{\overset{k_{f_\mathrm{r}}}{\rightleftharpoons}} \sum_{\mathrm{s}\in\mathcal{S}} \nu''_{\mathrm{r,s}} S_\mathrm{s} \tag{31}$$

The net reaction rate is then given by

$$R_\mathrm{r} = \left[k_{f_\mathrm{r}} \Pi_\mathrm{s} \left(\frac{\rho_\mathrm{s}}{M_\mathrm{s}}\right)^{\nu'_{\mathrm{r,s}}} - k_{b_\mathrm{r}} \Pi_i \left(\frac{\rho_\mathrm{s}}{M_\mathrm{s}}\right)^{\nu''_{\mathrm{r,s}}}\right] \cdot \sum_{\mathrm{s}\in\mathcal{S}} \left(Z_{\mathrm{r,s}} \frac{\rho_\mathrm{s}}{M_\mathrm{s}}\right) \tag{32}$$

where the species molar mass is M_s and the efficiency of species s as a third-body in reaction r is $Z_{\mathrm{r,s}}$. The net mass production rates for species s from all reactions are obtained as

$$\dot{\omega}_\mathrm{s} = M_\mathrm{s} \sum_{\mathrm{r}\in\mathcal{R}} \left(\nu''_{\mathrm{r,s}} - \nu'_{\mathrm{r,s}}\right) R_\mathrm{r}. \tag{33}$$

The diffusion flux $\mathbf{J}_\mathrm{s} = \rho_\mathrm{s}\mathbf{V}_\mathrm{s}$ appearing in Eq. (24) also needs to be modeled. In this study, we use a simple expression based on Fick's law with a mass correction [49, 50] given by

$$\mathbf{J}_\mathrm{s} = -cM_s D_s \nabla Y_\mathrm{s} + cY_s \sum_{\mathrm{i}\in\mathrm{S}} M_\mathrm{i} D_\mathrm{i} \nabla Y_\mathrm{i}. \tag{34}$$

Here, $c = \sum_{\mathrm{s}\in\mathcal{S}}(\rho_\mathrm{s}/M_\mathrm{s})$, and D_s is the averaged diffusion coefficient for species s, defined below

$$D_\mathrm{s} = \frac{1 - X_\mathrm{s}}{\sum_{\mathrm{r}\neq\mathrm{s}} X_\mathrm{r}/D_{\mathrm{s,r}}}. \tag{35}$$

The thermochemical library Mutation++ [1] is used to compute thermodynamic and transport properties at different conditions. The reader is referred to the description provided in [1, 48] for further details on the library.

C. Numerical framework

The governing equations are solved using a high-fidelity in-house flow solver for the direct numerical simulation of hypersonic flows in chemical non-equilibrium, as described in [51]. Compact schemes are used for a spatial discretization, and explicit time-integration is performed using Runge-Kutta schemes. More details about the discretization can be found in the aforementioned paper.

1. *Coupling with Mutation++*

The thermodynamic and transport properties, as well as the source terms resulting from the chemical kinetics models, are extracted from the Mutation++ library [1] to close the governing equations, Eqs. (23–26). The library, originally written in `C++`, is coupled with the numerical solver using a wrapper in `Fortran 95`.

For the remainder of this study, we will only consider an air mixture composed of five species $\mathcal{S} = [N_2, O_2, NO, N, O]$. However, we stress that the algorithm can be applied to any gas mixture. At each grid point, we have access to the local thermodynamic state and composition $\tilde{\mathbf{q}}_{th}$ (in dimensional units),

$$\tilde{\mathbf{q}}_{th} = \begin{bmatrix} \rho & \rho e & \rho_N & \rho_O & \rho_{NO} & \rho_{O_2} \end{bmatrix}. \tag{36}$$

$\tilde{\mathbf{q}}_{th}$ is then passed to Mutation++, which returns the necessary thermochemical properties needed to close the reactive compressible Navier-Stokes Eqs. (23–26),

$$\tilde{\mathbf{z}} = \begin{bmatrix} \mu & \kappa & P & T & h_s & \omega_s & D_s \end{bmatrix} \quad s \in \mathcal{S}. \tag{37}$$

Hence, calls to Mutation++ can be seen as an input/output problem $\tilde{\mathbf{z}} = \tilde{f}(\tilde{\mathbf{q}}_{th})$ where the function $\tilde{f}$ represents the library.

D. Mach-10 boundary layer – test case

In this section, the test-case used to showcase the applicability and performance of the proposed algorithm is presented. We simulate a Mach-10 adiabatic flat-plate boundary layer in Earth atmosphere, based on the work of Marxen *et al* [2, 3]. The thermodynamic and freestream conditions are presented in Table I. Details of the numerical grid can be found in [51].

Test case	
M_∞	10
Re_∞	10^5
$T_\infty[K]$	350
$p_\infty[Pa]$	3596

TABLE I. Thermodynamic and free stream conditions for Mach-10 adiabatic flat-plate boundary layer test case (from [2]).

1. *Base flow*

This case has already been validated against literature in [51]. Due to the wall temperature approaching $T_{wall} \approx 4900K$ near the inflow, N_2 and O_2 rapidly start to produce their monoatomic counterpart, as well as NO, through endothermic chemical reactions. This is illustrated in Fig. 9c presenting all the mass fraction profiles at the streamwise location of $Re_x = 2000$. Close to the wall, the O_2 mass fraction decreases while O and NO are created. To a smaller extent, N is also created through N_2 dissociation. The mean flow and temperature profiles at a streamwise Reynolds number of $Re_x = 2000$ are presented in Fig. 9a and Fig. 9b, alongside the same case simulated using a thermally perfect gas assumption (ignoring non-equilibrium effects). The base flows differ significantly depending on the assumption used for the gas. Wall temperature decreases significantly from a thermally perfect gas to a finite-rate chemistry assumption, and the boundary layer becomes thicker. Overall, these results highlight and corroborate the importance of including chemical non-equilibrium effects for the accurate simulation of high-enthalpy hypersonic flows.

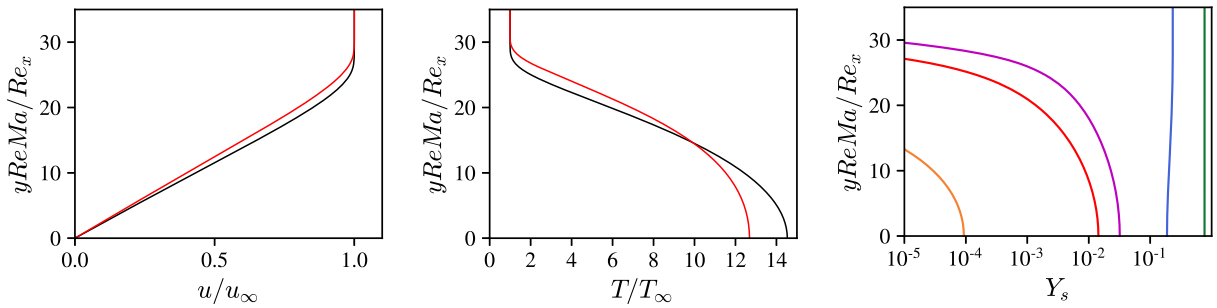

FIG. 9. Profiles of (a) streamwise velocity, (b) temperature, (c) species mass fractions from left to right N, NO, O, O_2 and N_2, at $Re_x = 2000$. (a,b) Red line: simulation in chemical non-equilibrium, black line: perfect gas assumption.

IV. RESULTS

The results are presented in two parts. First, each step of the off-line training is assessed, and then the performance of the model in predicting the output quantities when coupled to the Navier-Stokes solver is analyzed.

A. Model training

$N = 100,000$ thermodynamic state vectors $\tilde{\mathbf{q}}_{th}$ are sampled from the converged solution and concatenated into the input dataset $\tilde{\mathbf{X}}$. The corresponding Mutation++ output dataset $\tilde{\mathbf{Z}}$ is also obtained.

1. *Input/output encoding*

The architecture and hyperparameters of the input/output-encoder used for this test case are provided in Table II.

Architecture

Encoder		Decoder	
Layer	Size	Layer	Size
Input	6	Latent space	2
Fully connected	12	Fully connected	6
Fully connected	6	Fully connected	12
Latent space	2	Output	18

Hyperparameters

Parameter	Value	Parameter	Value
Learning rate	$1e^{-3}$	Epochs	2000
Loss	Mean-squared error	Batch size	256
Activation function	tanh	Optimizer	Adam (keras default)

TABLE II. Architecture and hyperparameters of the input/output-encoder used to train the IO-E network on the $Ma = 10$ adiabatic flat plat boundary layer test case.

An important parameter is the number of dimensions required to properly unfold the input manifold. To this end, the architecture and hyperparameters, as well as the random seed to initialize the network weights, are held constant, and only the dimension of the latent space is varied. We follow the reconstruction loss $\|\widehat{\mathbf{Z}} - \mathbf{Z}\|_2$ of the testing set as a function of the latent space dimension d. As seen in Fig. 10, the loss saturates after $d = 2$, suggesting that two dimensions are sufficient to represent the input manifold, originally in $\mathbb{R}^6$.

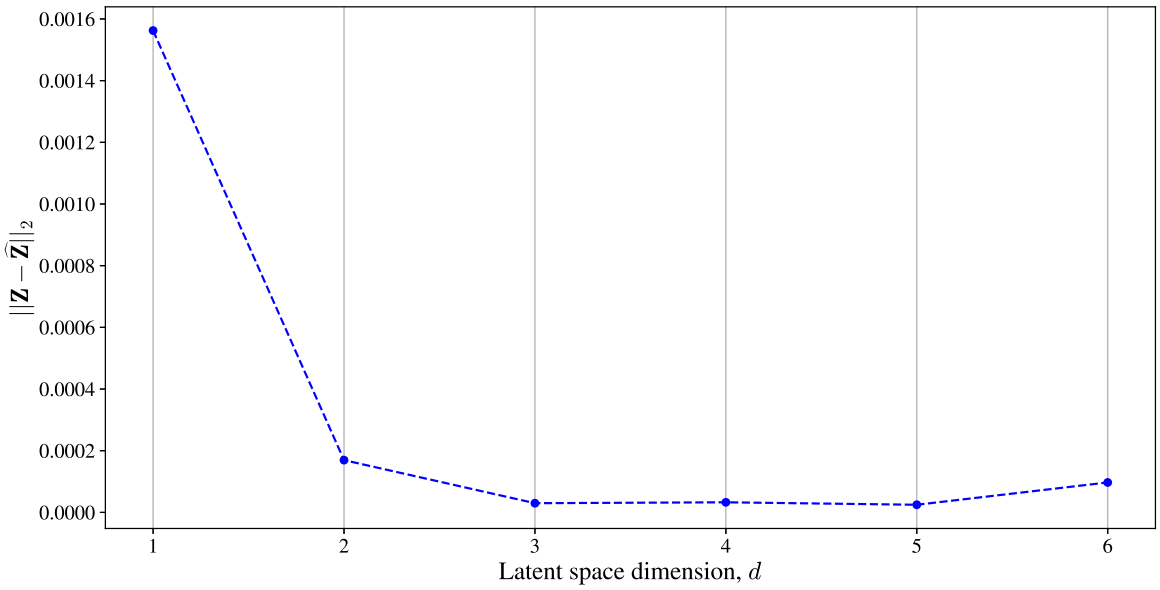

FIG. 10. Reconstruction loss $\|\widehat{\mathbf{Z}} - \mathbf{Z}\|_2$ of the embedding done by IO-E with respect to the number of latent space dimensions d.

2. Newman's community clustering

The second step is to cluster the data on the low-dimensional manifold using Newman's algorithm. The only hyperparameter needed for Newman's clustering algorithm is the threshold ϵ to determine the adjacency matrix A from the

distance matrix Δ, as described in Section II A 3. To showcase the robustness of the number of clusters with respect to the threshold ϵ, the algorithm is applied over a range of thresholds, chosen as multiples of the mean of the distance matrix $\overline{\Delta}$. As shown in Fig. 11, over the range tested, the algorithm returns $N_c = 2$ clusters before over-fitting with extreme values.

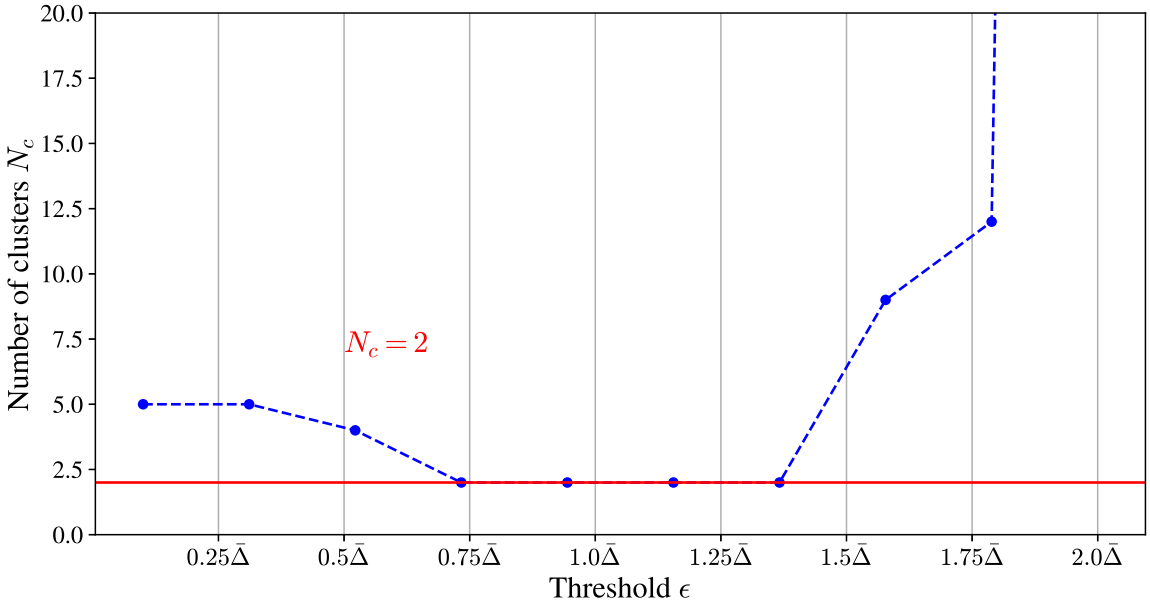

FIG. 11. Number of clusters N_c obtained with Newman's algorithm as a function of the threshold ϵ, given as multiples of the distance matrix mean $\overline{\Delta}$.

Fig. 12a shows the two clusters obtained in the embedded space. As expected, the two clusters define different regions in the reduced space. To gain more physical insight, Fig. 12b shows randomly selected points of the training set, mapped back to their original location in the Cartesian space and colored by their cluster number. Contours of temperature T in Kelvin are added. Each cluster fills a region of the flow with different levels of chemical non-equilibrium. The blue cluster represent the free stream, where temperature is low and chemistry is frozen. Alternatively, the red cluster corresponds to the near-wall region with dissociated species and high temperatures.

The random forest classifier has been trained with $n_{tree} = 20$. This number proved sufficient to obtain an acceptable prediction accuracy of new points clusters, as seen in Fig. 8 above.

3. *Surrogate model construction*

Once the clusters are determined, a separate RBF interpolant is trained for each cluster as described in Section II A 4. To enforce continuity of the surrogate through the clusters, the nearest centroids shared between the two clusters are added to the training set.

To assess the number of centroids needed, the RBFs are trained simultaneously, for both clusters, with the same number of centroids N_R (note that this number can be varied to accommodate clusters of different sizes). The error of the model for the testing set $\|\widehat{\mathbf{Z}} - \mathbf{Z}\|_2$ is plotted against the number of centroids N_R in Fig. 13 (blue curve). As expected, the error decreases as the number of centroids is increased. To assess potential overfitting, one can follow the evolution of the mean of the squared RBF coefficients $\overline{\Lambda^2}$,

$$\overline{\Lambda^2} = \frac{1}{N_R} \sum_{i=1}^{N_R} \lambda_i^2 \tag{38}$$

where a high value will likely indicate overfitting. On the right axis of Fig. 13, we see that $\overline{\Lambda^2}$ increases with the number of RBF centroids. Therefore, choosing the right N_R is a trade-off between low error and low overfitting. In

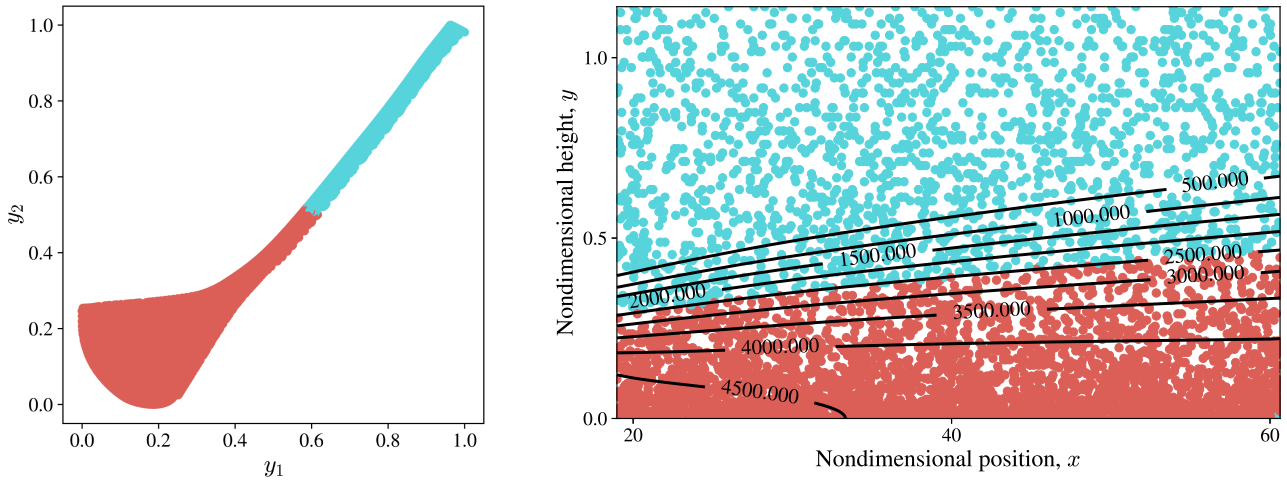

FIG. 12. Training points $\mathbf{Y}$ colored by their cluster number. (a) In the latent space found by IO-E. (b) At the Cartesian location they were sampled from, with contours of temperature T in Kelvin.

this case, a value of $N_R = 250$ for each cluster has been retained.

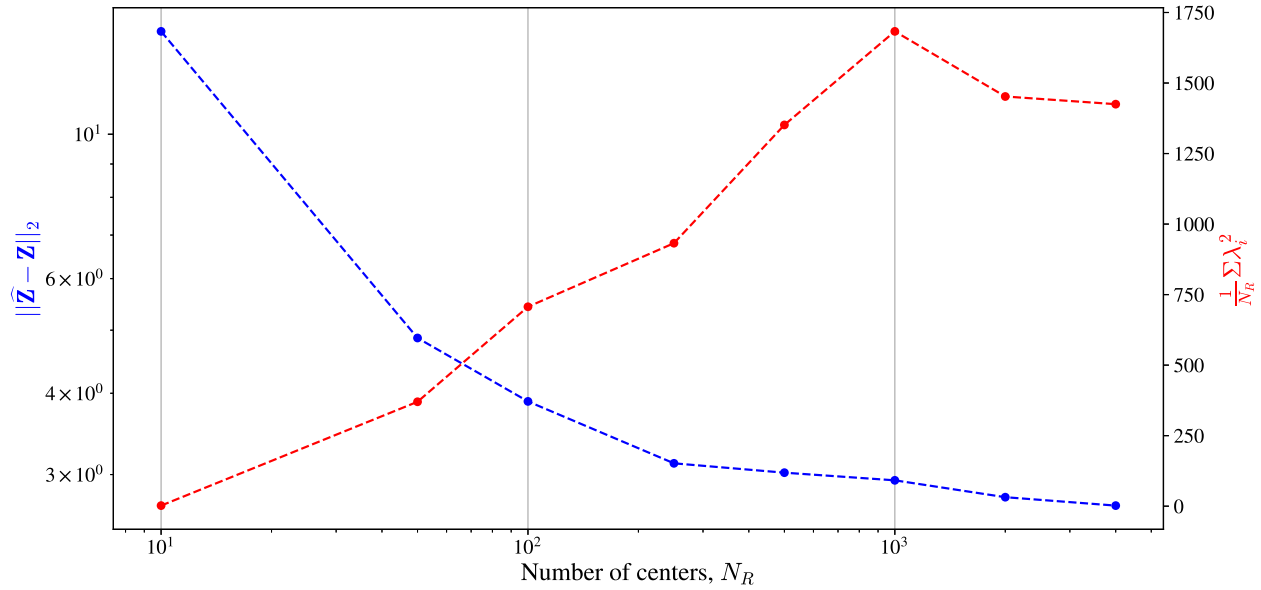

FIG. 13. Left axis: reconstruction error of surrogate model $\|\widehat{\mathbf{Z}} - \mathbf{Z}\|_2$ on a testing set. Right axis: mean of squared RBF coefficients $\overline{\Lambda^2}$. Both with respect to the number of RBF centers N_R used in training.

B. Model accuracy

The reduced library is tested (off-line) on a full snapshot (which includes also the training points used to build the model) to assess the capacity of the model to interpolate new points not encountered during training. Four

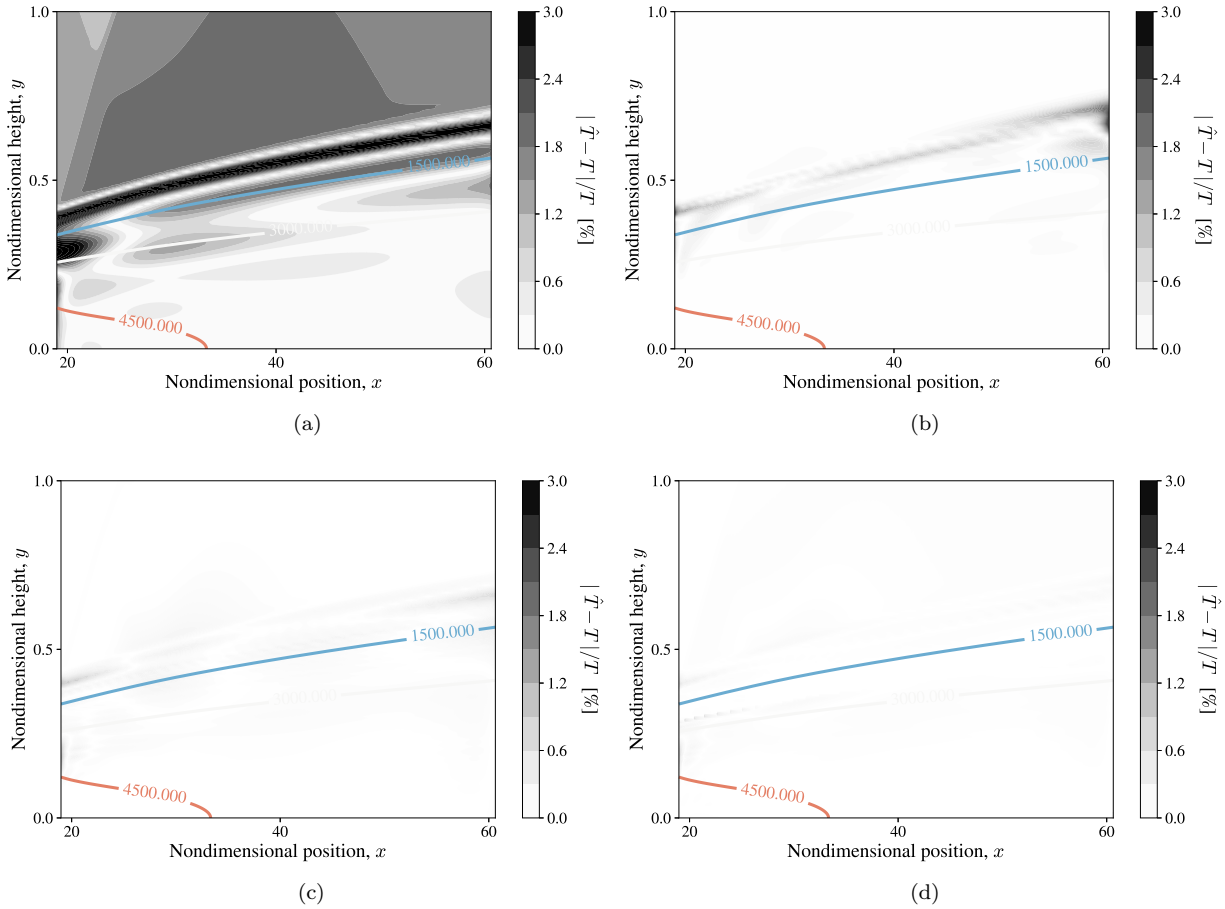

FIG. 14. Comparison of the relative temperature error $|\hat{T}-T|/T$ in percent with contours of temperature. (a) Model 1: full IO-E (b) Model 2: $d=6, N_c=1, N_R=250$ (c) Model 3: $d=2, N_c=1, N_R=250$ (d) Model 4: $d=6, N_c=2, N_R=250$

configurations of the data-driven model are tested: (i) model 1, using the full IO-E for prediction (ii) model 2, with no dimensionality reduction and no clustering ($d=6$, $Nc=1$, $N_R=250$), (iii) model 3, with dimensionality reduction, but without clustering, ($d=2$, $Nc=1$,$N_R=250$), and (iv) model 4, with both dimensionality reduction and clustering ($d=2$, $Nc=2$,$N_R=250$).

Fig. 14 displays the relative error (as a percentage) between the temperature T given by Mutation++ and the prediction of the data-driven model $\hat{T}$, for the four configurations enumerated above; contours of temperature in Kelvin are also added to highlight the evolution of the flow. First, the prediction of model 1 produces the highest error by a wide margin. This showcases the difficulty of properly training a neural network for prediction in high dimensions. The figure shows the maximum relative error in all three remaining models to be only a few percent, and located around the edge of the boundary layer, where the gradients are strongest. In addition, the error in model 2 is higher than the error of model 3, even though some information is lost in the latter due to the encoding step. This is a direct consequence of the curse of dimensionality: as the number of dimensions increases, the sampling volume in the input space increases exponentially. However, in this case, we kept the number of RBF centers N_R fixed. To get to the same level of accuracy, N_R should be increased in model 2, resulting in a performance loss. Finally, the error decreases slightly from model 3 to model 4. This improvement originates from the clustering step. In fact, each of the two clusters have $N_R=250$ centers. Hence, the input space is actually populated with $N_R=500$ RBF centers in model 4. According to Fig. 13, this likely improves the accuracy of the surrogate surface. The added number of centroids, however, does not result in a loss of performance. Assuming that a fraction α of the N_t query points are in cluster 1, then $1-\alpha$ query points are in cluster 2. The evaluation time of the surrogate surface linked to cluster 1 then scales roughly as $\alpha d N_t N_R C_{RBF}$. Similarly, for cluster 2, it scales as $(1-\alpha) d N_t N_R C_{RBF}$. Finally, the total evaluation time, i.e., the sum of the two, remains $d N_t N_R C_{RBF}$. This can be easily generalized to a higher number of clusters.

These results demonstrate that the preprocessing steps involved in the construction of the model improve overall

performance while maintaining a high level of accuracy.

C. Model stability

The resulting data-driven model (model 4, with all pre-processing steps) is coupled to the flow solver in a time-marching simulation. Starting from the solution obtained with Mutation++, the simulation is restarted using the reduced library only, also referred as a closed-loop prediction. After running for a couple of flow-through times, the solution remains stable. The base-flow profiles are compared for various quantities of interest in Fig. 15. Excellent agreement is found between the profiles. This validates the accuracy and suitability of the data-driven model to simulate hypersonic flows in chemical non-equilibrium over the enthalpy range observed during the training step.

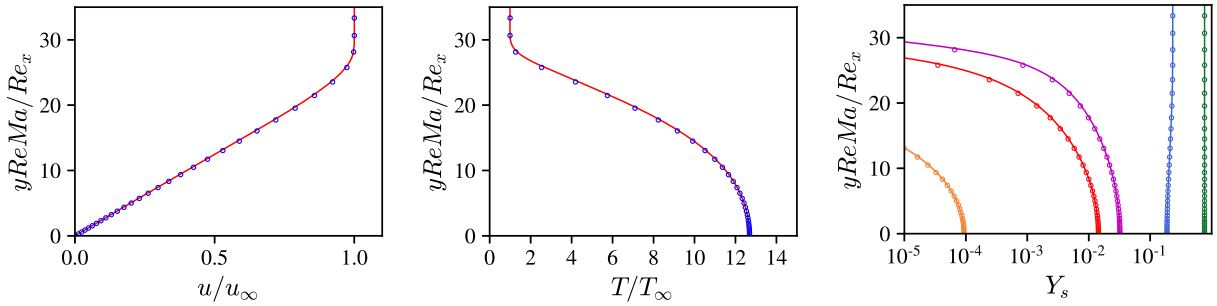

FIG. 15. Comparison of profiles of (a) streamwise velocity, (b) temperature, (c) species mass fractions from left to to right N, NO, O, O_2 and N_2 at $Re_x = 2000$. Solid line and symbols correspond to the solution obtained using Mutation++ and the data-driven model, respectively.

D. Model performance

To compare the performance of the data-driven model to the full library, we performed a scaling study. Mutation++ is a serial library, hence its time complexity can be expressed as $O(C_{M++}N_t)$ where N_t is the number of independent, evaluated thermodynamic states. C_{M++} is empirically determined in Fig. 16. For the data-driven model, we recall that its time complexity is $O(C_{ML}N_t)$, where $C_{ML} = O(HC_{ac}L + n_{tree}depth + N_R C_{RBF})$.

Both curve fits, shown in Fig. 16, suggest that, in practice, both models scale as $O(N_t)$ with exponents close to unity. The ratio of the prefactor is $C_{ML}/C_{M++} \approx 0.52$. We can therefore expect a 52% CPU gain by using the data-driven model instead of Mutation++. In fact, we assessed a CPU time reduction of 50% during the simulation, with a grid of size $N \approx 400,000$. This confirms a speedup through the use of a surrogate model. Moreover, fine tuning of the hyperparameters may allow even higher CPU gains as C_{ML} is proportional to a linear combination of the hyperparameters. Finally, we stress that the data-driven algorithm is an unoptimized python implementation competing with a compiled C++ library. We thus expect even larger CPU gains in the future with an optimized implementation and added adaptivity.

V. CONCLUSIONS

In this paper, we presented a novel technique to reduce any high-dimensional look-up library to a lower-dimensional surrogate, and thus reduce the CPU costs of numerical simulations that rely on these libraries. Several machine learning techniques have been used: encoding based on deep neural networks, community clustering, surrogate modeling and classification in a three-step learning phase. In the first step, the proposed input/output-encoder architecture has been shown to outperform partial least-squares (PLS) for dimensionality reduction of input/output relations. Clustering was performed using Newman's algorithm. It discovered physically consistent clusters in the low-dimensional latent space without *a-priori* knowledge of the number of clusters. Then, a random-forest classifier was trained, which reliably predicted the cluster of previously unencountered data points. Finally, a radial basis function network

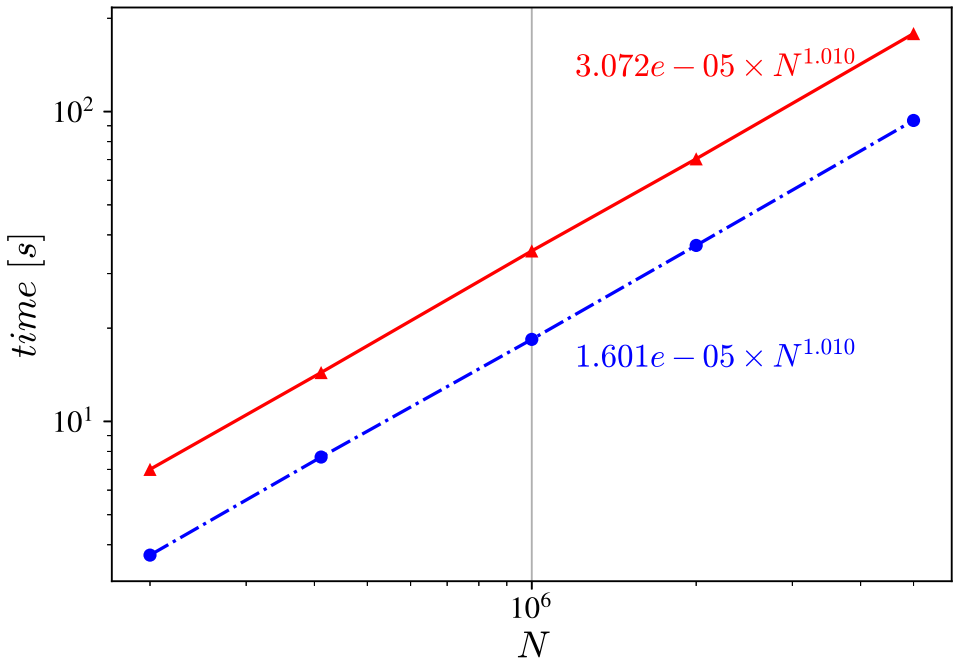

FIG. 16. Comparison of the time complexity of Mutation++ (solid line with arrows) and the data-driven model (dash dotted line with circles). The best non-linear least-squares fit of the form CN^{α} is added.

was constructed on each cluster to obtain a continuous and local representation of the library via a reduced-order surrogate model. The combination of these pre-processing steps has been shown to improve the efficiency of the model on our test case of a Mach-10 adiabatic boundary layer in chemical non-equilibrium. After training, the model replaced Mutation++ and converged rapidly to a new solution. The newly computed base flow is recovered accurately when compared to the true solution (obtained with Mutation++). During this demonstration, we observed a 50 % CPU time decrease to compute the thermochemical properties of the mixture. This computational framework can be readily ported into other application fields to accelerate simulations that rely on high-dimensional look-up tables to model complex flow behavior such as combustion, phase-change or fluid-particle interactions.
Finally, future steps in algorithmic development will include on-the-fly adaptivity of the model to tackle unsteady flow problems, and leveraging analytical solutions during the initial training set.

ACKNOWLEDGMENTS

This work was supported by the Imperial College London - CNRS PhD Joint Program and was granted access to the HPC/AI resources of TGCC under the allocation 2021-A0102B12426 and 2022-A0122B13432 made by GENCI. Part of the calculations were also performed using MeSU computing platform at Sorbonne University.

[1] J. B. Scoggins, V. Leroy, G. Bellas-Chatzigeorgis, B. Dias, and T. E. Magin, Mutation++: Multicomponent thermodynamic and transport properties for ionized gases in c++, SoftwareX **12**, 100575 (2020).
[2] O. Marxen, T. E. Magin, E. S. G. Shaqfeh, and G. Iaccarino, A method for the direct numerical simulation of hypersonic boundary-layer instability with finite-rate chemistry, Journal of Computational Physics **255**, 572 (2013).
[3] O. Marxen, G. Iaccarino, and E. S. G. Shaqfeh, Nonlinear instability of a supersonic boundary layer with two-dimensional roughness, Journal of Fluid Mechanics **752**, 497 (2014).
[4] G. V. Candler, Rate Effects in Hypersonic Flows, Annual Review of Fluid Mechanics **51**, 379 (2019).
[5] M. Di Renzo and J. Urzay, Direct numerical simulation of a hypersonic transitional boundary layer at suborbital enthalpies, Journal of Fluid Mechanics **912** (2021).

[6] X. Zhong and X. Wang, Direct numerical simulation on the receptivity, instability, and transition of hypersonic boundary layers, Annual Review of Fluid Mechanics **44**, 527 (2012).

[7] M. Panesi, A. Munafò, T. Magin, and R. Jaffe, Nonequilibrium shock-heated nitrogen flows using a rovibrational state-to-state method, Physical Review E **90**, 013009 (2014).

[8] P. Paredes, M. M. Choudhari, F. Li, J. S. Jewell, and R. L. Kimmel, Nonmodal growth of traveling waves on blunt cones at hypersonic speeds, AIAA Journal **57**, 4738 (2019).

[9] J. D. Anderson, *Hypersonic and High-Temperature Gas Dynamics*, 3rd ed. (American Institute of Aeronautics and Astronautics (AIAA), Washington, DC, 2019).

[10] B. Franzelli, B. Fiorina, and N. Darabiha, A tabulated chemistry method for spray combustion, Proceedings of the Combustion Institute **34**, 1659 (2013).

[11] M. Pini, A. Spinelli, G. Persico, and S. Rebay, Consistent look-up table interpolation method for real-gas flow simulations, Computers & Fluids **107**, 178 (2015).

[12] O. Marxen, T. E. Magin, G. Iaccarino, and E. S. G. Shaqfeh, A high-order numerical method to study hypersonic boundary-layer instability including high-temperature gas effects, Physics of Fluids **23**, 084108 (2011).

[13] S. B. Pope, Computationally efficient implementation of combustion chemistry using in situ adaptive tabulation, (1997).

[14] G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of control, signals and systems **2**, 303 (1989).

[15] Y. LeCun, Une procedure d'apprentissage ponr reseau a seuil asymetrique, Proceedings of Cognitiva 85 , 599 (1985).

[16] W. Liu and S. Batill, Gradient-enhanced neural network response surface approximations, in *8th Symposium on Multidisciplinary Analysis and Optimization* (2000) p. 4923.

[17] D. S. Broomhead and D. Lowe, *Radial basis functions, multi-variable functional interpolation and adaptive networks*, Tech. Rep. (Royal Signals and Radar Establishment Malvern (United Kingdom), 1988).

[18] M. D. Buhmann, Radial basis functions, Acta numerica **9**, 1 (2000).

[19] R. Jin, W. Chen, and T. W. Simpson, Comparative studies of metamodelling techniques under multiple modelling criteria, Structural and multidisciplinary optimization **23**, 1 (2001).

[20] J. Peter, M. Marcelet, S. Burguburu, and V. Pediroda, Comparison of surrogate models for the actual global optimization of a 2d turbomachinery flow, in *Proceedings of the 7th WSEAS international conference on simulation, modelling and optimization* (Citeseer, 2007) pp. 46–51.

[21] F.-J. Chang, J.-M. Liang, and Y.-C. Chen, Flood forecasting using radial basis function neural networks, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) **31**, 530 (2001).

[22] D. G. Krige, A statistical approach to some basic mine valuation problems on the witwatersrand, Journal of the Southern African Institute of Mining and Metallurgy **52**, 119 (1951).

[23] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, Design and analysis of computer experiments, Statistical science **4**, 409 (1989).

[24] J. P. Kleijnen, Kriging metamodeling in simulation: A review, European journal of operational research **192**, 707 (2009).

[25] C. Soize and R. Ghanem, Physical systems with random uncertainties: chaos representations with arbitrary probability measure, SIAM Journal on Scientific Computing **26**, 395 (2004).

[26] M. A. Bouhlel, N. Bartoli, A. Otsmane, and J. Morlier, Improving kriging surrogates of high-dimensional design models by partial least squares dimension reduction, Structural and Multidisciplinary Optimization **53**, 935 (2016).

[27] A. Attili, F. Bisetti, M. E. Mueller, and H. Pitsch, Formation, growth, and transport of soot in a three-dimensional turbulent non-premixed jet flame, Combustion and flame **161**, 1849 (2014).

[28] G. Bansal, A. Mascarenhas, and J. H. Chen, Direct numerical simulations of autoignition in stratified dimethyl-ether (dme)/air turbulent mixtures, Combustion and Flame **162**, 688 (2015).

[29] A. Bhagatwala, J. H. Chen, and T. Lu, Direct numerical simulations of hcci/saci with ethanol, Combustion and flame **161**, 1826 (2014).

[30] L. Hawchar, C.-P. El Soueidy, and F. Schoefs, Principal component analysis and polynomial chaos expansion for time-variant reliability problems, Reliability Engineering & System Safety **167**, 406 (2017).

[31] D. Bettebghor, N. Bartoli, S. Grihon, J. Morlier, and M. Samuelides, Surrogate modeling approximation using a mixture of experts based on em joint estimation, Structural and multidisciplinary optimization **43**, 243 (2011).

[32] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*, Vol. 2 (Springer, 2009).

[33] Y. Yang, Regression with multiple candidate models: selecting or mixing?, Statistica Sinica , 783 (2003).

[34] L. Lees, Laminar heat transfer over blunt-nosed bodies at hypersonic flight speeds, Journal of Jet Propulsion **26**, 259 (1956).

[35] J. Shlens, A tutorial on principal component analysis, arXiv preprint arXiv:1404.1100 (2014).

[36] P. Baldi and K. Hornik, Neural networks and principal component analysis: Learning from examples without local minima, Neural networks **2**, 53 (1989).

[37] J. A. Wegelin, A survey of partial least squares (pls) methods, with emphasis on the two-block case, (2000).

[38] M. E. Newman, Modularity and community structure in networks, Proceedings of the national academy of sciences **103**, 8577 (2006).

[39] G. Sun and S. Wang, A review of the artificial neural network surrogate modeling in aerodynamic design, Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering **233**, 5863 (2019).

[40] M. J. Powell, The theory of radial basis function approximation in 1990, Advances in numerical analysis , 105 (1992).

[41] S. N. Wood, Thin plate regression splines, Journal of the Royal Statistical Society: Series B (Statistical Methodology) **65**, 95 (2003).
[42] F. Schwenker, H. A. Kestler, and G. Palm, Three learning phases for radial-basis-function networks, Neural networks **14**, 439 (2001).
[43] L. Breiman, Random forests, Machine learning **45**, 5 (2001).
[44] E. Frank, M. Hall, L. Trigg, G. Holmes, and I. H. Witten, Data mining in bioinformatics using weka, Bioinformatics **20**, 2479 (2004).
[45] P. A. Gnoffo, R. N. Gupta, and J. L. Shinn, *NASA Technical Paper*, Tech. Rep. 2867 (NASA, 1989).
[46] E. Josyula and P. Vedula, Fundamental fluid transport equations for hypersonic nonequilibrium flows, in *Hypersonic Nonequilibrium Flows: Fundamentals and Recent Advances*, edited by E. Josyula (American Institute of Aeronautics and Astronautics (AIAA), 2015) pp. 1–43.
[47] J. B. Scoggins and T. E. Magin, Development of Mutation++: Multicomponent Thermodynamic and Transport Properties for Ionized Plasmas written in C++, in *11th AIAA/ASME Joint Thermophysics and Heat Transfer Conference* (American Institute of Aeronautics and Astronautics (AIAA), 2014) iD: AIAA 2014-2966.
[48] J. B. Scoggins, *Development of numerical methods and study of coupled flow, radiation, and ablation phenomena for atmospheric entry*, Ph.D. thesis, Université Paris-Saclay and von Karman Institute for Fluid Dynamics (2017).
[49] J. O. Hirschfelder, C. F. Curtiss, and R. B. Bird, Molecular theory of gases and liquids, Molecular theory of gases and liquids (1964).
[50] J. D. Ramshaw, Self-consistent effective binary diffusion in multicomponent gas mixtures, (1990).
[51] A. T. Margaritis, C. Scherding, O. Marxen, P. J. Schmid, and T. Sayadi, High-fidelity computational tool for chemically reacting hypersonic flow simulations, (In preparation).