Adaptive Depth Neural Networks for Scale-Bridging Modeling of Multiphase Reacting Flows

Michael E. Mueller
Department of Mechanical and Aerospace Engineering
Princeton University

Sankaran Sundaresan
Department of Chemical and Biological Engineering
Princeton University

Contributors:
G. D’Alessio, C.E. Lacey
Motivation

The application of ML algorithms can be particularly effective for reacting flows, because of their physical and chemical complexity.
Program Overview

• Overarching Problem
  – Development of a general framework for **scale-bridging** modeling between full-fidelity simulations and coarse-grained simulations
  – Emphasize **industrially-relevant** applications: multiphase chemically reacting flows (e.g., fluidized beds)

• Computational Investigation
  – Inaccessible with DNS: too many scales, too many species\(^1,2\)
  – Requires filtered TFM and an appropriate closure model

• Program Outcome
  – An automated workflow for data-based scale-bridging modeling between DEM and filtered TFM

---


Training Data

- Premixed hydrogen jet flame\(^3\)
  - R: Stoichiometric H\(_2\)/Air mixture diluted with N\(_2\) (20\% by volume)
  - P: Equilibrium products

- Re = 5000

- Multiple Ka cases:
  - K1: Ka = 3.7
  - K2: Ka = 54

- D = 1.08 mm (K1), 4.32 mm (K2)

- DNS data is filtered with a Gaussian filter of varying filter sizes
  - \(\Delta = 2, 4, 8, 16\)

In combustion, the progress variable $C$ is often defined to track the progress of reaction. As we are operating with a hydrogen flame, it is set equal to:

$$C = Y_{H_2O}$$

For coarse-grain modeling, an unclosed quantity that is essential in CFD simulations is the filtered dissipation, which is defined as:

$$\tilde{\chi}_{CC} = 2D_c \frac{\partial C}{\partial x_j} \frac{\partial C}{\partial x_i}$$

Pierce model\(^4\): (physical model) $$\tilde{\chi}_{CC} \approx 2(\tilde{D}_C + D_{SFS}) \frac{\partial \tilde{C}}{\partial x_j} \frac{\partial \tilde{C}}{\partial x_j}$$

Artificial Neural Networks (ANNs) are non-linear universal approximators, consisting of a structure of interconnected layers of neurons.
Numerical modeling for CFD

Several questions are still open:

1) Can we pre-process the data to increase the accuracy of the mathematical model and also to accelerate the training time?

2) Can we automate the selection of the inputs, without prior knowledge of the physics?

3) How can we automatically select the best architecture and hyperparameters for an ANN? Can we enforce physical constraint?
Proposed pipeline

Input data

- Outlier removal
- Data sampling

Data pre-processing

- Principal Component Analysis
- Pi groups analysis

Dimensionality reduction

- Artificial Neural Networks

Numerical Modeling

Variable(s) of interest
Proposed pipeline

Can we pre-process the data to increase the accuracy of the mathematical model and also to accelerate the training time?

Input data → Outlier removal and Data sampling → Principal Component Analysis and Pi groups analysis → Artificial Neural Networks → Variable(s) of interest

Data pre-processing → Dimensionality reduction → Numerical Modeling
Having an input matrix accounting for a large number of statistical observations could be an issue in terms of computational cost, memory requirements and tendency to overfit.

Thus, for many practical applications it is a good strategy to sample the original matrix.

\[ X = \begin{bmatrix} \alpha_1 & \beta_1 & \ldots & \omega_1 \\ \alpha_2 & \beta_2 & \ldots & \omega_2 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_n & \beta_n & \ldots & \omega_n \end{bmatrix} \]

It is possible to condition the input using one variable: \( \alpha \)

Variable used for the conditioning
Data sampling

All the rows of the input matrix $X$ where alpha is bounded between a given interval are grouped together.

Group 1:

$G_1 = \bigcup_{i=1}^{m} x_i \in X \mid \alpha_i \in [b, c]$  

A certain number of observations is then retained (randomly) from each group, $G_j$.
Data sampling

**Full data size:** 4,000,000 statistical observations

**Sampled data size:** 750,000 statistical observations
Can we automate the selection of the inputs, without prior knowledge of the physics?

Proposed pipeline:

- **Input data**
  - Outlier removal
  - Data sampling

- **Data pre-processing**
  - Dimensionality reduction

- **Principal Component Analysis**
  - Pi groups analysis

- **Artificial Neural Networks**
  - Variable(s) of interest

- **Numerical Modeling**
Before feeding the input matrix to any kind of statistical algorithm, it is necessary to pre-process the columns so that all the variables’ units are coherent.

This operation is usually accomplished by scaling each variable with a statistical quantity, e.g., with its **standard deviation**.
Physics-Based Dimensionality Reduction

From the Buckingham Pi Theorem, it is known that $k$ fundamental dimensions can be found: the input matrix can thus be expressed by using only $(p-k)$ *dimensionless* groups.

$$\pi_j = \prod_{i=1}^{p} x_i^{\alpha_{ij}}$$

$j$-th dimensionless group

$i$-th feature

Exponent of the $i$-th feature in the $j$-th dimensionless group

Alternatively, it is possible to find $(p-k)+1$ *dimensionally consistent* groups by forming an extra group accounting the $k$ fundamental dimensions.

Considering the Pi groups, we can get dimensionally consistent input groups that match the dimension of the output. At the same time, we also reduce the number of input variables.
Physics-Based Dimensionality Reduction

**Fundamental dimensions:**
Mass of product, Mass of mixture, time, distance.

\[
\tilde{\chi}_{CC} = 2D_c \frac{\partial C}{\partial x_j} \frac{\partial C}{\partial x_i} \\
\tilde{\chi}_{CC} [\text{=}] \frac{kg_C^2}{kg_m^2 s}
\]

<table>
<thead>
<tr>
<th></th>
<th>Inputs</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Original Inputs</strong></td>
<td>( \bar{c}, c_v, \left</td>
<td>\frac{\partial \bar{c}}{\partial x_i} \right</td>
</tr>
<tr>
<td></td>
<td>(</td>
<td>\bar{S}_{ij}</td>
</tr>
<tr>
<td><strong>Dimensionally consistent inputs</strong></td>
<td>( \bar{c}^2 \bar{D}_c, \frac{c_v \bar{D}_c}{\Delta L^2}, \left</td>
<td>\frac{\partial \bar{c}}{\partial x_i} \right</td>
</tr>
</tbody>
</table>
Physics-Based Dimensionality Reduction

- **8 Dimensional Features**
  - Network Architecture: 33 Layers, 36 Neurons
  - Mean Absolute Error: 0.208
  - Training Time: 3 hours

- **5 Dimensionally Consistent Features**
  - Network Architecture: 13 Layers, 34 Neurons
  - Mean Absolute Error: 0.078
  - Training Time: 1.5 hours

- With fewer features: **smaller network, lower testing error, lower training time**

- For large $p$ and small $k$, $(p-k)+1$ is still large, so may need to further reduce dimensionality of the feature space
Dimensionality reduction via PCA

Original input matrix

\[ X = \begin{bmatrix} \alpha_1 & \beta_1 & \ldots & \omega_1 \\ \alpha_2 & \beta_2 & \ldots & \omega_2 \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_n & \beta_n & \ldots & \omega_n \end{bmatrix} \]

Input matrix: ‘n’ statistical observations of ‘p’ variables

Covariance matrix computation and decomposition

\[ C = \frac{1}{n-1} X^T X \]

\[ C = A \Lambda A^T \]

**Principal Components**: eigenvectors obtained from the decomposition of the matrix \( C \)

**Eigenvalues**: portion of information (original data variance) accounted by each PC

Original data

Principal Components extraction

Size reduction
Dimensionality reduction via PCA

The application of the PCA algorithm is appropriate for this framework as it is possible to *automatically* determine the optimal dimensionality of the reduced manifold by examining the eigenvalues’ magnitude distribution\(^5,^6\)

\[
L = \begin{bmatrix}
\lambda_1 & 0 & \cdots & 0 \\
0 & \lambda_2 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & \lambda_p
\end{bmatrix}
\]

\[
L \mid \lambda_1 > \lambda_2 > \cdots > \lambda_p
\]

\[
t_q = \frac{\sum_{j=1}^{q} \lambda_j}{\sum_{i=1}^{p} \lambda_i}
\]

In this case, from the analysis of the eigenvalues’ distribution, a 5-dimensional manifold is enough to explain almost the 100% of the original data variance. **This is the same dimensionality obtained from the Pi-group analysis** (dimensionally-consistent inputs).


Proposed pipeline

How can we automatically select the best architecture and hyperparameters for an ANN? Can we enforce physical constraint?

- Input data
- Outlier removal and Data sampling
- Principal Component Analysis and Pi groups analysis
- Artificial Neural Networks
- Variable(s) of interest

Data pre-processing
Dimensionality reduction
Numerical Modeling
An iterative algorithm is now used to select the best architecture in an unsupervised fashion.

\[ N = \left( \frac{n_{\text{neurons\_start}}}{n_{\text{layers\_start}}} \right) \]

- Train Model
- Evaluate Model Error
- Extrapolate Network Architecture of Next Iteration
- Update Error Fit Function
- Error < Threshold
- DNN

\[ \text{Error}(N) = aN^b + c \]
1) Finding a way to enforce physical constraint into the mathematical model, to increase the regression accuracy

Many physical variables are strictly positive or non-negative, by definition (e.g., chemical species mass fractions). Ideally, it is possible to force the network to predict a positive variable by considering a log space. **Realistically, it might be complicated to find a solution to automate the enforcing.**

**Case 1 - Prediction in physical space**

In this case, the prediction of the variable of interest can be positive, negative, or zero.

**Case 2 - Prediction in log space**

In this case, we are forcing the prediction of the variable of interest \( y \) to be strictly positive.
2) Integrate *on-the-fly* learning curves diagnostics to avoid overfitting.

By means of **early stopping**, it is possible to interrupt the ANN training before the network starts overfitting.
3) Extend the framework to multiphase chemically reacting flows

The main objective is to train the framework on massive CFD-DEM dataset or Two-Fluid Model (TFM) numerical simulations, to then automatically get the reduced model (i.e., a closure model) for the corresponding Filtered-TFM.

Moreover, we are also interested in performing data analysis on the CFD-DEM or TFM training set to discover how to optimally select the best input variables, in an unsupervised fashion.

---

Project Schedule

• **Year One**
  – Identification of target applications and closure models
  – Framework development and *a priori* testing

• **Year Two**
  – *A posteriori* validation of the proposed framework using selected target applications and closure models
  – Scale-bridging modeling implementation
    – The optimized framework for the automated training will be included into Nodeworks workflow environment software.
    – The software will be tested for data-based scale-bridging modeling between DEM and filtered TFM

*Close collaboration with NETL researchers will be required for these tasks*
Thank you for your attention