Proceedings of the 12th International Conference on Fluid Flow, Heat and Mass Transfer (FFHMT 2025)
July 15, 2025 - July 17, 2025 | Imperial College London Conference, London, United Kingdom
Paper No. 170
DOI: 10.11159/ffhmt25.170

Application of Singular Value Decomposition and Autoencoder for Supersonic Flow over Backward Facing Step

Shivam Sanjay Singh[1], Rudra N. Roy[1]
[1]School of Mechanical Sciences, Indian Institute of Technology, Goa
Farmagudi, Ponda-403401, Goa, India
singh.sanjay.21063@iitgoa.ac.in; rudra@iitgoa.ac.in

Abstract **-** This paper explores the integration of *PythonFOAM* with the hybrid pressure-based solver *rhoPimpleCentralFoam* to simulate supersonic flow over a backward facing step, aiming to validate the coupling of these solvers. Flow field simulation was performed using Reynolds averaged Navier Stokes **(**RANS)-based turbulence model with the hybrid solver. Streaming singular value decomposition (SVD) was applied to identify coherent flow structures, capturing essential features such as boundary layer separation and shock wave formation. The SVD modes were then utilized to reconstruct the velocity field, with the mean flow field obtained showed a close match to the original computational fluid dynamics (CFD) results, highlighting the effectiveness of this approach. Furthermore, a deep neural network autoencoder was applied to compress the flow field data, further demonstrating the integration of *PythonFOAM* with the solver. The autoencoder learns a compact representation of the velocity field, and the reconstructed field from this compressed representation aligns closely with the CFD results, confirming the model's ability to approximate complex flow dynamics. The results obtained demonstrated the successful coupling of the solvers and underscored the potential of reduced-order modelling techniques for solving complex flow problems.

Keywords**:** Supersonic flow, Singular value decomposition, Autoencoder, OpenFOAM

1. Introduction

Designing high-speed airbreathing engines involves understanding several complex phenomena of supersonic combustion, such as turbulent mixing, shock interaction and heat release rate. The supersonic flow past a backwards-facing step (BFS) is a classical problem that mimics several behaviours that appear in the supersonic air-breathing engine. A thorough understanding of the complex flow physics encountered in supersonic BFS can benefit many design and development processes in these fields.

With the advancement of computational capabilities, large scale simulations have made significant strides in enhancing our understanding of the complex physics of supersonic flows [1-3]. Recently, reduced-order modelling techniques have gained attention because of their ability to capture essential physics and dynamics of high-fidelity models at a fraction of the computational cost. These reduced-order models often involve data-driven approaches such as proper orthogonal decomposition (POD) [4], dynamic mode decomposition (DMD) [5], etc. In fluid dynamics, the POD method has been extensively used to identify coherent structures and perform reduced-order modelling. POD is based on singular value decomposition (SVD) and represents the original system in a reduced form by truncating the modes. Further, Galerkin projection can be applied to the governing equations in a high-fidelity model to obtain a reduced system of equations. Moreover, the use of non-intrusive reduced order modelling, such as neural networks like autoencoder for dimensionality reduction, has been demonstrated in several studies [6, 7].

When performing forming reduced-order modelling, users often face challenges exporting the data pertaining to large eddy simulation (LES)/Reynold's averaged Navier Stokes (RANS). In this context, in-situ methods offer advantages over traditional data processing techniques. Recently, Malik et al. [8] developed a framework to interface OpenFOAM and Python for performing in-situ data analysis. OpenFOAM is a well-established open-source finite volume solver for computational fluid dynamics (CFD) applications. It offers built-in structured and unstructured meshing facilities and MPI-based utilities for solving problems in an efficient manner. OpenFOAM has been used to study a wide range of problems in the areas of multi-phase heat transfer [9], rarefied gas dynamics [10], supersonic flows [11], combustion [12, 13], etc. The development work by Malik et al. [8] demonstrated the integration of data-science capabilities using Python/C++ API with OpenFOAM.

The interfacing of OpenFOAM with Python not only leverages the in-situ data analysis but also opens the avenues for the application of machine learning (ML) in CFD applications using OpenFOAM and would be useful for the implementation of several reduced-order modelling techniques.

The present work aims to integrate the *PythonFOAM* framework developed by Malik et al. [8] with the hydrid pressure-based solver, i.e., *rhoPimpleCentralFoam*-solver [14], applicable for supersonic flows. The *rhoPimpleCentralFoam*-solver has been extensively validated previously by Janhavi and Roy [11] for supersonic flow over backward facing step. The interfaced solver, named *Python-rhoPimpleCentralFoam* will initially be used to perform SVD on supersonic flow over backward facing step. Further, a deep neural network autoencoder will be used for compressing the flow-field information mainly to demonstrate the potential use of machine learning in *Python-rhoPimpleCentralFoam* solver.

2. Numerical Details

In this section, details about the governing equations, singular value decomposition and autoencoder are provided.

2.1 Governing equations

The governing equations for continuity, momentum, and energy transport are given as [15],

Continuity equation
$$\frac{\partial \bar{\rho}}{\partial t} + \frac{\partial \bar{\rho}\tilde{u}_i}{\partial x_i} = 0 \tag{1}$$

Momentum equation
$$\frac{\partial (\bar{\rho}\tilde{u}_i)}{\partial t} + \frac{\partial}{\partial x_j}\left[\bar{\rho}\tilde{u}_i\tilde{u}_j + \bar{p}\delta_{ij} + \overline{\rho u_i'' u_j''} - \bar{\tau}_{ij} \right] = 0 \tag{2}$$

Energy equation
$$\frac{\partial (\bar{\rho}\tilde{E})}{\partial t} + \frac{\partial}{\partial x_j}\left[\bar{\rho}\tilde{u}_j\tilde{E} + \tilde{u}_j\bar{p} + \overline{u_j'' p} + \overline{\rho u_i'' E''} + \bar{q}_j - \overline{u_i \tau_{ij}} \right] = 0 \tag{3}$$

In Eqs. (1-3) and tilde refers to density-weighted averaging and overbar indicates averaging through Reynolds decomposition, t is the time, x is the special coordinate, ρ is the density, u is the velocity vector, p is the pressure, δ_{ij} is the Kronecker delta, q is the heat flux, τ_{ij} is the shear stress and E is total energy. In the present study, k-ε [16] based RANS turbulence model was used to obtain closure for the term $-\rho\overline{u_i' u_j'}$ appearing in the momentum equation.

2.2 Singular Value Decomposition

The singular value decomposition (SVD) is a matrix factorization technique that decomposes a real $M \times N$ matrix as

$$A = U \sum V^T \tag{4}$$

where $A \in \mathbb{R}^{M \times N}$, $U \in \mathbb{R}^{M \times N}$, $\sum \in \mathbb{R}^{N \times N}$, and $V \in \mathbb{R}^{N \times N}$, $\mathbb{R}$ represents the real space, M and N are a number of snapshots of data collected for the analysis, and the number of degrees of freedom in each snapshot, respectively. In Eq. (4), U are the left singular vectors, $\sum$ are the diagonal entries or Eigen values and V are the right singular vectors. The superscript ^{T} represents matrix transpose. The computation requirement in terms of scales and memory for the SVD is O($M N^2$) and O($M N$), respectively. For typical computational fluid dynamics (CFD) applications, the analysis using classical SVD would be challenging as the degree of freedom may grow very large. To circumvent this difficulty Levy and Lindenbaum [17] proposed a streaming variant of SVD. The streaming SVD is been performed by extracting the first K left singular vectors, which correspond to the K largest coherent structures which results in the reduction of the operational cost and memory footprint of SVD to O($M N K$) and O($M K$), respectively. Here the left singular eigenvectors is updated in a batch-like manner. The algorithm proposed and developed by Levy and Lindenbaum [17] has been used in the present work to perform streaming SVD.

2.3 Autoencoder

An autoencoder belongs to the class of deep neural networks, widely used for performing various reduced-order modelling for flow physics. Figure 1 shows the schematic of the representation of the autoencoder architecture. It consists of two main parts: an encoder, which compresses the input data into a lower-dimensional latent space using a number of

hidden layers, and a decoder, which reconstructs the original data from the latent space using the appropriate number of hidden layers. The network is trained to reduce the reconstruction error through a loss function. The latent space captures the most salient features of the data compressed into some latent variables containing arbitrary data, enabling dimensionality reduction and noise filtering. In this study, an autoencoder was utilized to analyse snapshots of supersonic flow over a backward facing step. The latent space dimensionality was set to 4, aiming to extract dominant flow features analogous to proper orthogonal decomposition (POD) modes. The network architecture is designed with the encoder progressively reducing dimensionality through dense layers with decreasing neuron counts (50, 25, 10, and 4), while the decoder mirrors this structure, reconstructing the flow field from the latent space. In each layer, a Swish activation function was employed, and an ADAM optimizer was used with a learning rate of 0.001.

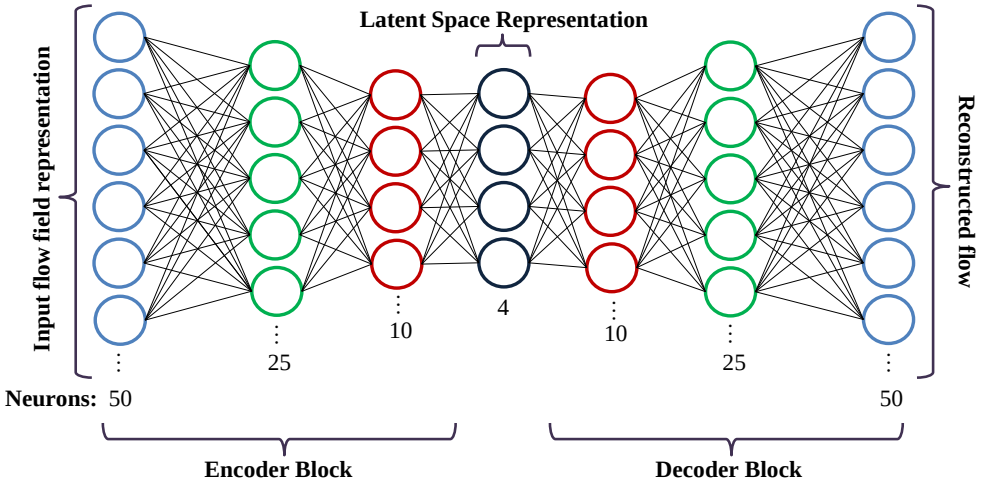

Fig. 1. Schematic of autoencoder architecture

3. Workflow and Computational Details

The overall workflow of the Python and *rhoPimpleCentralFoam*-solver is summerized here. The hybrid pressure-based solver, i.e., *rhoPimpleCentralFoam* used in this study combines PIMPLE algorithm with the KT scheme [14]. This algorithm relates the change in the pressure field with a change in the velocity and density fields using discretized pressure equation [14, 18]. A Python/C-API has been utilized to embed Python in OpenFOAM. The *rhoPimpleCentralFoam*-solver was modified to include a Python interpreter that remained live throughout the simulation. Once the framework for interfacing OpenFOAM and Python was set, *the python_module.py* file was created in the working directory. The python_module.py file comprises Python module functions such as SVD and autoencoder. Figure 2 shows the flow chart of the coupled Python and *rhoPimpleCentralFoam* solver. At the initial stage, the hybrid solver and Python/C-API coupling were initialized. Further, the Pimple algorithm was executed, and at each time step, data obtained from the *rhoPimpleCentralFoam*-solver were fed to the NumPy array. Further, the data were retrieved from the NumPy array for performing SVD/autoencoder calculations. Finally, the processed data were fed back to the flow solver for writing the data in OpenFOAM I/O format.

The validated 2D computational setup of backward facing setup by Janhavi and Roy [11] was used to perform SVD and autoencoder calculations in this study. The free stream velocity is 520 m/s, Reynolds number is 1.024×10^5 and Mach number is 2. The static temperature and pressure are 167 K and 35 kPa, respectively [19]. The computational geometry comprises top and bottom walls where no slip and zero pressure gradient boundary conditions were applied [11]. Uniform velocity boundary conditions were used at the inlet, whereas zero gradient boundary conditions were provided at the outlet boundary. The turbulence fields were modelled using SST k-ω models in conjunction with *rhoPimpleCentralFoam*-solver

in OpenFOAM. The number of grids selected was 180 and 150 in the axial direction and radial direction, respectively based on the grid independence study [11]. The results obtained are discussed in the subsequent section.

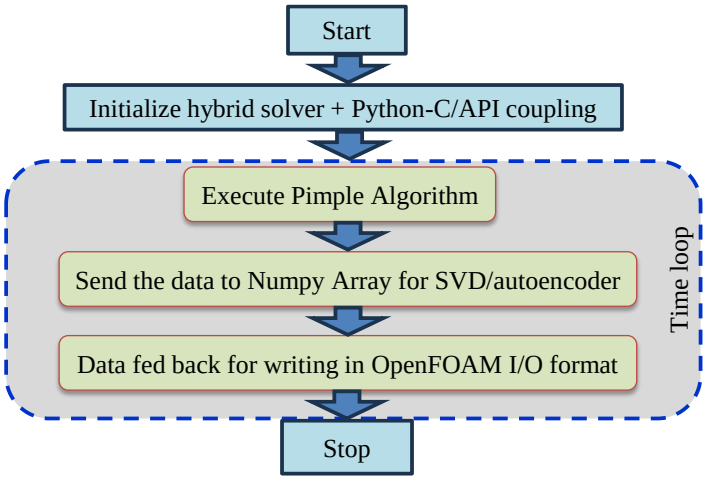

Fig. 2. Flow chart of the coupled Python and *rhoPimpleCentralFoam* solver

3. Results and Discussion

The results obtained from SVD and autoencoder for the supersonic flow over a backward facing step using the coupled solver are highlighted in this section.

Figures 3 and 4 illustrate the four singular vectors corresponding to four different flow modes for the *u*-component and *v*-component velocity, respectively which were obtained using the streaming SVD at the final time step. The first singular vector in Fig. 3 represents Mode 1 of the *u*-component, which accounts for 96.887% of the total energy. This dominant mode primarily captures the mean flow behaviour, reflecting the overall characteristics of the flow field. Specifically, it highlights key flow features, such as boundary layer separation, shock wave formation, and the general flow patterns in the system. Additionally, Mode 1 effectively represents the formation of the expansion and reattachment waves, as well as their interaction with the step and the downstream regions of the flow. The first singular vector of the *v*-component, i.e. mode 1 shown in Fig. 4 captures 83.736 % of energy and reveals a well-defined expansion region downstream of the step. Mode 2 of *u*-component and *v*-component velocity captures 1.749 % and 9.135 % of energy, respectively. Here singular vectors show high values near the step edge, indicating that these modes are related to localized flow features near the step. The structure obtained in mode 3 for the *u*-component is quite similar to the mode 3 for the *v*-component. Additionally, mode 4 exhibits similar structures as observed in modes 1 and 2 for both the velocity components. However, the energy associated with modes 3 and 4 is significantly lower compared to mode 1. Figure 5 provides a comparison between the velocity field obtained from the hybrid solver and the velocity field reconstructed from the SVD using the four modes. The reconstructed velocity field, while derived from only four modes, provides a remarkably accurate representation of the major flow features. This includes important flow phenomena such as the recirculation zone, shock wave, and expansion fan. The reconstructed velocity field shows a high level of agreement with the results obtained from the CFD simulations, demonstrating the effectiveness of using these modes to capture the essential features of the flow field, even when fewer modes are used.

Further, a deep neural network autoencoder was used to compress the flow-field information mainly to demonstrate the coupling of Python-rhoPimpleCentralFoam-solver. In this study with a batch size of 128 snapshot of *u*-component were collected and were used to train the deep neural network. Further the trained network weights were used to reconstruct the *u*-component velocity at the end of 1000 iterations. Figure 6 presents a comparison between the *u*-component velocity field obtained from the hybrid solver and reconstructed *u*-component velocity produced by the autoencoder. It may be observed that the reconstructed velocity field closely resembles the actual solution obtained from the CFD simulation. This indicates

that the autoencoder, by learning the nonlinear relationships between the low-dimensional embeddings and the original high-dimensional velocity field, can effectively approximate the flow field.

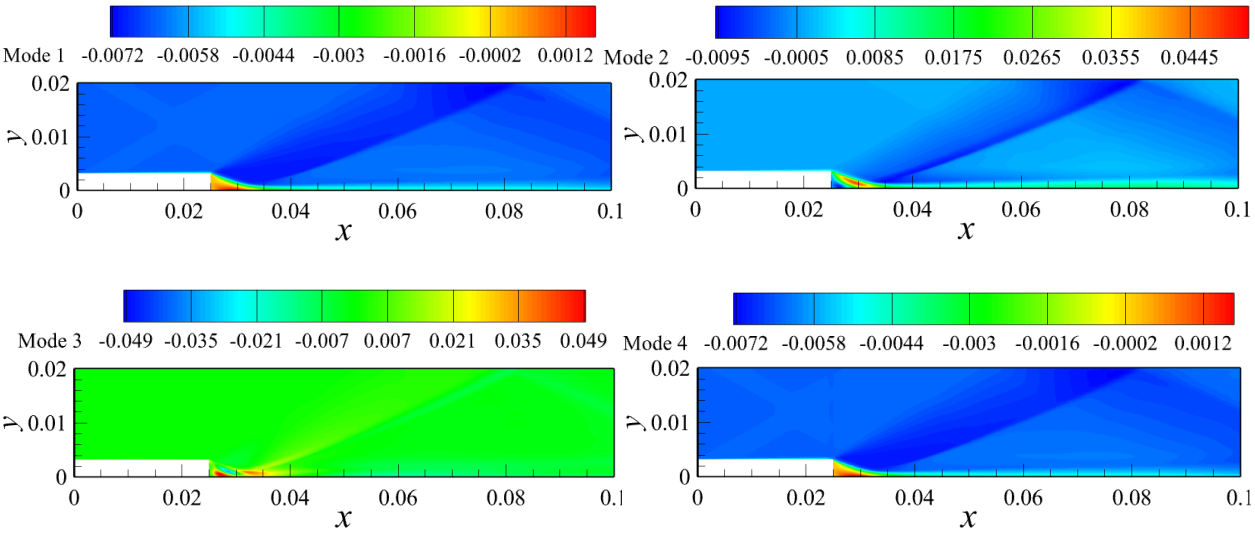

Fig. 3. Four singular vectors for *u*-component at the final time step

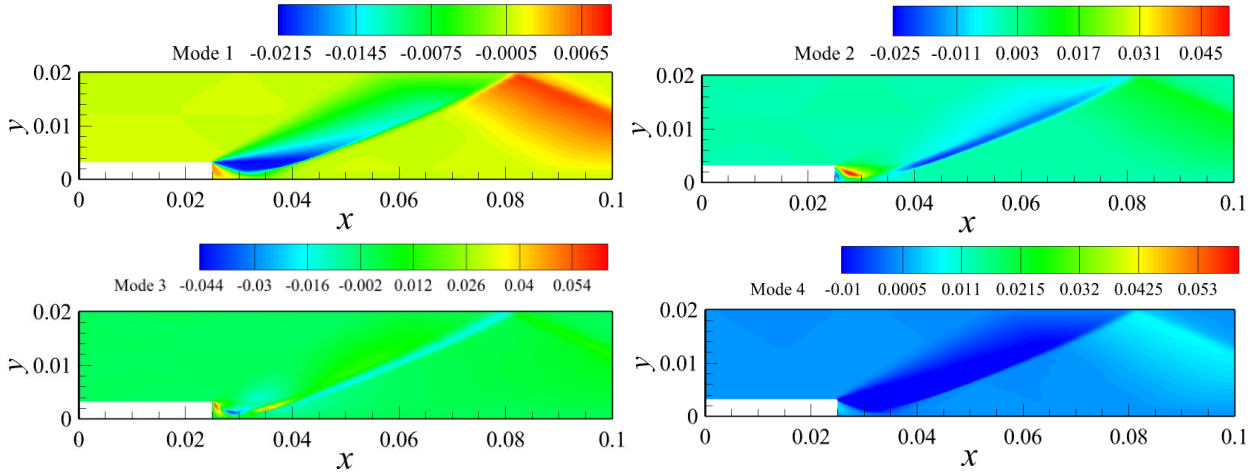

Fig. 4. Four singular vectors for *v*-component at the final time step

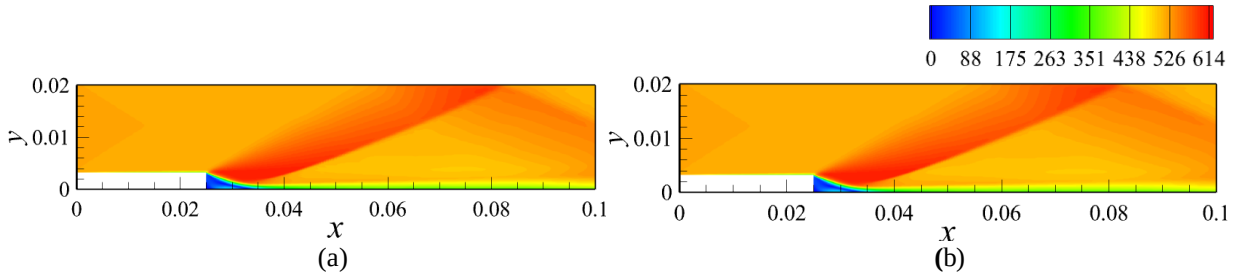

Fig. 5. Velocity magnitude obtained from (a) hybrid solver and (b) reconstructed-SVD

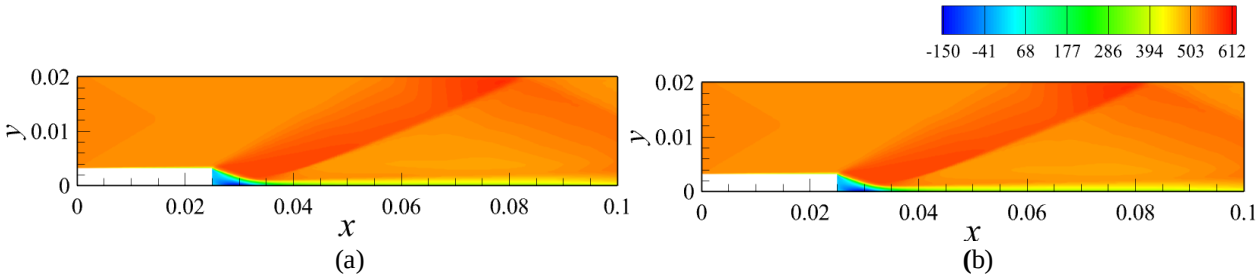

Fig. 6. *u*-component velocity obtained from (a) hybrid solver and (b) reconstructed-autoencoder

4. Conclusion

This article discusses the integration of *PythonFOAM* with the hybrid pressure-based solver, i.e., *rhoPimpleCentralFoam*, focusing on its application to the supersonic flow over a backward facing step case. The primary aim of this study was to validate the coupling of these two solvers through the use of streaming SVD and a deep neural network autoencoder. The flow field calculations were initially performed using RANS-based turbulence model with the hybrid pressure-based solver. To begin with, streaming SVD was applied to extract the coherent structures present in the flow field. The four modes derived from the SVD analysis revealed crucial flow features such as boundary layer separation, shock wave formation, and the overall flow patterns within the system. By utilizing these derived modes, the velocity field was then reconstructed, and the mean flow field obtained closely resembled the original solution, demonstrating the effectiveness of this technique in capturing the essential characteristics of the flow. Additionally, a deep neural network autoencoder was employed to compress the flow-field information. This step was mainly intended to demonstrate the successful coupling of *PythonFOAM* with the *rhoPimpleCentralFoam*-solver. The autoencoder learned a low-dimensional representation of the velocity field, and upon decoding, the reconstructed velocity field was found to closely match the actual solution obtained from the CFD simulation, further validating the approach. This study highlights the successful coupling of these two solvers and establishes a solid foundation for the development of reduced-order modelling techniques. These techniques have significant potential in addressing and solving complex flow problems by reducing the computational cost and enhancing the efficiency of simulations.

References

[1] W. Hu, S. Hickel and B. V. Oudheusden, "Dynamics of a supersonic transitional flow over a backward-facing step," *Phys. Rev. Fluids*, vol. 4, pp. 103904, 2019.

[2] W. Li and H. Liu, "Large-eddy simulation of shock-wave/boundary-layer interaction control using a backward facing step," *Aerosp. Sci. Techno.*, vol. 84, pp. 1011-1019, 2019.

[3] R. K. Soni, N. Arya and A. De, "Characterization of turbulent supersonic flow over a backward-facing step," *AIAA J.*, vol. 55, pp. 1511-1529, 2017.

[4] A. Chatterjee, "An introduction to the proper orthogonal decomposition," *Curr. Sci.*, vol. 78, pp. 808-817, 2000.

[5] P. J. Schmid, "Dynamic mode decomposition of numerical and experimental data," *J. Fluid Mech.*, vol. 656, pp. 5-28, 2010.

[6] S. Wiewel, M. Becher, N. Thuerey, "Latent space physics: towards learning the temporal evolution of fluid flow," *Comput. Graph Forum* vol. 38, pp. 71-82, 2019.

[7] K. Lee and K. T. Carlberg, "Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders," *J. Comput. Phys.*, vol. 404, pp. 108973, 2020.

[8] R. Maulik, D. K. Fytanidis, B. Lusch, V. Vishwanath and S. Patel, "PythonFOAM: In-situ data analyses with OpenFOAM and Python," *J. Comput. Sci.*, vol. 62, pp. 101750, 2022.

[9] C. Kunkelmann and P. Stephan, "CFD simulation of boiling flows using the volume-of-fluid method within OpenFOAM," *Numer. Heat Transf. A.*, vol. 56, pp. 631–646, 2009.
[10] C. White, M. K. Borg, T. J. Scanlon, S. M. Longshaw, B. John, D. R. Emerson and J. M. Reese, "dsmcFoam+: An OpenFOAM based direct simulation Monte Carlo solver," *Comput. Phys. Commun.*, vol. 224, pp. 22-43, 2018.
[11] J. Gharate and R. N. Roy, "Modelling of supersonic and subsonic flows using hybrid pressure based solver in Openfoam," *Proceedings of the 8th International Conference of Fluid Flow, Heat and Mass Transfer (FFHMT'21)*, 2021, Paper No. 107.
[12] P. P. Singh and R. N. Roy, "Evaluation of flow and flame characteristics of turbulent bluff-body CH_4-H_2 flame using LES-FPV approach," *J. Turbul.*, vol. 25, pp. 2352490, 2024.
[13] P. P. Singh and R. N. Roy, "Nonpremixed and premixed FPV modeling of a turbulent CH_4-H_2 bluff-body flame," *Combust. Sci. Technol.*, pp. 1-27, 2024.
[14] M. Kraposhin, M. Banholzer, M. Pfitzner and I. K. Marchevsky, "A hybrid pressure-based solver for nonideal single-phase fluid flows at all speeds*," Int. J. Num. Methods Fluids*, vol. 88, pp. 79–99, 2018.
[15] H. K. Versteeg and W. Malalasekera, *Introduction to computational fluid dynamics*. Pearson Education Limited: 2nd Ed., 2007.
[16] B. E. Launder and D. B. Spalding, "The numerical computation of turbulent flows," *Comp. Methods Appl. Mecha. Eng.*, vol. 3, pp. 269–289, 1974.
[17] A. Levy and M. Lindenbaum, "Sequential Karhunen-Loeve basis extraction and its application to images," *Proceedings 1998 International Conference on Image Processing (ICIP98)*, 1998, vol. 2, pp. 456460.
[18] M. Kraposhin, A. Bovtrikova and S. Strijhak, "Adaptation of Kurganov-Tadmor numerical scheme for applying in combination with the PISO method in numerical simulation of flows in a wide range of Mach numbers," *Procedia Comp. Sci.*, vol. 66, pp. 43-52, 2015.
[19] J. C. McDaniel, D. G. Fletcher and R. J. Hartfield, "Staged transverse injection into Mach 2 flow behind a rearward-facingstep: a 3D compressible flow test case for hypersonic combustor CFD validation," *AIAA Paper*, 1992.