The 8th European Congress on Computational Methods in Applied Sciences and Engineering
ECCOMAS Congress 2022
5-–9 June 2022, Oslo, Norway

EXPLORATIVE IN-SITU ANALYSIS OF TURBULENT FLOW DATA BASED ON A DATA-DRIVEN APPROACH ECCOMAS CONGRESS 2022

Christian Gscheidle [1] **and Jochen Garcke**[1,2]

[1] Fraunhofer Center for Machine Learning and SCAI, Sankt Augustin, Germany,
christian.gscheidle@scai.fraunhofer.de, jochen.garcke@scai.fraunhofer.de

[2] Institut für Numerische Simulation, Universität Bonn, Germany

Key words: Turbulence simulation, Explorative analysis, Modal Decomposition, In-situ algorithms

Abstract. The Proper Orthogonal Decomposition (POD) has been used for several years in the post-processing of highly-resolved Computational Fluid Dynamics (CFD) simulations. While the POD can provide valuable insights into the spatial-temporal behaviour of single transient flows, it can be challenging to evaluate and compare results when applied to multiple simulations. Therefore, we propose a workflow based on data-driven techniques, namely dimensionality reduction and clustering to extract knowledge from large simulation bundles from transient CFD simulations. We apply this workflow to investigate the flow around two cylinders that contain complex modal structures in the wake region. A special emphasis lies on the formulation of in-situ algorithms to compute the data-driven representations during run-time of the simulation. This can reduce the amount of data in- and output and enables a simulation monitoring to reduce computational efforts. Finally, a classifier is trained to predict characteristic physical behaviour in the flow only based on the input parameters.

1 INTRODUCTION

In the last decades, highly-resolved numerical flow simulations, e.g. Large Eddy Simulations (LES) and Direct Numerical Simulations (DNS), have become an invaluable engineering tool to get insights into the physics of turbulent flows. However, the analysis of large simulation bundles, e.g. from variations in the geometry, boundary conditions or material properties, can be a quite challenging task. A post-processing purely based on analytical scalar or integral quantities can capture only specific aspects of the solution, while multi-scale simulations usually contain much more information that can be exploited. Data-driven techniques, such as the Proper Orthogonal Decomposition (POD) promise a more global perspective on the data and delivers valuable insight into the spatio-temporal behaviour of transient flows.

Up to now, the comparison of these modal structures can not easily be automated and is usually done in a visual analysis, which can be difficult if the number of simulations increase. Therefore, we propose a data-driven workflow based on dimensionality reduction and clustering techniques that has proven successful in the past to identify similarities and patterns in the high-dimensional simulation results [1], [2]. Furthermore, in conjunction with in-situ analysis

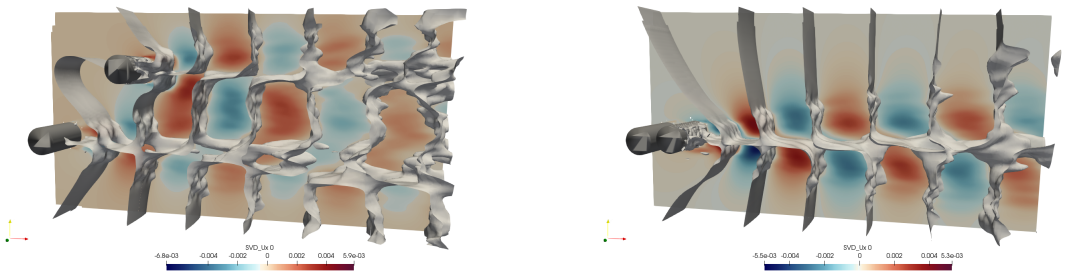

Figure 1: Iso-contours of the three dimensional modal structures computed for two different cylinder configurations; in the background, slice parallel to the flow direction showing the modal structures in the midplane

approaches, these techniques can also be employed to monitor the simulation [3]. That way, trends in the flow solution over time, bifurcating behaviour, or characteristic phenomenons, such flow separation, can be identified and tracked.

We will apply linear and non-linear techniques of dimensionality reduction, namely the Proper Orthogonal Decomposition and Diffusion Maps, with the goal to derive suitable low-dimensional representations of the chaotic time-series data from turbulent flow simulations. We will apply these techniques to investigate the turbulent flow around two cylinder that contains a variety of interesting spatial and temporal modes. Besides, we focus on an in-situ algorithm for the POD to compute the representations during run-time to save data I/O and increase the resolution of results. Finally, we perform a clustering of these modes to identify characteristic groups of flow behaviour in the high-dimensional simulations data and train a classifier to predict these cluster based on the input data.

2 EXPLORATIVE ANALYSIS OF FLOW DATA

For the explorative analysis of CFD simulation bundles, we suggest the workflow below that is represented in figure 2 and discussed in detail in the following sections:

1. Set up CFD parameter study with varying geometrical boundary conditions
2. Extract modal features from the transient flow data during runtime
3. Collect dominant modes and apply diffusion maps for a non-linear dimensionality reduction
4. Find groups of similar simulations by applying a k-means clustering
5. Use classifier to identify decision regions and make prediction for new input parameters

2.1 In-Situ Proper Orthogonal Decomposition

As a first preprocessing step we want to extract spatial - temporal features from the transient flow field and reduce the formal dimension of the data. A suitable technique for this purpose is the the Proper Orthogonal Decomposition (POD), which has been used for several decades to study turbulent flows by providing a decomposition of a flow field into spatial modes and

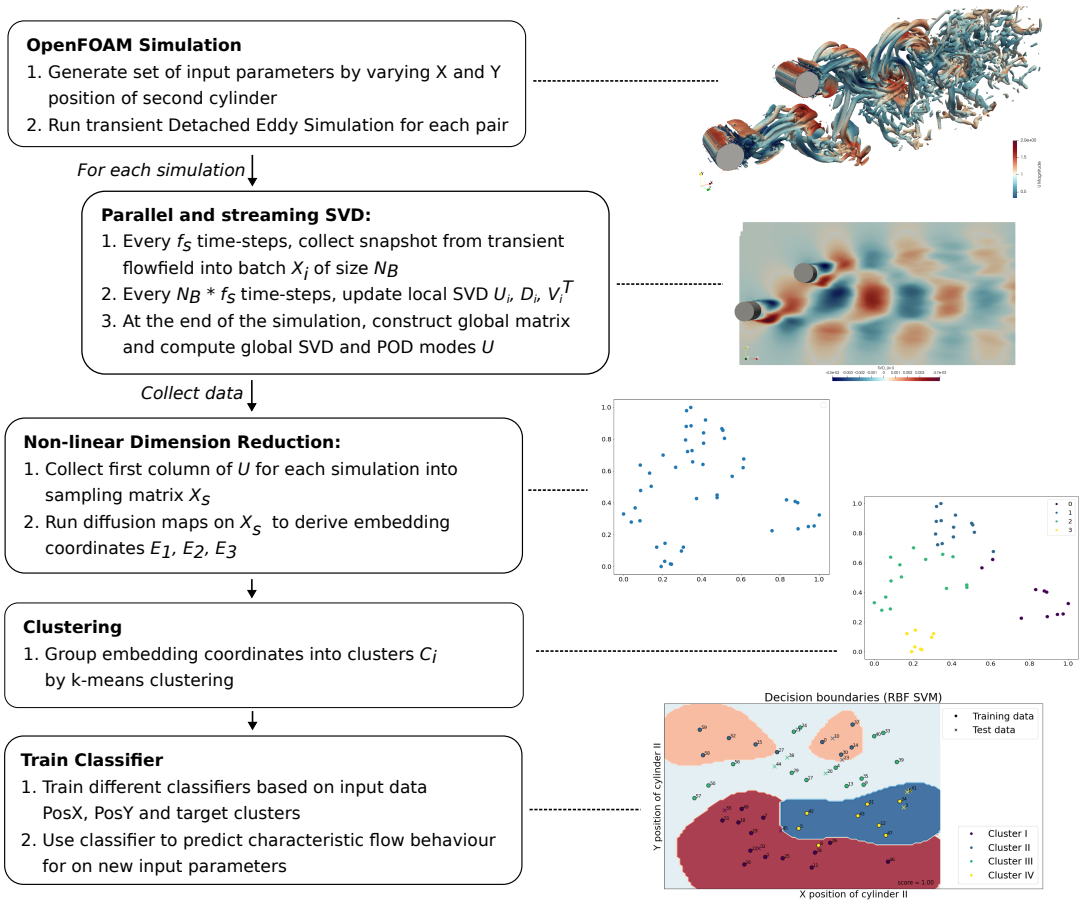

Figure 2: Proposed workflow layout for an explorative analysis of transient CFD data

temporal coefficients. The typical three dimensional structures of the modes for an example flow around two cylinders are depicted in figure 1. To construct the POD, solution vectors are taken from the CFD simulation at certain points in time and collected into a snapshot matrix X of size $m \times n$, where the rows stand for the spatial dimension and the columns represent one snapshot in time. Next, the eigenvalues and eigenvectors of the covariance matrix $C = XX^T$ are computed, in which the latter are called *POD modes*. The *POD modes* can also be derived as the left singular matrix U from the *Singular Value Decomposition* (SVD) of X : $X = UDV^T$. Respectively, the *POD coefficients* correspond to the rows of the scope matrix $T = DV^T$. The SVD approach has some advantages in practise and will be used from here on.

From an algorithmic perspective, two main challenges arise during the computation of the *POD modes* from CFD data. First, we need to run the SVD in parallel to speed up computations and overcome memory limits of a single node. Second, we would like to run it in a streaming fashion, meaning during runtime of the simulation, while the CFD results are being generated and the solution is still in memory. This can avoid large amounts of data in- and output but also allows for a simulation monitoring by investigated intermediate results of the simulation. Especially the latter is an important motivation of an explorative analysis when employed to

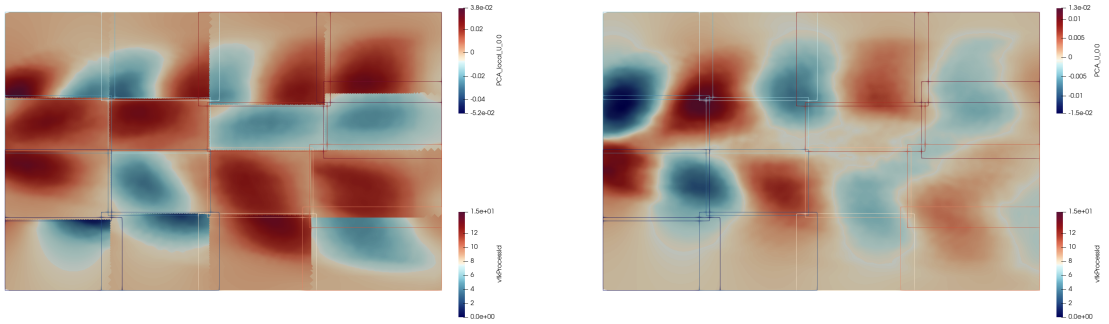

Figure 3: Parallel computation of the POD modes in two steps: local SVD (left) and final results (right)

identify unwanted behavior. This can for example be used as an abort criterion for the simulation and thus potentially save computing time.

Parallel Execution of the SVD

There are multiple approaches for a distributed SVD execution optimized for different scenarios (compare e.g. [8] and [9]). As we already have a predefined decomposition of the geometrical mesh, we chose the split-and-merge approach for the SVD as described in [4]. That way, we can take advantage of the data already being distributed to multiple MPI ranks in the CFD simulation and reduce communication overhead during the analysis steps. A second advantage lies in the simple formulation of the algorithm, including a streaming approach (see below) and a straightforward implementation.

In a first step, the snapshot matrix X is divided by rows into s smaller matrices X_i that can be chosen as the local flow solutions on each MPI rank. It follows a local SVD on each X_i that will be done in parallel on all ranks at the same time. Next, the SVD of a combined eigen-matrix Y of size $n \times ns$ has to be computed which is usually much smaller than the original snapshot matrix of size $m \times n$. The overall algorithms can be summarized in four steps [4]:

1. Partition X by rows into $X = [X_1^T, ..., X_s^T]^T$
2. Perform SVD for each X_i: $X_i = \bar{U}_i D_i V_i^T$
3. Perform SVD for the combined eigen-matrix $Y = [V_1 D_1, ..., V_s D_s]^T$: $Y = U_y D_y V_y^T$
4. Output $\tilde{U}U_y, D_y$ and V_y as the three components of SVD of X, where $\tilde{U}$ is the block diagonal matrix of all U_i.

Figure 3 shows an example of the local and global results of a parallel SVD computation including the processor boundaries with overlapping regions.

Streaming Execution of the SVD

The algorithm, as it is defined above, would need the entire snapshot matrix to be available at once in memory. However, this is oftentimes not feasible for a practical in-situ workflow.

Instead we can split each local matrix X_i again by columns which leads to batches of local snapshots over a couple of time-steps. Starting off with an *SVD* of the local snapshot matrix $X_i^{1:t}$ up to time-step t, we can apply the following algorithm to update the local *SVD* with a new batch X_i^{t+1} iteratively [4]:

a. Find the *SVD* $X_i^{t+1} = U_i^{t+1} D_i^{t+1} {V_i^{t+1}}^T$, and let $\tilde{V} = diag(V_i^{1:t}, V_i^{t+1})$ be a diagonal block matrix.

b. Let $W = (U_i^{1:t} D_i^{1:t}, U_i^{t+1} D_i^{t+1})$ and find the *SVD* $w = U_w D_w V_w^T$.

c. Output $U_i^{1:t+1} = Uw$, $D_i^{1:t+1} = D_w$ and $V_i^{1:t+1} = \tilde{V} V_w$ as the three components of *SVD* of $X_i^{1:t+1}$.

During the simulation only the current *SVD* results, as well as a small batch of flow snapshots has to be held in memory. Applied in conjunction with the parallel approach from above, the batch updating of the *SVD* will replace step 2. from above. Whenever a global solution is needed steps 3. and 4. can be executed.

The parallel and streaming approach for the POD is implemented in Python based on Scipy for the local *SVD* computations. This ensures a flexible, but efficient implementation and a straightforward in-situ execution in a Catalyst ParaView pipeline, while connecting to any CFD code that supports the Catalyst interface.

2.2 Dimensionality Reduction, Clustering and Classification

The clustering of the simulations and their corresponding modes into groups of similar behaviour is done in two steps. First, a dimensionality reduction is performed by applying the diffusion map algorithm. It is based on the assumption that there are low-dimensional structures in the data that can be extracted by a diffusion process over the data points, followed by a spectral decomposition of the global diffusion distance matrix. Thereby a non-linear embedding of the data points into the Euclidean space is achieved that preserves local diffusion distances which makes it very powerful for exploring data from non-linear physical systems. For more details of the approach we refer to [5] and [6]. Based on the embedded data a k-means clustering is performed. It separates the samples into a predefined number of groups by minimizing the distances of the points to their group-mean (see e.g. [7]).

The k-means algorithm is an unsupervised machine learning technique, thus it cannot be used for predictions. As we would like to be able to predict the characteristic behaviour of the modes for new input parameter, we additionally train a classifier on the clustering results from before. Here, we use a Support Vector Machine (SVM) with a non-linear radial basis functions (RBF) kernel. More details can be found in [7].

3 DOUBLE CYLINDER USE-CASE

As a simple, but interesting use-case for an explorative analysis, we choose the three-dimensional flow around two cylinders with identical diameters D at $Re_D = 10,000$. We generate a parameter study of a total of 60 simulations by varying the X and Y position of the second cylinder between $0 \leq PosX \leq 5$ and $0 \leq PosY \leq 3$. Values are sampled randomly from a latin

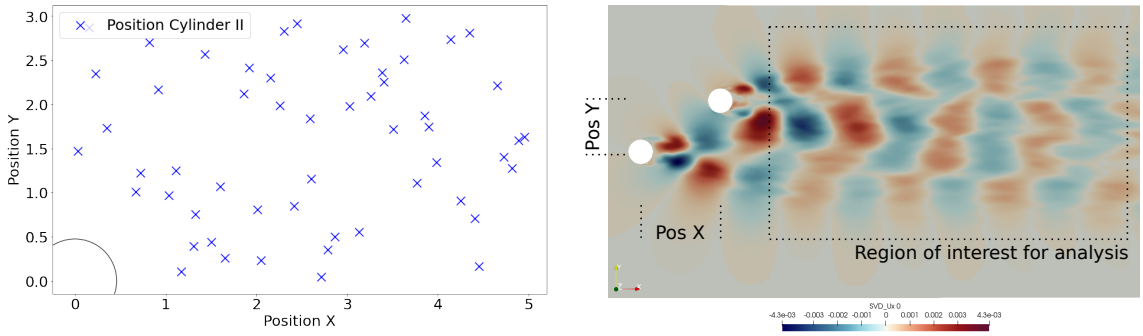

Figure 4: Parameter variations (left) and geometrical setup (right) of the study including the region of interest for analysis

hypercube and scaled to the desired boundaries. Here, to avoid an overlap of the cylinders, the squared area, where $(PosX, PosY) \in [0,1]^2$, is ignored, see figure 4a. The transient Detached Eddy Simulations (DES) are carried out in OpenFOAM Version 6.0, using ParaView Catalyst for in-situ data processing. To allow for an automatic generation of the mesh based on the varying input geometries, we use SnappyHexMesh. The resulting meshes consist of around 2 million mostly hexahedral cells per simulation. The overall duration of one run corresponds to 150 non-dimensional time units based on the cylinder diameter and input velocity.

The streaming POD algorithm from section 2.1 is implemented in Python and wrapped as a ParaView Plugin, which allows a simple online execution as part of a Catalyst pipeline. A total of 600 flow snapshots are processed online in six consecutive batches of size 100 throughout the simulation. Based on flows around single cylinders for similar Reynolds number, we expect one dominant mode for each cylinder that shows periodic behaviour in the wake region. Depending on the vertical and horizontal position of the cylinders, these modal structures will interact with each other in different ways, such as cancelling out or amplifying each other or generating completely new spatial structures.

We will investigate the principal shape of the dominant POD mode by identifying groups of behaviour over all simulations, such as structures with two or three periodic oscillations in the vertical directions or cases where no clear dominant mode is present. Due to the changing geometries and meshes, we focus the analysis on the region behind both cylinders, as defined in figure 4b and interpolate all data to a common reference mesh. As the POD decomposition is computed in parallel on all simulation cores, there is no need to restrict the domain during runtime. Thus, in the most efficient workflow, we do the interpolation only once at the end of the simulation.

4 RESULTS

In the first step of the analysis, the dominant POD modes of all simulations represented as vectors are combined into a sampling matrix X_s. A non-linear dimensionality reduction is performed by applying the diffusion map algorithm as described in section 2.2 to the matrix X_s. Thereby the high-dimensional spatial modes are transformed into a lower dimensional embedding. The first two embedding coordinates are shown in figure 5 (left). We observe a

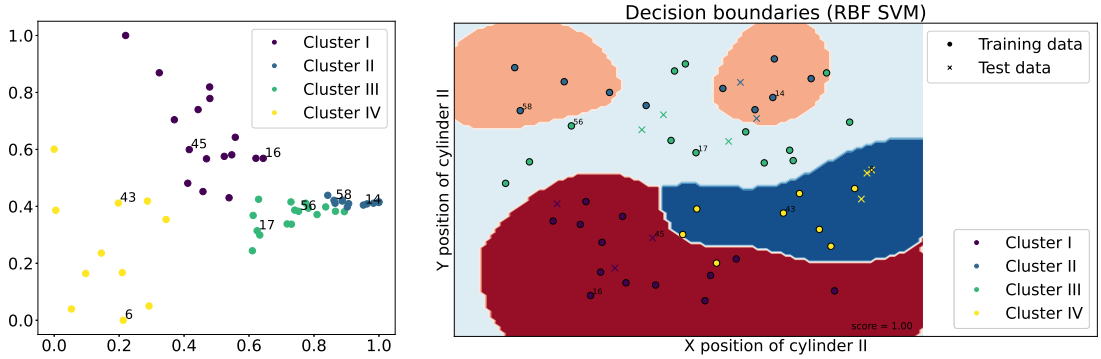

Figure 5: Clustering and Classification

structure that can be divided into several smaller groups. Note that generally it is difficult to select the number of clusters. We choose four cluster for the following analysis. As we will discuss later on, this represents best the different characteristic physical behaviour in the data. Next, the k-means clustering algorithm is applied to assign each simulation to one cluster as represented in colours in figure 5 (left).

To get a better understanding of how particular simulations are grouped, we chose two example modes from each cluster and compare them in figure 6. Similar characteristic shapes of the modes are found for all samples in each cluster. For Cluster I and II, sharp periodic structures can be identified that repeat two and three times, respectively, in the vertical direction. In Cluster IV, there are also three vertical periodic structures that however decay much stronger in the downstream direction. In Cluster III the modal structures are more diffus and do not contain any clearly repeating patterns. Similar behaviour is observed for all points in each cluster, even though, at the connecting regions between two cluster, oftentimes, there are intermediate solutions that could be assigned to both adjacent clusters.

In a further analysis step, we would like to be able to predict the characteristic modal structure for a set of input parameters for which there are no high resolution CFD results. To achieve this, we first plot the input parameters PosX versus PosY in figure 5 (right), coloured with the cluster number from before. While most of the clusters can still be observed as continuous regions in the original parameter space, it seems that there is no simple linear relationship between the input parameters and cluster numbers. Thus, a non-linear classifier, such as the Kernel SVM that we use in the following, seems most promising to predict the flow behaviour. For the training and validation of the classifier, the data is randomly split into a *training* dataset containing 48 samples and a *test* dataset with 12 samples. Finally, the predicted regions are colored with the corresponding cluster number in figure 5 (right) to show the decision boundaries of the classifier. The overall accuracy of the classification of the test data lies in the range of 0.9 - 1.0 depending on the random selection of the training and test samples.

We observe five coherent regions, whereby cluster II is split into two separate sections. Interestingly to note, all simulation cases in both sections of Cluster II show the same principal behaviour of three periodic structure while this is not the case for the simulations that lie in between and that are correctly identified to belong to Cluster III. The overall distribution of the class boundaries leads to the conclusion that only for specific relative locations of the two

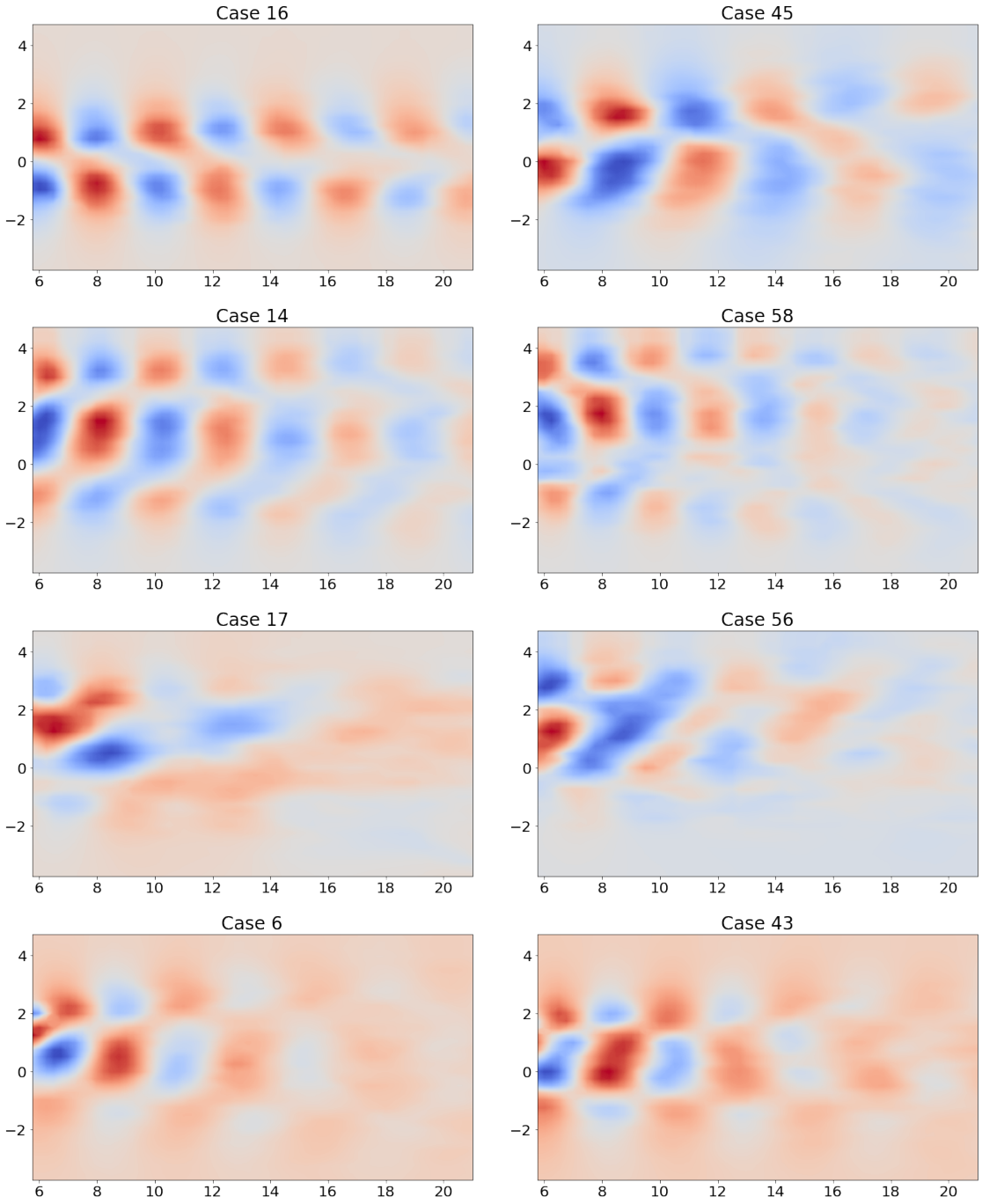

Figure 6: Two selected POD modes per cluster; Cluster I: Case 16 & Case 45, Cluster II: 14 & 58, Cluster III: 17 & 56, Custer IV: 6 & 43.

cylinders to each other, periodic spatial modes are present. These findings could potentially be exploited in various engineering scenarios, for example in cases where oscillations in the flow solution should be avoided at specific frequencies to prevent an amplification of eigen-modes in the surrounding structures.

5 CONCLUSION

We proposed a workflow for the explorative analysis of transient CFD simulations with the goal to simplify the comparison of large simulation bundles. The key technical elements are i) the in-situ extraction of POD modes, ii) a non-linear dimensionality reduction via diffusion maps and III) clustering of the data. The data analytics workflow was implemented in Python and attached to a OpenFOAM simulation using the ParaView Catalyst interface. A parameter study of 60 simulations was constructed by simulating the flow around two cylinders with varying relative positions to each other.

In the low dimensional embedding of the dominant POD modes, we found four different clusters that represent different characteristic function shapes. Differences lie in the number and vertical position of periodic structure, as well as the decay in the stream-wise direction. Finally, we train a classifier to predict the principal class of physical behaviour of the dominant mode for a new set of input parameters. Predictions were accurate in more than 90% of the investigated test cases without the use of information from the CFD simulation.

While this concludes the explorative analysis, the obtained results could be the input for further analysis steps. As an example, in cases where the physical solution varies in a non-linear way over the parameter space, it could be difficult to build a global surrogate modal to predict new solutions. However, when applying our workflow first, the global parameter space can be divided into smaller regions with similar physical behaviour. Based on these, local prediction models can be trained that - combined together - can provide better prediction over the entire space.

REFERENCES

[1] J. Garcke, R. Iza-Teran and C. Gscheidle, Analysis of Turbulent Flow Data Based on a Spectral Basis Representation. *NAFEMS World Congress*, 2019.

[2] R. Iza-Teran and J. Garcke, A geometrical method for low-dimensional representations of simulations, *SIAM/ASA Journal on Uncertainty Quantification*, 2019, Vol. 7, No. 2 : pp. 472-496

[3] C. Gscheidle, J. Meng and J. Garcke, An efficient and flexible in-situ data analysis framework for CFD simulations. *NAFEMS World Congress*, 2021.

[4] F. Liang, R. Shi and Q. Mo. *A Split-and-Merge Approach for Singular Value Decomposition of Large-Scale Matrices*. Stat Interface. 2016

[5] M. Belkin and P. Niyogi, *Laplacian Eigenmaps for Dimensionality Reduction and Data Representation*. Neural Computation, vol. 15, no. 6, pp. 1373-1396, 1 June 2003, doi: 10.1162/089976603321780317.

[6] R. Coifman and S. Lafon, *Diffusion maps.* Applied and Computational Harmonic Analysis, Volume 21, Issue 1, 2006, Pages 5-30, ISSN 1063-5203, https://doi.org/10.1016/j.acha.2006.04.006.

[7] C. M. Bishop, *Pattern Recognition and Machine Learning.* Springer-Verlag New York, 2006, ISBN 978-1-4939-3843-8.

[8] G. M. Oxberry, T. Kostova-Vassilevska, W. Arrighi, and K. Chand *Limited-memory adaptive snapshot selection for proper orthogonal decomposition.* Int. J. Numer. Meth. Engng, 109: 198– 217, 2017, doi: 10.1002/nme.5283.

[9] C.G. Baker, K.A. Gallivan and P. Van Dooren, *Low-rank incremental methods for computing dominant singular subspaces.* Linear Algebra and its Applications, Volume 436, Issue 8, 2012, Pages 2866-2888, ISSN 0024-3795, https://doi.org/10.1016/j.laa.2011.07.018.