Linear and Nonlinear Dimensionality Reduction from Fluid Mechanics to Machine Learning

Miguel A. Mendez
von Karman Institute for Fluid Dynamics, Sint-Genesius-Rode, Belgium

Abstract

Dimensionality reduction is the essence of many data processing problems, including filtering, data compression, reduced-order modeling and pattern analysis. While traditionally tackled using linear tools in the fluid dynamics community, nonlinear tools from machine learning are becoming increasingly popular. This article, halfway between a review and a tutorial, introduces a general framework for linear and nonlinear dimensionality reduction techniques. Differences and links between autoencoders and manifold learning methods are highlighted, and popular nonlinear techniques such as kernel Principal Component Analysis (kPCA), isometric feature learning (ISOMAPs) and Locally Linear Embedding (LLE) are placed in this framework. These algorithms are benchmarked in three classic problems: 1) filtering, 2) identification of oscillatory patterns, and 3) data compression. Their performances are compared against the traditional Proper Orthogonal Decomposition (POD) to provide a perspective on their diffusion in fluid dynamics.

Keywords: Dimensionality Reduction, Kernel PCA, ISOMAPs, Locally Linear Embedding
Contents

1 Introduction 3

2 The Problem Set 4

3 Autoencoders and Manifold Learning 5
   3.1 From POD (PCA) to MDS ........................................... 5
   3.2 From POD (PCA) to kernel PCA (POD) ........................ 7
   3.3 From MDS to ISOMAPS ............................................ 9
   3.4 From MDS to LLE .................................................. 10
   3.5 A note on the Decoding Process ................................ 11

4 Selected Test Cases 12
   4.1 Test Case 1: Background Removal in PIV Images .............. 12
   4.2 Test Case 2: Turbulent flow past a cylinder in transient conditions ...... 14
   4.3 Test Case 3: An impinging jet flow .............................. 15

5 Results 16
   5.1 Test Case 1: Background Removal Problem ....................... 16
   5.2 Test Case 2: Transient Flow past a Cylinder ...................... 18
   5.3 Test Case 3: Impinging Jet Flow ................................ 22

6 Conclusions and Perspectives 23
1 Introduction

Dimensionality reduction is an essential branch of machine learning and data engineering, concerned with identifying ways to encode high dimensional data in a low dimensional representation (see Alpaydın (2020); Bishop (2011); Murphy (2012)). Many of its methods are widespread across different disciplines under different names.

In fluid dynamics, the popularity of dimensionality reduction can be traced back to the introduction of the Proper Orthogonal Decomposition (POD) as a tool to identify (and define) coherent structures in turbulent flows (Lumley, 1970; Lumley and Poje, 1997). The idea that a seemingly chaotic dynamic could be pictured as a linear combination of coherent structures has fueled the interest in the “dynamical system perspective” of turbulent flows (see Berkooz et al. (1993); Holmes et al. (1996)) and the development of model order reduction techniques based on Galerkin projection (see Ahmed et al. (2021); Benner (2020); Benner et al. (2015); Cordier and Bergmann (2013) for exhaustive reviews).

The diffusion of the POD in both experimental and numerical fluid dynamics is largely due to the works of Sirovich (1987, 1989, 1991) and Aubry (1991); Aubry et al. (1991) who have proposed efficient algorithms to compute the POD and analyzed their link to the original formulation by Lumley (1970) (see also George (2016)). With the continuous development of experimental fluid mechanics– and in particular Particle Image Velocimetry (PIV)–, the use of POD to deduce coherent structures from experimental data has boomed in the last two decades (see for example Bi et al. (2003); Citriniti and George (2000); Gordeyev and Thomas (2000); Mendez et al. (2020, 2018); Pollard et al. (2017); Semeraro et al. (2012); Sullivan and Pollard (1996)). Focusing on the literature in experimental fluid dynamics, the POD has also been extensively used as a filter, for example to remove outliers in PIV measurement (Higham et al., 2016; Raiola et al., 2015) or to pre-process velocity fields for pressure integration (Charonko et al., 2010), to pre-process images (Mendez et al., 2017), to fill ‘gaps’ in experimental data (Saini et al., 2016) to construct efficient regressors and interpolators (Bouhoubeiny and Druault, 2009; Casa and Krueger, 2013; Karri et al., 2009; Ratz et al., 2022), to validate numerical simulations (Kriegseis et al., 2010) or to build estimators of quasi-periodic flows (Bourgeois et al., 2013; Loiseau et al., 2018). Moreover, the POD has been used to enhance adaptive least square problems (see Yao et al. (2017)), fault diagnostics (Shen et al., 2022) or optimal pressure placement (Castillo and Messina, 2020).

In the last decade, many variants and hybrid formulations of the POD have been developed (see Mendez et al. (2019); Sieber et al. (2016); Towne et al. (2018)) alongside alternative data-driven decompositions. Among these, the most popular alternative is the Dynamic Mode Decomposition (DMD) proposed by Schmid (2010) and Rowley et al. (2009), also existing in many variants (see Clainche and Vega (2017); Jovanović et al. (2014); Kutz et al. (2016); Noack et al. (2016); Tu et al. (2014)). The DMD can be seen as a variant of the Principal Oscillation Pattern (POP), and Linear Inverse Method (LIM) approaches diffused in climatology (Hasselmann, 1988; Penland, 1996; von Storch and Xu, 1990) in the late ’80s. Although differing slightly in their algorithmic implementation, all DMD formulations fit a linear dynamical system to the dataset to identify the dominant oscillatory patterns.

More broadly, the study of ways of decomposing a flow as a linear combination of “coherent patterns”, referred to as “modes” has evolved into a well-established branch of data processing, often referred to as data-driven modal analysis (see Taira et al. (2017, 2020)). Within the machine learning literature, all modal decompositions fall in the category of linear dimensionality techniques. Although fundamental both conceptually and in practice, these are only a fraction of dimensionality reduction methods routinely used in data science (see Ayesha et al. (2020); Ghojogh et al. (2019b); Veliangiri et al. (2019)). Most of the recent development in the field focused on nonlinear methods for dimensionality reduction. These, somewhat surprisingly, are currently relatively unexplored in fluid dynamics. A recent review on these methods is provided.
by Csala et al. (2022), who focused on their application to numerical data.

Among (nonlinear) dimensionality reduction techniques, one can identify two main classes of methods: (1) autoencoders and (2) manifold learning methods. Autoencoders seek a compressed representation of the data while preserving as much information as possible. The POD – known as Principal Component Analysis (PCA) in the machine learning literature (Ghojogh et al., 2019a)– is a linear autoencoder (Baldi and Hornik, 1989; Milano and Kounoutsakos, 2002). Manifold learning methods are not concerned with the loss of information in the reconstruction but seek low dimensional representation that preserves as much as possible some measure of similarity (Zheng and Xue, 2009). The difference between these two classes of methods is subtle (some autoencoders can be used for manifold learning and vice-versa) but essential: most algorithms for manifold learning lead to compressed representations that do not admit ‘an inverse’, that is a mapping back to the original space.

Among the autoencoders, the most classic nonlinear approaches are Artificial Neural Networks (ANN) autoencoders (Alain and Bengio, 2012; Goodfellow et al., 2016) and kernel PCA (Ghojogh et al., 2019a; Schölkopf et al., 1997). While ANN are becoming increasingly popular in fluid dynamics (Agostini, 2020; Eivazi et al., 2020; Fukami et al., 2020, 2021; Guastoni et al., 2021; Pawar et al., 2019; Singh et al., 2020), the kPCA has not yet been exploited in the field. Among the manifold learning techniques, the most popular ones are arguably the Locally Linear Embedding (LLE) Ghojogh et al. (2020b); Roweis and Saul (2000); Saul and Roweis (2001) and ISOMAPs (Ghojogh et al., 2020a; Tenenbaum et al., 2000). These are now entering the fluid dynamics community as alternatives to the POD for reduced order modeling (Chen and He, 2022) and for finding compressed representation of fluid flows (Ehlert et al., 2019; Farzamnik et al., 2022; Tauro et al., 2014).

The scope of this work is to provide a tutorial and a comparative analysis of kPCA, LLE and ISOMAP, benchmarking these relatively ‘new’ methods against the POD for three applications of great interest to the experimental fluid dynamicist: 1) filtering, 2) identification of oscillatory patterns, and 3) data compression.

The rest of the article is structured as follows. Section 2 introduces the problem set and the mathematical framework. Section 3 analyzes the connection between the POD and the kPCA, LLE and ISOMAP. Following the footsteps of the excellent tutorial by Ghojogh et al. (2020a), this is done by building bridges with the most elementary manifold learning methods known as Multi-dimensional scaling (MDS, see Torgerson (1952)). Section 4 introduces the three selected test cases. The first is an image filtering problem: the background removal from PIV images. The second is a manifold learning problem, namely identifying oscillatory patterns in a vortex shedding problem. The third is a data compression problem concerned with analysing a flow configuration that does not exhibit a clear low dimensional representation. Results are presented in Section 5 while conclusions and perspectives are summarized in Section 6.

2 The Problem Set

Our starting point is a dataset rearranged in the form of a snapshot matrix $D \in \mathbb{R}^{n_s \times n_t}$. This contains $n_t$ snapshots (columns) collecting $n_s$ data points each. Using a Python-like notation, we denote as $d_k = D[:,k] \in \mathbb{R}^{n_s}$ the k-th column (snapshot) in $D$. The reshaping into a column vector is carried out regardless of the nature of the dataset.

Considering for example a planar PIV measurement, providing a velocity field $u(x_i, y_j, t_k) = (u(x_i, y_j, t_k), v(x_i, y_j, t_k))^T$ over a regular grid $x = (x_i, y_j)$ with $i = 1, \ldots n_x$ and $j = 1, \ldots n_y$. A snapshot vector can be obtained by flattening the two scalar fields $u, v$ into vectors and sticking these one below the other to have $n_s = 2n_xn_y$. If the dataset contains a video sequence, then each snapshot is obtained by reshaping the images.

In this article, we assume that data is available in a Cartesian and uniform grid with spacing $\Delta x$ in both directions. Then, in a first approximation one can assume that the data is constant
within an area \( \Delta x^2 \) (piece-wise constant interpolation). Denoting as \( \Omega \) the domain (area) over which the velocity was sampled, the \( L_2 \) norm in \( \Omega \) can then be linked to the \( l_2 \) norm in \( \mathbb{R}^{n_s} \) (see Mendez et al. (2019)). Similarly, assuming that the snapshots have been collected at uniform time intervals \( \Delta t \), so that \( t_k = k\Delta t \), with \( k = 0, \ldots n_t - 1 \) and \( f_s = 1/\Delta t \) is the sampling frequency, the \( L_2 \) norm in the time interval of duration \( T = n_t \Delta t \) can be linked to the \( l_2 \) norm in \( \mathbb{R}^{n_s} \).

We here focus on dimensionality reduction technique in the space domain, i.e. along the columns of \( \mathbf{D} \). Therefore, we seek to map our dataset \( \mathbf{D} \in \mathbb{R}^{n_s \times n_t} \) into a compressed representation \( \mathbf{Z} \in \mathbb{R}^{n_r \times n_t} \). The column \( \mathbf{z}_k = \mathbf{Z}[:, k] \in \mathbb{R}^{n_r} \) is thus the reduced representation of the snapshot \( \mathbf{d}_k \in \mathbb{R}^{n_s} \), with \( n_r \ll n_s \).

In autoencoders, the goal is to find the reduced representation that preserves most of the relevant information. We thus define two mappings. The first, denoted as encoder \( \mathcal{E} \), deals with the compression \( \mathbf{E} : \mathbb{R}^{n_s} \rightarrow \mathbb{R}^{n_r} \), i.e. \( \mathbf{Z} = \mathcal{E}(\mathbf{D}) \). The second, which we denote as decoder \( \mathcal{D} \), deals with the reverse process \( \mathbf{D} : \mathbb{R}^{n_r} \rightarrow \mathbb{R}^{n_s} \), i.e. \( \mathbf{D} = \mathcal{D}(\mathbf{Z}) \). The tilde here denotes “approximation of”, because the mapping \( \mathbb{R}^{n_s} \rightarrow \mathbb{R}^{n_r} \rightarrow \mathbb{R}^{n_s} \), denoted as autoencoder \( \mathcal{A} = \mathcal{D}(\mathcal{E}(\cdot)) \), is lossy. Preserving essential information means having \( \tilde{\mathbf{D}} \approx \mathbf{D} \).

In manifold learning, there is usually no decoder. The goal is to find the compressed representation that preserves (as much as possible) some measure of pair-wise similarity in the original space and in the reduced space. That is, defining as \( S_d(\cdot) : \mathbb{R}^{n_s} \times \mathbb{R}^{n_s} \rightarrow \mathbb{R} \) a similarity measure in \( \mathbb{R}^{n_s} \) and \( S_z(\cdot) : \mathbb{R}^{n_r} \times \mathbb{R}^{n_r} \rightarrow \mathbb{R} \) a similarity measure in \( \mathbb{R}^{n_r} \), preserving similarity means having \( S_d(\mathbf{d}_i, \mathbf{d}_j) \approx S_z(\mathbf{z}_i, \mathbf{z}_j) \) for all \( i, j \).

### 3 Autoencoders and Manifold Learning

#### 3.1 From POD (PCA) to MDS

We begin by analyzing the link between classic POD and Multidimensional Scaling (MS). This link connects methods from modal decomposition to methods from manifold learning. Assuming uniform sampling in space and time, as mentioned in the previous section, the POD is equivalent to the Principal Component Analysis (PCA).

The POD (PCA) is a linear autoencoder. The linear encoder \( \mathcal{E} : \mathbb{R}^{n_s} \rightarrow \mathbb{R}^{n_r} \) is a projection onto an orthogonal\(^1\) basis of \( n_r \) vectors, that is \( \mathbf{Z} = \Phi^T \mathbf{D} \), with \( \Phi \in \mathbb{R}^{n_s \times n_r} \) the matrix collecting the basis elements along the columns. These basis elements, denoted as \( \phi_r = \Phi[:, r] \), are the coherent structures (or fields) in the data and the columns \( \mathbf{z}_k = \mathbf{Z}[:, k] \in \mathbb{R}^{n_r} \) are the reduced representation of the snapshot \( \mathbf{d}_k = \mathbf{D}[:, k] \in \mathbb{R}^{n_s} \) with respect to that basis. The decoder is a linear combination of these structures, that is \( \mathbf{D} = \Phi \mathbf{Z} \). The autoencoder is thus \( \tilde{\mathbf{D}} = \Phi \Phi^T \mathbf{D} \), and all mappings are given once the basis matrix \( \Phi \) is given.

In classic POD (PCA), this matrix is the one that minimizes

\[
J(\Phi) = || \mathbf{D} - \tilde{\mathbf{D}} ||_2^2 = || \mathbf{D} - \Phi \Phi^T \mathbf{D} ||_2^2 ,
\]

where \( || \cdot ||_2 \) is the induced \( l_2 \) norm.

To hope for a unique solution, we impose that these vectors have unitary length, hence \( \Phi^T \Phi = I \), with \( I \) the identity matrix of appropriate size. The minimization of \( J(\Phi) \), under the orthonormality constraints, leads to the well-known eigenvalue problem (see Bishop (2011))

\[
( \mathbf{D} \mathbf{D}^T ) \Phi_P = \Phi_P \Lambda_P ,
\]

with \( \Lambda_P = \text{diag}(\lambda_{P1}, \lambda_{P2}, \ldots , \lambda_{Pn_r}) \) the diagonal matrix collecting the non-negative eigenvalues \( \lambda_r \) associated to the basis element \( \phi_{Pr} \). A unique decomposition can be found if all eigenvalues

\(^1\)Note that orthogonality is not a constraint but a convinient choice for an autoencoder aiming at preserving information. The most popular non-orthogonal autoencoders is the DMD.
are distinct (which is not the case in uncorrelated noise). In what follows, we use the subscript \( \mathcal{P} \) to distinguish the PCA/POD bases from the others and define \( C := DD^T \).

In fluid dynamics, \( C \) is the spatial correlation matrix and solving for \( \Phi \) via (2) is known as ‘classic POD’ or ‘space-only’ POD (George, 2016; Towne et al., 2018). In machine learning and statistics, \( C \) is the covariance matrix, and the classic POD is known as ‘classic’ PCA (Ghojogh and Crowley, 2019). An alternative formulation considers the encoding (projection) along the rows rather than columns (i.e. in space rather than time in our settings).

To analyze their link, let us factorize the reduced representation as \( Z_\mathcal{P} = \Sigma_\mathcal{P} \Psi_\mathcal{P}^T \), with \( \Sigma_\mathcal{P} = \text{diag}(\sigma_{p1}, \ldots, \sigma_{pn}) \in \mathbb{R}^{n_r \times n_r} \) a diagonal matrix containing the \( l_2 \) norms of the rows \( Z_\mathcal{P} \), i.e. \( \sigma_{p_r} = \|z_r\|_2 \). Then, \( \Psi_\mathcal{P}^T \) is a normalized version of \( Z_\mathcal{P} \) and the columns of \( \Psi_\mathcal{P} \in \mathbb{R}^{n_r \times n_r} \) describe how each of the coefficients in a reduced representation \( z \in \mathbb{R}^{nr} \) change to describe all the snapshots. We now have:

\[
Z_\mathcal{P} = \Phi_\mathcal{P}^T D = \Sigma_\mathcal{P} \Psi_\mathcal{P}^T \rightarrow \tilde{D} = \Phi_\mathcal{P} \Sigma_\mathcal{P} \Psi_\mathcal{P}^T.
\]

Equation (3) is a truncated singular value decomposition (SVD) of the snapshot matrix. Combining this with (2), one can see that \( \Sigma_\mathcal{P} = \Lambda_\mathcal{P}^{1/2} \) and that the columns of \( \Psi \) must also be orthonormal, i.e. \( \Psi_\mathcal{P}^T \Psi_\mathcal{P} = I \). These are eigenvectors of another important matrix, i.e.

\[
(D^T D) \Psi_\mathcal{P} = \Psi_\mathcal{P} \Lambda_\mathcal{P},
\]

with eigenvalues shared with (2). In what follow, we define \( K := D^T D \). In fluid dynamics, \( K \) is the temporal correlation matrix and computing the reduced representation \( Z \) from \( \Psi \) using (4) (thus avoiding computing \( \Phi \)) is known as ‘snapshot POD’ (Holmes et al., 1996). In machine learning, \( K \) is the Gram matrix and the snapshot POD is known as ‘dual PCA’ (Ghojogh and Crowley, 2019). Moreover, in fluid dynamics, the columns of \( \Phi \) are spatial structures while the columns of \( \Psi \) are temporal structures of each mode. In machine learning, the columns of \( \Phi_\mathcal{P} \) are principal components while the rows of \( Z_\mathcal{P} \) are referred to as scores.

The two alternatives of encoding/decoding yields two ways of autoencoding the same dataset, i.e. \( \tilde{D} = \Phi_\mathcal{P} \Phi_\mathcal{P}^T D \) or \( \tilde{D} = D \Psi_\mathcal{P} \Psi_\mathcal{P}^T \). Therefore, the fact that this is the linear autoencoder that minimizes (1) implies that the matrices \( \Phi \Phi^T \) and \( \Psi \Psi^T \) are as close as possible to the identity matrices. Note now that the temporal correlation matrix in the reduced space is:

\[
Z_\mathcal{P}^T Z_\mathcal{P} = (\Phi_\mathcal{P}^T D)^T (\Phi_\mathcal{P}^T D) = D^T \Phi_\mathcal{P} \Phi_\mathcal{P}^T D.
\]

Thus we see that the temporal correlation matrix in the reduced space (\( Z_\mathcal{P}^T Z_\mathcal{P} \)) is as close as possible to the one in the original space (\( D^T D \)) if \( \Phi_\mathcal{P} \Phi_\mathcal{P}^T \) is as close as possible to the identity matrix. We have entered manifold learning: the temporal correlation matrix (or Gram matrix) can be seen as a measure of similarity, i.e. \( S_d(d_k, d_j) = d_k^T d_k \) and \( S_z(z_k, z_j) = z_k^T z_k \), and we have just shown that the reduced representations in the PCA/POD is such that \( S_d \approx S_z \) as much as possible.

We can now move to the final step. Let \( r_{i,j} \) denote the squared Euclidean distance between two snapshots \( d_i \) and \( d_j \). We thus have:

\[
r_{i,j} = \|d_i - d_j\|^2 = d_i^T d_i - 2d_i^T d_j + d_j^T d_j.
\]

Denoting as \( R \) the matrix collecting all the square distances, i.e. \( R[i,j] = r_{i,j} \), and defining \( k := \text{diag}(K) \), equation (6) can be written as \( R = 1k - 2K + k1^T \), where \( 1 \in \mathbb{R}^{n_t} \) is the vector of ones. Because the matrix of distances \( R \) is invariant to shifts of the origin, it is worth centering the data by removing the average rows and the average columns of \( R \). This operation is called double-centering and can be performed using the centering matrix \( H := I - 1/n_t 11^T \in \mathbb{R}^{n_t} \), so
that the double centered distance matrix is $HRH$. It is now possible to derive the following important relation between $K$ and $R$ (see Ghojogh et al. (2020a)): 

$$HKH = -\frac{1}{2}HRH.$$  

(7)

Noticing that centering operation is idempotent, if the data matrix has been centered (that is the temporal average has been removed to all columns), then $D \leftarrow DH$, and $K \leftarrow HKH$. We thus have a connection between a measure of similarity in terms of correlation or angles (i.e. $K$) and a measure of similarity in terms of distances (i.e. $R$).

For a centered dataset, if the Euclidean inner product is used to measure angles and the Eulerian distance is used to measure distances, preserving angles is equivalent to preserving distances. Computing $K$ from (7) and proceeding with the snapshot POD leads to the ‘classic’ Multidimensional Scaling (MDS) by Torgerson (1952). But the idea can be generalized. The Kernel PCA introduces a different way of measuring angles while the ISOMAP introduces a different way of measuring distances. To conclude this section, the following Python function can be used to compute the $n_r$ dimensional POD encoding $Z_P$ for the data matrix $D$ using the snapshot-based approach:

```python
def PCA_TRANS(D, n_r=2):
    # Compute Temporal Correlation Matrix K
    K=D.T@D; n_t=np.shape(D)[1]
    # Diagonalize and Sort
    lambdas, Psi_P=linalg.eigh(K, subset_by_index=[n_t-n_r, n_t-1])
    lambdas, Psi_P = lambdas[::-1], Psi_P[:,::-1]
    # Take the first n_r modes
    Sigma_P=np.sqrt(lambdas[:n_r]); Psi_P = Psi_P[:,:n_r]
    # Encode
    Z_P=np.diag(Sigma_P)@Psi_P.T
    return Z_P, Sigma_P
```

In the previous code, it is assumed that the package package `linalg` is importent from scipy (Virtanen et al., 2020) and numpy (Harris et al., 2020) is imported as np. A computationally more efficient approach consist in using the randomized SVD (Halko et al., 2011), available in the scikit-learn (Pedregosa et al., 2011):

```python
def PCA_TRANS_rand(D, n_r=2):
    # Compute Temporal Correlation Matrix K
    K=D.T@D
    # Diagonalize using truncated SVD
    svd = TruncatedSVD(n_r); svd.fit_transform(K)
    Psi_P = svd.components_.T; Lambda_P=svd.singular_values_
    Sigma_P=np.sqrt(Lambda_P)
    # Encode
    Z_P=np.diag(Sigma_P)@Psi_P.T
    return Z_P, Sigma_P
```

This approach exploits the fact that the SVD of $K$ is equivalent to its eigenvalue decomposition and the randomized approach is much faster. However, it is also numerically less accurate. The randomised approach can be used to replace the eigenvalue decomposition of hermitian matrices also in the kPCA and ISOMAPs.

### 3.2 From POD (PCA) to kernel PCA (POD)

The kernel (kPCA) (Schölkopf et al., 1997) is a kernelized version of the PCA which modifies the left hand side of (7) by replacing the temporal correlation (Gram) matrix with a kernel matrix. The underlying idea is to perform the PCA on a dataset which has been first transformed by a nonlinear function $\xi : \mathbb{R}^n_s \rightarrow \mathbb{R}^n_f$. This function brings the data onto a higher (potentially infinite) dimensional space called feature space.
The motivation for this transformation is that linear operations in the feature space (e.g. projections) are nonlinear in the original space, and the function \( \xi \) can (hopefully) highlight special features in the data. The PCA in this nonlinear space is the kernel PCA.

Let us briefly review the idea behind the kernelization, which is performed using the popular kernel trick (Schölkopf et al., 1997). This consists in avoiding operations in the feature space (and even an explicit definition of \( \xi \)) by using a function, called kernel function \( \kappa(\mathbf{d}_i, \mathbf{d}_j) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R} \), to compute the inner products in \( \mathbb{R}^n \). Thus we have:

\[
\kappa(\mathbf{d}_i, \mathbf{d}_j) = \langle \xi(\mathbf{d}_i), \xi(\mathbf{d}_j) \rangle = \xi^T(\mathbf{d}_i)\xi(\mathbf{d}_j). 
\]

(8)

In this work, we consider a Gaussian kernel \( \kappa(\mathbf{d}_i, \mathbf{d}_j) = \exp(-\gamma r_{i,j}) \), with \( \gamma > 0 \) an hyperparameter. This is usually of the order \( \gamma \sim 1/n_s \) or can be set by fixing the least value of \( \kappa(\mathbf{d}_i, \mathbf{d}_j) \). Denoting as \( r_M = \max(r_{i,j}) \) the largest squared Euclidean distance in the dataset and as \( \kappa_m \) the lowest limit for the kernel function, one has \( \gamma = -\ln(\kappa_m)/r_M \).

Defining \( K_{\xi} := \xi(\mathbf{D})^T\xi(\mathbf{D}) \in \mathbb{R}^{n_s \times n_s} \), one thus has \( K_{\xi}[i,j] = \kappa(\mathbf{d}_i, \mathbf{d}_j) \). Furthermore, let \( C_{\xi} := \langle \xi(\mathbf{D})\xi(\mathbf{D})^T \in \mathbb{R}^{n_s \times n_s} \rangle \) and let \( \Phi_{\xi} = [\phi_{\xi1}, \ldots, \phi_{\xi n_s}] \) be the set of principal components in the feature space. Approximating these as linear combinations of the features gives

\[
\phi_{\xi r} = \sum_{i=1}^{n_s} \Psi_i[r,i]\xi(\mathbf{d}_i) \rightarrow \Phi_{\xi} = \xi(\mathbf{D}) \Psi_{\xi}. 
\]

(9)

Then, multiplying the eigenvalue problem for \( K_{\xi} \) by \( \xi^T(\mathbf{D}) \) and introducing (9) gives

\[
\xi(\mathbf{D})^T\xi(\mathbf{D})\xi(\mathbf{D})\Psi_{\xi} = \xi(\mathbf{D})^T\xi(\mathbf{D})\Psi_{\xi}\Lambda_{\xi} \rightarrow K_{\xi}^2\Psi_{\xi} = K_{\xi}\Psi_{\xi}\Lambda_{\xi}. 
\]

(10)

As long as \( K_{\xi} \) is non-singular, this results in an eigenvalue problem for \( K_{\xi} \). Therefore, the kernelized version of equation (3), that is the kPCA encoder, is

\[
Z_{\xi} = \Phi_{\xi}^T \xi(\mathbf{D}) = \Sigma_{\xi}\Psi_{\xi}^T \quad \text{or, equivalently,} \quad Z_{\xi} = \Psi_{\xi} \Phi_{\xi}, 
\]

(11)

with \( \Sigma_{\xi} = \Lambda_{\xi}^{1/2} \).

We thus see that the encoding algorithm is identical to the snapshot POD once the temporal correlation matrix is replaced by the kernel matrix. On the other hand, the decoding is a much more complex (and generally not well posed) problem. The ‘natural’ decoder requires identifying both \( \xi() \) and the eigenfunctions \( \Phi_{\xi} \), to construct the truncated SVD in the feature space, and the inverse \( \xi^{-1}() \) to map the approximation from the feature space to the original space. None of these operation is usually possible. We return to this point in Section 3.5.

The python function below can be used to compute the \( n_r \) dimensional encoding \( Z_{\xi} \) using a Gaussian kernel with \( \gamma \) given by the minimal kernel value \( \kappa_m \).

```python
def kPCA_TRANS(D,n_r=2,k_m=0.1,cen='y'):

# Compute Euclidean distances
r_ij=pdist(D.T,'sqeuclidean'); R=squareform(r_ij)

# Look for the largest distance and define gamma:
M=np.max(R); gamma=-np.log(k_m)/M

# Compute the Kernel Matrix
K_zeta=np.exp(-gamma*R) ; n_t=np.shape(D)[1]

# Center the Kernel Matrix (if cen is 'y'):
if cen == 'y':

H=np.eye(n_t)-1/n_t*np.ones_like(K_zeta)
K_zeta=H@K_zeta@H.T

# Diagonalize and Sort
lambdas,Psixi=linalg.eigh(K_zeta,subset_by_index=[n_t-n_r,n_t-1])

# Encode
Z_xi=np.diag(Sigma_xi)@Psixi.T

return Z_xi, Sigma_xi
```
The functions `pdist` and `squareform` are imported from `scipy.spatial.distance`. The last steps in lines 13 to 18 are identical to the POD implementation while the optional argument `cent` allows the user to decide if performing the kernel centering using the matrix $H$ introduced in Section 3.1. Such centering might not be relevant in non-stationary datasets.

### 3.3 From MDS to ISOMAPs

The isometric feature mapping (ISOMAP, Tenenbaum et al. (2000)) modifies the right hand side of (7) by replacing the Euclidean distance with an approximation of the geodesic distance. This is the shortest path between two points that preserve the topology of the manifold over which they lay. This notion is illustrated in the sketch in Figure 1.

Let $d_1, d_2, d_3, d_4$ be four points in $\mathbb{R}^2$, laying on a curved manifold (shown with dashed lines). The Euclidean distance $r_{1,4}$ ignores the shape of the manifold while the geodesic distance is computed on the manifold. The approximated geodesic distance in this example is $g_{1,4} = r_{1,2} + r_{2,3} + r_{3,4}$.

![Figure 1: Schematic illustration of the geodesic distance versus Euclidean distance. The Euclidean distance between the point $d_1$ and $d_4$ is $r_{1,4}$, shown in red. The (approximated) geodesic distance is $g_{1,4} = r_{1,2} + r_{2,3} + r_{3,4}$.](image)

The approximated geodesic distance is also referred to as curvilinear distance (Lee et al., 2002) and can be approximated by summing the next-neighbours distances. The identification of the neighbors for each snapshot can be carried out using classic k-nearest neighbours algorithms (Rivest et al., 2009). Assuming, for the sake of simplicity, that the data has been centered, one can identify the kernel matrix $K_I$:

$$K_I = -\frac{1}{2}HG^2,$$

with $G[i,j] = g_{i,j}$ the matrix collecting the approximated geodesic distances. The reader is referred to Choi and Choi (2004) for the kernel formulation of the ISOMAP, to Ghojogh et al. (2021) for an analysis of the links between kPCA and ISOMAPs and to Cox and Cox (2008) for a discussion on the centering of this kernel matrix. Once $K_I$ is computed, the ISOMAP encoding proceeds like the snapshot POD, that is

$$Z_I = \Sigma_I \Psi_I^T \quad \text{with} \quad K_I \Psi_I = \Sigma_I \Psi_I.$$

Concerning the decoding, like the kPCA and all manifold learning techniques, the problem is ill posed. We return to this point in section 3.5. The script that follows provides a python function to compute the $n_r$ dimensional encoding $Z_I$ for an ISOMAP with $k$ nearest neighbors.
11 \( H = \text{np.eye(n_t)} - \frac{1}{n_t} \times \text{np.ones_like(G)} \)
12 \( K_I = H @ G @ H^T \)
13 # Diagonalize and Sort
14 \( \text{lambdas, Psi_I = linalg.eigh(K_I , subset_by_index=[n_t-n_r , n_t-1])} \)
15 \( \text{lambdas, Psi_I = lambdas[::-1]} , \text{Psi_I[:,::-1]} ; \text{Sigma_I= np.sqrt(lambdas)} \)
16 # Encode
17 \( Z_I = \text{np.diag(Sigma_I)} @ \text{Psi_I.T} \)
18 return Z_I , Sigma_I

The functions \texttt{NearestNeighbors} and \texttt{kneighbors_graph} are imported from \texttt{sklearn.neighbors} in the scikit-learn (Pedregosa et al., 2011) library. The function \texttt{shortest_path} is imported from \texttt{scipy.sparse.csgraph}. These functions have many options on the metrics and the algorithms used to construct the graphs; the provided function, which uses default settings, serves purely illustrative purposes. The reader is referred to the documentation for more information and is invited to consult the more sophisticated implementation in scikit-learn.

### 3.4 From MDS to LLE

The Locally Linear Embedding (LLE, Roweis and Saul (2000)) is conceptually similar to ISOMAPS, but it is a ‘local’ approach: the goal is preserving distances between neighbours and not within the whole set of snapshots. If neighbours are close enough, the geodesic distance and the Euclidean distances are similar (hence ISOMAPS reduces to MDS), and the underlying manifold can be approximated as a set of linear patches. This makes the algorithm more efficient than ISOMAP in that it avoids computing distances between points that are far apart.

The algorithm is composed of three steps. The first step is a k-nearest neighbour search like in ISOMAPS. This produces a matrix \( N_j \in \mathbb{R}^{n_s \times k} \) collecting the \( k \) nearest neighbours for the snapshot \( d_j \). The second step consists in a locally linear fit: we seek to approximate every snapshot as a linear combination of its neighbours. Therefore, we minimize the following

\[
J_1(W) = \sum_{j=1}^{n_t} ||d_j - N_j w_j||_2^2, \tag{14}
\]

where \( W \in \mathbb{R}^{n_t \times k} = [w_1, \cdots w_{n_t}]^T \) is the matrix collecting the weights for each snapshot \( w_j \) (stacked row-wise). This minimization is constrained by the condition \( 1^T w_j = 1 \), that is the sum of weights for each snapshot must be unitary. This minimization admits a closed form solution (see Ghojogh et al. (2020b)), which reads

\[
w_j = \frac{L_j^{-1}}{1^T L_j^{-1} 1} \text{ with } L_i := (d_i 1 - N_j)^T (d_i 1 - N_j) \in \mathbb{R}^{k \times k}. \tag{15}
\]

The matrix \( L_j \) is the correlation matrix centered at each snapshot \( j \) and measuring the correlation between neighbours. Its small size makes the inversion computationally irrelevant.

Finally, the last step consists in finding the reduced representation that preserves the previously derived locally linear approximation. We thus need the compressed representation \( Z_I \in \mathbb{R}^{n_r \times n_t} \) minimizing the following cost function:

\[
J_2(Z_L) = \sum_{j=1}^{n_t} ||Z_L \delta_j - Z_L w_j||_2^2 = ||Z_I - Z_I \tilde{W}||_F, \tag{16}
\]

where \( \delta_j \in \mathbb{R}^{n_t} \) is the delta vector equal to 1 at the \( j \)-th entry and zero everywhere else, \( || \cdot ||_F \) is the Frobenious norm and \( \tilde{W} \in \mathbb{R}^{n_t \times n_t} \) is the augmented matrix of weights, with entries equal to the previously computed weights \( w_j[i] \) if the snapshot \( j \) is among the neighbours of \( i \) and zero
if it is not. Also this minimization admits a closed form solution, which leads to an eigenvalue problem (see Ghojogh et al. (2020b)):

\[ MZ_L^T = Z_L^T \Lambda L \]

with \( M := (I - \tilde{W})^T(I - \tilde{W}) \in \mathbb{R}^{n_t \times n_t} \).

(17)

However, differently from the previous method, the minimization in (16) is provided by the eigenvectors linked to the smallest eigenvalues of \( M \). The following python function can be used to compute the \( n_r \)-dimensional encoding \( Z_L \) using the LLE with \( k \) nearest neighbours:

```python
def LLE_TRANS(D, n_r=3, k=5):
    # Compute k nearest neighbors distances
    nbrs_ = NearestNeighbors(n_neighbors=k+1); nbrs_.fit(D.T)
    # Compute Matrix W (and make it dense)
    W = barycenter_kneighbors_graph(nbrs_, n_neighbors=k); n_t=np.shape(D)[1]
    # Compute Matrix M = (I - W)^T(I - W) in sparse format
    M=(eye(n_t)-W).T@(eye(n_t)-W)
    # Diagonalize and Sort (use sparse eig)
    lambdas, Psi_L = eigsh(M, n_r + 1, sigma=0.0)
    # Remove the lowest eigenvalue
    return Psi_L[:,1:].T, lambdas[1:]
```

This function uses `barycenter_kneighbors_graph` from sklearn.manifold’s `locally_linear` and the functions `eye` and `eigsh` from scipy’s `sparse.linalg`. The first is used to compute the matrix \( W \) using an efficient implementation of (15). This is returned as a sparse array. `eye` creates an identity matrix as a sparse object while `eigsh` is a sparse eigenvalue solver for hermitian matrices. This solver allows to look for eigenvalues near a user defined value \( \sigma \) (in this case 0). The first eigenvalue is usually zero to machine precision and associated to a constant eigenvector; both are removed in the return line.

3.5 A note on the Decoding Process

Finding the decoding function in nonlinear manifold learning is an ill posed problem. In the vast literature on kPCA, this is known as the ‘pre-image’ problem (see Bakır et al. (2003); Bakir et al. (2004); Honeine and Richard (2011); Kwok and Tsang (2004); Mika et al. (1998); Zheng et al. (2006). The most common solution consists in using an interpolation or a regression approach. Given a set of snapshots \( d_j \in \mathbb{R}^{n_s} \) and their associated reduced representation \( z_j \in \mathbb{R}^{n_r} \), finding a function such that \( d_j \approx D(z_j) \) is a supervised learning problem for which machine learning offers an arsenal of tools. In kernel PCA, the most common is Kernel Ridge regression/interpolation (Bakır et al., 2003; Bakır et al., 2004) using the same kernel function used in the encoding. More sophisticated methods are reviewed by Honeine and Richard (2011).

In LLE, a natural approach is to use linear regression with the same weights computed during encoding process, as proposed by Saul and Roweis (2003). Given a vector \( z_l \in \mathbb{R}^{n_r} \), using a k-nearest neighbor algorithm to derive the set of weights such that \( z_l \approx P_l w_l \), with \( P_l \in \mathbb{R}^{n_r \times k} \) the matrix collecting the k-nearest neighbors of \( z_l \in \mathbb{R}^{n_r} \), the associated high dimensional counterpart (pre-image) is \( \tilde{d}_l = N_l w_l \), where \( N_l \in \mathbb{R}^{n_s \times k} \) is the matrix collecting the k nearest neighbors of \( \tilde{d}_l \in \mathbb{R}^{n_s} \). This approach was also used with excellent results by Ehlert et al. (2019). An elegant variant, based on a first order Taylor expansion of the decoder function, is presented by Farzamnik et al. (2022).

In this work, aiming at a first overview of these methods, we rely on k-nearest neighbors linear regression because it is the simplest (thus also less prone to over-fitting) and computationally cheaper. Given set of weights in (15) and given \( W \in \mathbb{R}^{n_t \times n_t} \) the associated augmented matrix of weights, the decoder is simply \( \tilde{D} = DW \). Interpolation is prevented by construction, since the diagonal entries of \( W \) are zero.

The following script provide a function to compute the decoding as a linear combination of the k nearest neighbors of each snapshot, identified from the reduced dimensional representation:
```python
def Decode(Z,D,k=10,reg=1e-6):
    # compute the W matrix
    W = barycenter_kneighbors_graph(Z, n_neighbors=k, reg=reg)
    # Compute the approximation
    D_tilde = (W @ D.T).T
    return D_tilde
```

As a note of warning, it is important to note the performance of any approximated decoding methods based on interpolation should be evaluated on data that has not been used for the encoding process. It is in fact intuitive that a compressed representation $z_l$ that is close to a vector $z_i$ computed by the encoding $z_i = \mathcal{E}(d_i)$ has a pre-image that is close to $d_i$ regardless of how well the encoding was performed. This might lead to the misleading conclusion that an autoencoder is lossless if the decoding is carried out on the same data that was encoded. To experiment with this potential problem, the reader is invited to perform kPCA and its inverse using scikit-learn and monitor the reconstruction error on the training data for different values of $n_r$.

Finally, we measure the performances of the autoencoding using two metrics, defined as

$$E = \frac{||D - \tilde{D}||_2}{||D||_2} \quad \text{and} \quad R_V = 1 - \rho^2(G, R_Z).$$

The first is the normalized $l_2$ error, which the POD/PCA seeks to minimize. The second is an adaptation of the residual variance by Tenenbaum et al. (2000), as proposed by Farzamnik et al. (2022). Here $R_Z$ denotes the matrix of Euclidean distances in the reduced domain while $\rho(A, B)$ denotes the Pearson’s correlation coefficient between two matrices $A, B$ (computed entry by entry). If $\rho \approx 0$, the Euclidean distances in the reduced space are well correlated with the geodesic distances in the original space. One could thus expect a better mapping of the underlying manifold.

## 4 Selected Test Cases

### 4.1 Test Case 1: Background Removal in PIV Images

We consider a filtering problem as a first test case. We test the nonlinear dimensionality reduction techniques described in the previous section on the background removal in PIV images. This problem is extensively described in Mendez et al. (2017), where the implementation of a POD (PCA)-based framework was first proposed for this problem. This is a special kind of denoising problem and nonlinear methods such as kPCA have been widely used for similar tasks (Honeine and Richard, 2011; Mika et al., 1998).

In the PIV background removal, the dataset at hand is a video sequence $D \in \mathbb{R}^{n_s \times n_t}$ composed of $n_t$ gray scale images of resolution $n_s$ (every image is reshaped into a column of $D$). We assume that this dataset is the sum of an ‘ideal sequence’, here denoted as $D_P$, containing the scattered light from particles, and a ‘background sequence’, here denoted as $D_B$, containing the time-varying sources of background noise (e.g. laser reflections).

In the autoencoder formulation of the denoising problem, we hope that the approximated field is $\tilde{D} \approx D_B$, such that one can filter the video sequence with a simple subtraction $D_P = D - \tilde{D}$. The rationale behind this approach is that the background noise in PIV images tends to be more correlated than PIV particles (which can be well approximated by a sequence of random patterns). Hence, an autoencoder might well approximate the background’s evolution and filter less correlated contributions. The idea is common in computer vision, where it is used to distinguish moving objects from static backgrounds (Bouwmans et al., 2020).

The test case selected in this work is the first synthetic case presented in Mendez et al. (2017), to which the reader is referred for more details. The set of images has been made available at [https://osf.io/g7asz/download](https://osf.io/g7asz/download). Briefly, this consist of a set of four sources
of background noise. A snapshot of this dataset is shown in Figure 2 on the top. The first two noise sources are triangular areas of non-uniform and time varying illumination, extending from the top corners to the bottom of the image. The third consists of vertical and horizontal thick lines featuring light reflection and flare. The last one consists of Gaussian distributed background noise. The sequence consists of \( n_t = 100 \) images with \( n_s = 1024 \times 1024 \) pixels. The images have an 8-bit dynamic range and are rescaled in the range \([0, 1]\). Although two of the noise sources have a time-resolved evolution, a random shuffle is applied to simulate a non-time resolved acquisition.

Three 64 x 64 windows are used to evaluate the filtering performances. These are marked in yellow in the snapshot in Figure 2. A zoomed view in these windows is shown in Figure 2 for both the noisy and the underlying ideal images. The first windows (W1), is influenced by both the “vertical reflection” and the light non-uniformity. Since the first reaches saturations in various snapshots, its removal is particularly challenging. This is more severe in the third window (W3) while the background removal is much easier for window (W2) which is far from saturation.

In what follows, we evaluate the performances of the autoencoders by measuring how much the filtered image resembles the ideal one. For each window, we define and error \( E_B \) as the root
mean square difference between the filtered image and the ideal one, that is

\[
E_B = \frac{1}{\sqrt{n_t n_s}} ||D_p - \tilde{D}||_2.
\]  

(19)

4.2 Test Case 2: Turbulent flow past a cylinder in transient conditions

We consider the classic flow past a cylinder as a second test case. This is arguably the most popular test case in the community of reduced order modeling of fluid flows because it is characterized by an intrinsically low dimensionality. At Reynolds numbers \( Re = U_\infty d/\nu > 47 \), with \( U_\infty \) the free stream velocity, \( d \) the cylinder diameter and \( \nu \) the kinematic viscosity, this configuration produces the well-known von Karman vortex street. This condition is characterized by a regular shedding of vortices and can be well approximated by three POD modes (Noack et al., 2003). This configuration was also used by Ehlert et al. (2019) to showcase the LLE’s capability to identify the underlying manifold in a transient condition from \( Re = 40 \) to \( Re = 120 \).

In this test case, we consider a transient condition from \( Re \approx 4000 \) to \( Re \approx 2600 \), using an experimental dataset acquired via Time-Resolved PIV. The experimental set-up and related processing is presented in Mendez et al. (2020). The dataset consists of \( n_t = 13200 \) velocity fields with \( 71 \times 30 \) vectors sampled at \( f_s = 3 \) kHz and has been made available at https://osf.io/47ftd/download. A snapshot of the velocity field is shown in Figure 3(a), together with the location of two probes (1) and (2). Figure 3(b) shows the evolution of the free stream velocity sampled at probe 1, while Figure 3(c) shows the power spectral density of the velocity magnitude in probe (2). Interested readers are referred to Mendez (2022) for a tutorial on data processing on this test case using Python.

![Figure 3: a) Snapshot of the velocity field for the second test case: the TR-PIV measurements of the flow past a cylinder. b) Evolution of the free stream velocity (in probe 1), c) power spectral density of the velocity magnitude in probe 2.](image-url)
As visible from the probe 1, the free stream velocity was varied from $U_\infty = 12\text{m/s}$ to $U_\infty = 8\text{m/s}$. Because this variation is sufficiently slow, the flow remains in quasi-steady conditions and the vortex shedding occurs at a constant Strouhal number $St = \frac{fd}{U_\infty} \approx 0.19$. Accordingly, the shedding frequency changes from $f \approx 450\text{Hz}$ to $f \approx 300\text{Hz}$ as shown in Figure 3(c).

![Image](https://example.com/image.png)

(a)

(b) [Probe 1](#)
(c) [Probe 2](#)
(d) [Probe 3](#)

Figure 4: a) Snapshot of the velocity field for the third test case: the TR-PIV measurements of an impinging gas jet. The figures b)-c)-d) show the power spectral density of the velocity magnitude in the probes 1, 2 and 3.

4.3 Test Case 3: An impinging jet flow

We consider as third test case the a flow configuration which does not have an intrinsic low dimensionality like the previous. This is the flow of an impinging planar air jet, released at a mean velocity $U_j \approx 6.5\text{m/s}$ from an opening of $H = 4\text{mm}$ at a distance of $Z = 40\text{mm}$ from a flat solid wall. This leads to a Reynolds number at the slot’s exit of $Re_j = \frac{U_j H}{\nu} \approx 1700$. The dataset was obtained via time resolved PIV in Mendez et al. (2019) and consists of $n_t = 2728$ fields with $60 \times 114$ vectors, sampled at $f_s = 2\text{kHz}$. The reader is referred to Mendez et al. (2019) for more details on the experimental set up and the related processing. The dataset has been made available at [https://osf.io/c28de/download](https://osf.io/c28de/download).

A snapshot of the velocity field is shown in Figure 4(a), together with the location of three probes. This configuration is inherently multiscale and was chosen as a benchmark for the multiscale POD (see also Barreiro-Villaverde et al. (2021); Esposito et al. (2021); Ninni and Mendez (2020)).

The Figures 4(b)-4(d) show the power spectral density of the velocity magnitude at the three probes, scaled in terms of Strouhal number $St = \frac{fH}{U_j}$. The shear layer region close to
the nozzle is characterized by the advection of roll-like vortex structures at $St \approx 0.3$ (see probe 1). The frequency of these vortices decreases as their velocity decreases while approaching the wall ($St \approx 0.17$ in probe 2). Downstream the stagnation point, in the wall jet region, the flow is much slower and characterized by large scale structures at $St < 0.1$.

5 Results

The following three subsections are dedicated to each of the three test investigated case.

5.1 Test Case 1: Background Removal Problem

We first analyze the convergence of the implemented techniques for the first test case (cf. Section 4.1). The dataset is naturally in the range $[0, 1]$, and no pre-processing is applied. It is worth recalling that the mean removal has a detrimental impact on the background removal problem, so neither the correlation matrix $K$ nor the kernel matrix $K_\xi$ are centered. On the other hand, the geodesic distance matrix in the ISOMAP was centered for numerical stability.

The convergence results are shown in Figure 5 in terms of relative $l_2$ errors on the left and residual variance $R_V$ on the right (see (18) for definitions). Both quantities are plotted as function of $n_r$, that is the dimensionality of the encoding. The PCA has the best performance according to the first while the ISOMAPS has the best performance according to the second. These results were obtained with $\gamma = 0.1$ for the kPCa and $k = 10$ for the ISOMAPs and LLE.

Concerning the $l_2$ error convergence, the slow decay is expected for the set of PIV images since most of the variance is due to the contribution of the PIV particles, which is random. Interestingly, at $n_r > 2$ the error increases with $n_r$ for the nonlinear techniques. While this might be due to limits of the simple implemented decoder, this result is particularly interesting when considering the performances in terms of residual variance $R_V$ (Figure 5(b)): this is monotonically decreasing behaviour for the ISOMAP but quickly saturates for the PCA. This shows that minimizing $E$ and minimizing $R_V$ are contrasting goals for the problem at hand. The performances of kPCA and LLE fall between the PCA and ISOMAPs, with the LLE out-performing the kPCA in the residual variance for $n_r < 4$.

In this filtering problem, it is interesting to have the approximated images as close as possible to the underlying ideal PIV sequence. We thus analyze the performances of these methods in
terms of $E_B$ (see (19) for definition). Figure 6 collects the results for all methods and the three windows in Figure 2 and different values for $\gamma$ and $k$ in the nonlinear techniques.

The figure on the top is related to the PCA, acting as a reference. As expected, the background noise can be removed more easily in window W2 than in window W3. For the PCA, the best performances are achieved using $n_r = 3$, at which all curves have a minimum. This minimum is due to the convergence of the decomposition (see also Figure 5): the lower the reconstruction error, the more the approximation resembles the provided images, including background noise and particles. In other words, at larger $n_r$, the PCA background removal becomes too aggressive, and the resulting erosion of the PIV particles outweighs the gain in
Figure 7: Zoomed view on the windows W1, W2 and W3 (see Figure 2) after background removal with PCA (top) and kPCA (bottom). The results with ISOMAPs and LLE are indistinguishable from the ones with the kPCA and are thus omitted.

background removal.

A similar trend is observed in kPCA, which also has a minimum $E_B$ at $n_r = 4$. This corresponds (see 5) to a region of rather flat convergence for both $E$ and $R_V$. Interestingly, the value of $\gamma$ does not significantly impact the performances within the investigated four orders of magnitude ($\gamma = 0.1$ to $\gamma = 100$). Finally, the performances of ISOMAPs and LLE are similar, with the first slightly better than the second. The parameter $k$ has a moderate impact. For $k = 10$ and $k = 30$, a minima appears at $n_r = 2$ and the error increases linearly for $n_r > 2$.

Taking the best configuration for each method ($n_r = 3$ for the PCA, $n_r = 3$ and $\gamma = 0.1$ for the kPCA and $n_r = 2$ and $k = 10$ for ISOMAPs and LLE) produced nearly identical results. The filtered results (to be contrasted with Figure 2) are shown in Figure 7. The results from LLE and ISOMAPs are omitted because these are indistinguishable from the results obtained via kPCA. All methods remove three of the main sources of background noise but suffer with the first (the vertical and horizontal stripes) because this saturates in various snapshots. Better removal of these stripes can be obtained if $n_r = 1$ but at the cost of worsening the performances far from these regions.

Considering that this test case is extremely challenging from a denoising point of view, these results are largely satisfactory regarding background removal. On the other hand, these results show that none of the implemented nonlinear approaches outperforms the classic PCA in the investigated background removal problem. The better performances in terms of residual variance (hence manifold mapping) are of no help in this test case.

5.2 Test Case 2: Transient Flow past a Cylinder

We here consider the second test case (cf. Section 4.2). The dataset was normalized in the range $[0, 1]$. No mean removal was performed since the flow is in transient conditions. Figure 8 shows
the convergence of the $l_2$ norm error (Fig. 8(a)) and the residual variance (Fig. 8(b)) for all the implemented methods. For the kPCA, the parameter $\gamma$ is computed by setting $\kappa_m = 0.001$ (see Section 3.2) while in the ISOMAPS and LLE we have $k = 10$. The same number of nearest neighbours is used in the linear decoder for the nonlinear methods. These parameters were selected using a grid search for hyperparameter tuning, setting the minimization of $E_B(n_r)$ at $n_r = 3$ as a goal.

The results in Figure 8 confirm that the minimization of $E_B(n_r)$ does not imply the minimization of $R_V(n_r)$ (see Sec 3.5), as also observed in the previous section. However, in this problem, the ISOMAPs outperform all decompositions in both metrics. The kPCA gives the second best performance in terms of $R_V$ convergence while the PCA reaches a plateau already at $n_r > 2$. As discussed in Section 4.2, this test case is characterized by an intrinsically low dimensionality, with $n_r = 3$ known to capture the essential features (see also Mendez et al. (2020)). It is thus interesting that both kPCA and ISOMAPs feature an ‘elbow’ at $n_r = 3$ in the convergence of the residual variance error.

The evolution of the system in the reduced space ($\mathbb{R}^3$) is shown in Figure 9 for all investigated techniques. The figures on the left show a 3D view, while the figure on the right shows the projection on the $\psi_2 - \psi_3$ plane, which clearly shows the evolution of the transient. In all figures, the markers are colored by the time axes (in seconds) to reveal the system trajectories (cf. Figure 3b). All reduced representations picture the vortex shedding as a circular orbit in the $\psi_2 - \psi_3$ plane. As time evolves and the system moves from one steady state to another, an axisymmetric trajectory connects the two orbits.

The manifold identified by the PCA appears as a truncated cone, with the shedding at higher speed (earliest times) associated with larger diameters’ orbit. This is similar for ISOMAPs, although the evolution during the transient is curved and more closely follows the smooth evolution of the free stream. In kPCA and LLE, the manifold is also close to a truncated cone but with the opposite position of the orbits (larger orbit at the slowest velocity). In the case of kPCA, the slope in the cone is much amplified compared to PCA. Although these differences might appear minor, it is worth stressing that they are linked to significantly different values of $E_B$ and $R_V$ (cf. Figure 8). To qualitatively appreciated the quality of the reconstructions, Figure 10 shows one snapshot of the flow field using approximation using $n_r = 3$ for the four approaches. The selected snapshot is the one for which the original data is shown in Figure 3(a). All reconstructions are remarkably close to the original snapshot. The PCA reconstruction
provides the smoothest fields, blurring out fine details of the vortical structures, while the nonlinear techniques recover finer flow details more closely.
Figure 9: Trajectory of in low dimensional representation of the transient turbulent cylinder flow (test case 2, described in section 5.2) with $n_r = 3$ for the PCA(POD), kPCA, ISOMAPs and LLE. The figures on the left show the 3D view; the figures on the right show the ‘lateral’ projection to highlight the system evolution during the transient.
5.3 Test Case 3: Impinging Jet Flow

We finally consider the flow field of an impinging gas jet (Sec. 4.3). Since the dataset is statistically stationary, the mean flow is subtracted prior to the dimensionality reduction and normalized by the largest velocity in the dataset. The same hyperparameters used in the previous test case are used for all methods. Figure 11 shows the convergence of $E_B(n_r)$ and $R_V(n_r)$. The multi-scale nature of this dataset makes constructing low-dimensional models considerably more difficult than in the previous test case. This is clearly visible from the slow convergence of the $l_2$ error in the PCA. Nevertheless, all nonlinear decompositions significantly outperform the PCA convergence, with ISOMAPs giving the best convergence. As expected, ISOMAPs yield the best convergence also in terms of residual variance. Both kPCA and LLE perform well in $l_2$ error convergence but poorly in residual variance convergence. This confirms again that these metrics are generally not related.

Finally, the significant gains in the $l_2$ convergence are showcased in the reconstruction of a selected snapshot, considering $n_r = 4$. This is the velocity field shown in Figure 4. Figure 12 shows the reconstruction for the PCA on the top and the reconstruction for the ISOMAPs on the bottom. In both cases, the figures are complemented with the power spectral densities of the velocity field in the three probes indicated in the figures (see also Section 4.3). The continuous blue line is used for the spectra in the reduced model, while the dashed black line is the corresponding spectra in the original field.

At $n_r = 4$, the POD modes mostly capture the vortex shedding downstream the nozzle, in the shear layer region, at a distance of about 5 to 10 mm from the wall. Consequently, the PCA reconstruction captures most of the power spectral density in probe 2. On the other hand, it misses almost completely the flow features closer to the nozzle exit (where probe 1 is
located) and offers a poor reconstruction of the wall jet region (where probe 3 is located). As a result, the flow field reconstructed via PCA appears much smoother than the original one, as only the large-scale features are retained. On the other hand, the ISOMAPs reconstruction recovers most of the features, as expected from the remarkable convergence in Figure 11, and as demonstrated by the snapshot and the power spectral density plots.

6 Conclusions and Perspectives

This work offered a concise introduction to nonlinear dimensionality techniques and the notions of autoencoders and manifold learning. Autoencoders seek to preserve as much information as possible, while manifold learning techniques seek to preserve some measure of similarity. A general mathematical framework is reviewed and used to unify the treatment of kernel Principal Component Analysis (kPCA), isometric feature learning (ISOMAPs) and Locally Linear Embedding (LLE), as well as their relation to Principal Component Analysis (PCA), known as Proper Orthogonal Decomposition (POD) in fluid dynamics. Using a k-nearest linear interpolation as decoder, these decompositions were tested in three classic data processing problems for fluid dynamics: filtering, oscillatory pattern analysis and data compression.

In the selected filtering problem, namely the background noise removal in PIV images, it was found that none of the implemented nonlinear tools outperforms the PCA/POD. While we refrain from drawing definite conclusions from a single test case, this result highlights that the idea of preserving similarity, whether globally (as in ISOMAPs) or locally (as in LLE), might not be relevant to a filtering process.

In the oscillatory pattern detection problem, namely the dimensionality reduction of a transient cylinder wake flow, it was found that ISOMAPs and PCA have comparable $l_2$ reconstruction error if the reduced dimension is $n_r = 3$, as it is customarily done for this kind of flows. The resulting manifolds are qualitatively similar. Both are surfaces of revolution, but the generating function is approximately linear in the PCA (hence generating a conical frustum) and approximately sigmoidal in the ISOMAPs. The ISOMAPs outperform the PCA in the residual variance convergence; thus, one might conclude that the identified manifold is closer to the ‘true’ underlying one. The performances of kPCA and LLE are comparable (they outperform the PCA in residual variance but not in the reconstruction error) and result in similar manifolds:
a conical frustum with opposite orientation to the ones in PCA and ISOMAPs. This shows that a ‘small’ difference in the implemented metrics can lead to significant differences in the system trajectory in the reduced space. One should keep this in mind when analyzing the topology of low-dimensional manifolds derived from data.

Finally, in the investigated data compression problem, which considered the flow of an impinging gas jet, all nonlinear methods significantly outperformed the PCA (POD) in terms of $l_2$ error convergence. This result is remarkable considering the widely different scales characterizing this test case and the poor convergence of the PCA. The ISOMAPs and LLE bring the convergence error below 40% at $n_r = 10$. Inspecting the reconstruction of the flow field and the power spectral density in three selected probes showed that the ISOMAPs preserve most of the relevant features even at $n_r = 4$.

As a general conclusion, it is worth highlighting that all the analyzed test cases show that the minimization of the $l_2$ reconstruction error and the residual variance are vastly different (if not contrasting) objectives, and methods that excel in one metric might fail in the other. Nevertheless, the ISOMAPs show the best overall performance among the implemented methods. In the author’s opinion, whether this method will reach the ubiquity of PCA/POD depends on how much we can adapt its formalism to (and our interpretation of) classic problems.

In a filtering problem, for example, the better convergence of ISOMAPs implies that this is also more prone to retain noise in the autoencoded result. A multi-scale extension of the problem, whereby the ISOMAPs is used at different scales, could lead to a powerful filtering approach. In the pattern recognition problem, reduced representations from ISOMAPs or kPCA can reveal essential dynamics, but their mapping to the space domain is conceptually much more involved than the simple projection in a linear approach such as PCA.

The nonlinearity of these methods brings an important conceptual shift, and the familiar notions of modes, orthogonality and Galerkin projection are no longer relevant. Besides complicating the spatial localization of the essential dynamics, this significantly challenges the construction of ‘intrusive’ low dimensional models, i.e. the integration of low order representation from data with the governing equations. Finally, an important line of research is the decoding problem, with particular emphasis on the interpolative nature of methods such as the k-nearest linear approach used in this work or the regression techniques like the kernel Ridge regression encountered in the literature of kPCA. While this work focused on ‘in-sample’ error in the reconstruction (i.e. decoding the same data that was used to train the encoder), the robustness of the ‘out-of-sample’ reconstruction (i.e. decoding new data) remains to be assessed in problems of interest for fluid dynamics.

These exciting problems will surely attract and challenge many enthusiastic fluid dynamists to continue erasing the boundaries between ordinary data processing and machine learning.
Figure 12: Reconstruction of the TR-PIV measurements for the third test case (Section 4.3) with $n_r = 4$ using PCA (top panel) and ISOMAPs (bottom panel). These figures should be confronted with Figure 4, which shows the original snapshot. The plots under the velocity fields show the power spectral density of the velocity magnitude in the three probes (marked in the snapshots). The blue continuous lines refer to the reconstructed fields; black dashed lines refer to the original field.
References

Lionel Agostini. Exploration and prediction of fluid dynamical systems using auto-encoder technology. *Physics of Fluids*, 32(6):067103, jun 2020. doi: 10.1063/5.0012906.

Shady E. Ahmed, Suraj Pawar, Omer San, Adil Rasheed, Traian Iliescu, and Bernd R. Noack. On closures for reduced order models—a spectrum of first-principle to machine-learned avenues. *Physics of Fluids*, 33(9):091301, 2021. doi: 10.1063/5.0061577.

Guillaume Alain and Yoshua Bengio. What regularized auto-encoders learn from the data generating distribution, 2012.

Ethem Alpaydin. *Introduction to Machine Learning*. Adaptive Computation and Machine Learning series. MIT Press, London, England, 4 edition, March 2020.

N. Aubry. On the hidden beauty of the proper orthogonal decomposition. *Theor. Comput. Fluid Dyn.*, 2(5-6):339–352, aug 1991. doi: 10.1007/bf00271473.

Nadine Aubry, Régis Guyonnet, and Ricardo Lima. Spatiotemporal analysis of complex signals: Theory and applications. *Journal of Statistical Physics*, 64(3-4):683–739, aug 1991. doi: 10.1007/bf01048312.

Shaeela Ayesha, Muhammad Kashif Hanif, and Ramzan Talib. Overview and comparative study of dimensionality reduction techniques for high dimensional data. *Information Fusion*, 59: 44–58, jul 2020. doi: 10.1016/j.inffus.2020.01.005.

Gökhan H. Bakır, Jason Weston, and Bernhard Schölkopf. Learning to find pre-images. In *Proceedings of the 16th International Conference on Neural Information Processing Systems*, NIPS’03, page 449–456, Cambridge, MA, USA, 2003. MIT Press.

Gökhan Bakır, Bernhard Schölkopf, and Jason Weston. On the pre-image problem in kernel methods. In *Kernel Methods in Bioengineering, Signal and Image Processing*, pages 284–302. IGI Global, 2007. doi: 10.4018/978-1-59904-042-4.ch012.

Gökhan H. Bakır, Alexander Zien, and Koji Tsuda. Learning to find graph pre-images. In *Lecture Notes in Computer Science*, pages 253–261. Springer Berlin Heidelberg, 2004. doi: 10.1007/978-3-540-28649-3_31.

Pierre Baldi and Kurt Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, 2(1):53–58, jan 1989. doi: 10.1016/0893-6080(89)90014-2.

David Barreiro-Villaverde, Anne Gosset, and Miguel A. Mendez. On the dynamics of jet wiping: Numerical simulations and modal analysis. *Physics of Fluids*, 33(6):062114, jun 2021. doi: 10.1063/5.0051451.

Peter Benner, editor. *Snapshot-based methods and algorithms*. De Gruyter, Berlin, Germany, December 2020.

Peter Benner, Serkan Gugercin, and Karen Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM Review*, 57(4):483–531, jan 2015. doi: 10.1137/130932715.

G Berkooz, P Holmes, and J L Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics*, 25(1):539–575, jan 1993. doi: 10.1146/annurev.fl.25.010193.002543.
Weitao Bi, Yasuhiko Sugii, Koji Okamoto, and Haruki Madarame. Time-resolved proper or-

thogonal decomposition of the near-field flow of a round jet measured by dynamic particle

image velocimetry. Measurement Science and Technology, 14(8):L1–L5, jul 2003. doi:

10.1088/0957-0233/14/8/101.

Christopher M. Bishop. Pattern Recognition and Machine Learning. Springer-Verlag New York

Inc., April 2011. ISBN 0387310738.

Elkhadim Bouhoubeiny and Philippe Druault. Note on the POD-based time interpolation from

successive PIV images. Comptes Rendus Mécanique, 337(11-12):776–780, nov 2009. doi:

10.1016/j.crme.2009.10.003.

J. A. Bourgeois, B. R. Noack, and R. J. Martinuzzi. Generalized phase average with applications
to sensor-based flow estimation of the wall-mounted square cylinder wake. Journal of Fluid

Mechanics, 736:316–350, nov 2013. doi: 10.1017/jfm.2013.494.

Thierry Bouwmans, Necdet Serhat Aybat, and El hadi Zahzah. Handbook of Robust Low-Rank

and Sparse Matrix Decomposition. Chapman and Hall/CRC, 2020.

L. D. C. Casa and P. S. Krueger. Radial basis function interpolation of unstructured, three-
dimensional, volumetric particle tracking velocimetry data. Measurement Science and Tech-
nology, 24(6):065304, may 2013. doi: 10.1088/0957-0233/24/6/065304.

Alejandro Castillo and Arturo Roman Messina. Data-driven sensor placement for state recon-

struction via POD analysis. IET Generation, Transmission & Distribution, 14(4):656–664, jan 2020. doi: 10.1049/iet-gtd.2019.0199.

John J Charonko, Cameron V King, Barton L Smith, and Pavlos P Vlachos. Assessment of pressure field calculations from particle image velocimetry measurements. Measurement Science and Technology, 21(10):105401, aug 2010. doi: 10.1088/0957-0233/21/10/105401.

C. Chen and L. He. On locally embedded two-scale solution for wall-bounded turbulent flows. Journal of Fluid Mechanics, 933, jan 2022. doi: 10.1017/jfm.2021.1075.

H. Choi and S. Choi. Kernel isomap. Electronics Letters, 40(25):1612, 2004. doi: 10.1049/el:

20046791.

J. H. Citriniti and W. K. George. Reconstruction of the global velocity field in the axisymmetric

mixing layer utilizing the proper orthogonal decomposition. Journal of Fluid Mechanics, 418:

137–166, sep 2000. doi: 10.1017/s0022112000001087.

Soledad Le Clainche and José M. Vega. Higher order dynamic mode decomposition to identify and extrapolate flow patterns. Physics of Fluids, 29(8):084102, aug 2017. doi: 10.1063/1.4997206.

L. Cordier and M. Bergmann. Proper orthogonal decomposition: an overview. In L. David and C. Schram, editors, Advanced Post Processing of Experimental and Numerical Data, volume VKI-LS. von Karman Institute for Fluid Dynamics, 2013.

Michael A. A. Cox and Trevor F. Cox. Multidimensional Scaling, pages 315–347. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. ISBN 978-3-540-33037-0. doi: 10.1007/978-3-540-33037-0_14.

Hunor Csala, Scott T. M. Dawson, and Amirhossein Arzani. Comparing different nonlinear dimensionality reduction techniques for data-driven unsteady fluid flow modeling. Physics of Fluids, 34(11):117119, nov 2022. doi: 10.1063/5.0127284.
Arthur Ehlert, Christian N. Nayeri, Marek Morzynski, and Bernd R. Noack. Locally linear embedding for transient cylinder wakes. June 2019.

Hamidreza Eivazi, Hadi Veisi, Mohammad Hossein Naderi, and Vahid Esfahanian. Deep neural networks for nonlinear model order reduction of unsteady flows. *Physics of Fluids*, 32(10):105104, 2020. doi: 10.1063/5.0020526.

C. Esposito, M.A. Mendez, J. Steelant, and M.R. Vetrano. Spectral and modal analysis of a cavitating flow through an orifice. *Experimental Thermal and Fluid Science*, 121:110251, February 2021. doi: 10.1016/j.expthermflusci.2020.110251.

Ehsan Farzannik, Andrea Ianiro, Stefano Discetti, Nan Deng, Kilian Oberleithner, Bernd R. Noack, and Vanesa Guerrero. From snapshots to manifolds - a tale of shear flows. March 2022.

Kai Fukami, Taichi Nakamura, and Koji Fukagata. Convolutional neural network based hierarchical autoencoder for nonlinear mode decomposition of fluid field data. *Physics of Fluids*, 32(9):095110, September 2020. doi: 10.1063/5.0020721.

Kai Fukami, Kazuto Hasegawa, Taichi Nakamura, Masaki Morimoto, and Koji Fukagata. Model order reduction with neural networks: Application to laminar and turbulent flows. *SN Computer Science*, 2(6), September 2021. doi: 10.1007/s42979-021-00867-3.

William K. George. A 50-year retrospective and the future. In *Whither Turbulence and Big Data in the 21st Century?*, pages 13–43. Springer International Publishing, August 2016. doi: 10.1007/978-3-319-41217-7_2.

Benyamin Ghojogh and Mark Crowley. Unsupervised and supervised principal component analysis: Tutorial. June 2019.

Benyamin Ghojogh, Fakhri Karray, and Mark Crowley. Eigenvalue and generalized eigenvalue problems: Tutorial. March 2019a.

Benyamin Ghojogh, Maria N. Samad, Sayema Asif Mashhadi, Tania Kapoor, Wahab Ali, Fakhri Karray, and Mark Crowley. Feature selection and feature extraction in pattern analysis: A literature review. May 2019b.

Benyamin Ghojogh, Ali Ghodsi, Fakhri Karray, and Mark Crowley. Multidimensional scaling, sammon mapping, and isomap: Tutorial and survey. September 2020a.

Benyamin Ghojogh, Ali Ghodsi, Fakhri Karray, and Mark Crowley. Locally linear embedding and its variants: Tutorial and survey. November 2020b.

Benyamin Ghojogh, Ali Ghodsi, Fakhri Karray, and Mark Crowley. Unified framework for spectral dimensionality reduction, maximum variance unfolding, and kernel learning by semidefinite programming: Tutorial and survey. June 2021.

Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016.

S. V. Gordeyev and F. O. Thomas. Coherent structure in the turbulent planar jet. part 1. extraction of proper orthogonal decomposition eigenmodes and their self-similarity. *Journal of Fluid Mechanics*, 414:145–194, 2000. doi: 10.1017/S002211200000848X.

Luca Guastoni, Alejandro Güemes, Andrea Ianiro, Stefano Discetti, Philipp Schlatter, Hossein Azizpour, and Ricardo Vinuesa. Convolutional-network models to predict wall-bounded turbulence from wall quantities. *Journal of Fluid Mechanics*, 928:A27, 2021. doi: 10.1017/jfm.2021.812.
N. Halko, P. G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review*, 53(2):217–288, jan 2011. doi: 10.1137/090771806.

Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Halladane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

K. Hasselmann. PIPs and POPs: The reduction of complex dynamical systems using principal interaction and oscillation patterns. *Journal of Geophysical Research*, 93(D9):11015, 1988. doi: 10.1029/jd093id09p11015.

J E Higham, W Brevis, and C J Keylock. A rapid non-iterative proper orthogonal decomposition based outlier detection and correction for PIV data. *Measurement Science and Technology*, 27(12):125303, oct 2016. doi: 10.1088/0957-0233/27/12/125303.

Philip Holmes, John L. Lumley, and Gal Berkooz. *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge University Press, 1996. doi: 10.1017/cbo9780511622700.

Paul Honeine and Cedric Richard. Preimage problem in kernel-based machine learning. *IEEE Signal Processing Magazine*, 28(2):77–88, mar 2011. doi: 10.1109/msp.2010.939747.

Mihailo R. Jovanović, Peter J. Schmid, and Joseph W. Nichols. Sparsity-promoting dynamic mode decomposition. *Physics of Fluids*, 26(2):024103, feb 2014. doi: 10.1063/1.4863670.

S. Karri, J. Charonko, and P. P. Vlachos. Robust wall gradient estimation using radial basis functions and proper orthogonal decomposition (POD) for particle image velocimetry (PIV) measured fields. *Measurement Science and Technology*, 20(4):045401, feb 2009. doi: 10.1088/0957-0233/20/4/045401.

J Kriegseis, T Dehler, M Gnirß, and C Tropea. Common-base proper orthogonal decomposition as a means of quantitative data comparison. *Measurement Science and Technology*, 21(8):085403, jul 2010. doi: 10.1088/0957-0233/21/8/085403.

J. Nathan Kutz, Xing Fu, and Steven L. Brunton. Multiresolution dynamic mode decomposition. *SIAM Journal on Applied Dynamical Systems*, 15(2):713–735, jan 2016. doi: 10.1137/15m1023543.

J.T.-Y. Kwok and I.W.-H. Tsang. The pre-image problem in kernel methods. *IEEE Transactions on Neural Networks*, 15(6):1517–1525, nov 2004. doi: 10.1109/tnn.2004.837781.

John Aldo Lee, Amaury Lendasse, and Michel Verleysen. Curvilinear distance analysis versus isomap. In *European Symposium on Artificial Neural Networks*, pages 185–192, Bruges (Belgium), 2002.

Jean-Christophe Loiseau, Bernd R. Noack, and Steven L. Brunton. Sparse reduced-order modelling: sensor-based dynamics to full-state estimation. *Journal of Fluid Mechanics*, 844:459–490, apr 2018. doi: 10.1017/jfm.2018.147.

John L. Lumley. *Stochastic Tools in Turbulence*. DOVER PUBN INC, 1970. ISBN 0486462706.

John L. Lumley and Andrew Poje. Low-dimensional models for flows with density fluctuations. *Physics of Fluids*, 9(7):2023–2031, jul 1997. doi: 10.1063/1.869321.
M. A. Mendez, M. Balabane, and J.-M. Buchlin. Multi-scale proper orthogonal decomposition of complex fluid flows. *Journal of Fluid Mechanics*, 870:988–1036, may 2019. doi: 10.1017/jfm.2019.212.

M A Mendez, D Hess, B B Watz, and J-M Buchlin. Multiscale proper orthogonal decomposition (mPOD) of TR-PIV data—a case study on stationary and transient cylinder wake flows. *Measurement Science and Technology*, 31(9):094014, jun 2020. doi: 10.1088/1361-6501/ab82be.

M.A. Mendez, M. Raiola, A. Masullo, S. Discetti, A. Ianiro, R. Theunissen, and J.-M. Buchlin. Pod-based background removal for particle image velocimetry. *Experimental Thermal and Fluid Science*, 80:181–192, 2017. ISSN 0894-1777. doi: https://doi.org/10.1016/j.expthermflusci.2016.08.021.

M.A. Mendez, M.T. Scelzo, and J.-M. Buchlin. Multiscale modal analysis of an oscillating impinging gas jet. *Experimental Thermal and Fluid Science*, 91:256–276, feb 2018. doi: 10.1016/j.expthermflusci.2017.10.032.

Miguel Alfonso Mendez. Statistical treatment, fourier and modal decomposition. *VKI Lecture Series ‘Fundamentals and recent advances in Particle Image Velocimetry and Lagrangian Particle Tracking’, ISBN 978-2-87516-181-9*, January 2022.

Sebastian Mika, Bernhard Schölkopf, Alex Smola, Klaus-Robert Müller, Matthias Scholz, and Gunnar Rätsch. Kernel pca and de-noising in feature spaces. In M. Kearns, S. Solla, and D. Cohn, editors, *Advances in Neural Information Processing Systems*, volume 11. MIT Press, 1998.

Michele Milano and Petros Koumoutsakos. Neural network modeling for near wall turbulent flow. *Journal of Computational Physics*, 182(1):1–26, oct 2002. doi: 10.1006/jcph.2002.7146.

Kevin P. Murphy. *Machine Learning*. MIT Press Ltd, August 2012. ISBN 0262018020.

Davide Ninni and Miguel A. Mendez. MODULO: A software for multiscale proper orthogonal decomposition of data. *SoftwareX*, 12:100622, jul 2020. doi: 10.1016/j.softx.2020.100622.

Bernd R. Noack, Konstantin Afanasiev, Marek Morzyński, Gilead Tadmor, and Frank Thiele. A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *Journal of Fluid Mechanics*, 497:335–363, dec 2003. doi: 10.1017/s0022112003006694.

Bernd R. Noack, Witold Stankiewicz, Marek Morzyński, and Peter J. Schmid. Recursive dynamic mode decomposition of transient and post-transient wake flows. *Journal of Fluid Mechanics*, 809:843–872, 2016. doi: 10.1017/jfm.2016.678.

S. Pawar, S. M. Rahman, H. Vaddireddy, O. San, A. Rasheed, and P. Vedula. A deep learning enabler for nonintrusive reduced order modeling of fluid flows. *Physics of Fluids*, 31(8):085101, 2019. doi: 10.1063/1.5113494.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Cécile Penland. A stochastic model of IndoPacific sea surface temperature anomalies. *Physica D: Nonlinear Phenomena*, 98(2–4):534–558, nov 1996. doi: 10.1016/0167-2789(96)00124-8.

Andrew Pollard, Luciano Castillo, Luminita Danaila, and Mark Glauser, editors. *Whither Turbulence and Big Data in the 21st Century?* Springer International Publishing, 2017. doi: 10.1007/978-3-319-41217-7.
Marco Raiola, Stefano Discetti, and Andrea Ianiro. On PIV random error minimization with optimal POD-based low-order reconstruction. *Experiments in Fluids*, 56(4), mar 2015. doi: 10.1007/s00348-015-1940-8.

Manuel Ratz, Domenico Fiorini, Alessia Simonini, Christian Cierpka, and Miguel A. Mendez. Analysis of an unsteady quasi-capillary channel flow with time-resolved PIV and RBF-based super-resolution. *Journal of Coatings Technology and Research*, aug 2022. doi: 10.1007/s11998-022-00664-4.

Ronald L. Rivest, Thomas H. (Dartmouth College) Cormen, Charles E. (MIT) Leiserson, and Clifford (Columbia University) Stein. *Introduction to Algorithms*. MIT Press Ltd, July 2009. ISBN 0262033844.

Sam T. Roweis and Lawrence K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000. doi: 10.1126/science.290.5500.2323.

C.W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D.S. Henningson. Spectral analysis of nonlinear flows. *J. Fluid Mech.*, 641:115, nov 2009. doi: 10.1017/s0022112009992059.

Pankaj Saini, Christoph M. Arndt, and Adam M. Steinberg. Development and evaluation of gappy-POD as a data reconstruction technique for noisy PIV measurements in gas turbine combustors. *Experiments in Fluids*, 57(7), jul 2016. doi: 10.1007/s00348-016-2208-7.

Lawrence K. Saul and Sam T. Roweis. An introduction to locally linear embedding. 2001.

Lawrence K. Saul and Sam T. Roweis. Think globally, fit locally: unsupervised learning of low dimensional manifolds. *The Journal of Machine Learning Research*, 4:119–155, 2003. doi: 10.1162/153244304322972667.

P.J. Schmid. Dynamic mode decomposition of numerical and experimental data. *J. Fluid Mech.*, 656:5–28, jul 2010. doi: 10.1017/jfm.2010.1217.

Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Kernel principal component analysis. In *Lecture Notes in Computer Science*, pages 583–588. Springer Berlin Heidelberg, 1997. doi: 10.1007/bfb0020217.

Onofrio Semeraro, Gabriele Bellani, and Fredrik Lundell. Analysis of time-resolved PIV measurements of a confined turbulent jet using POD and Koopman modes. *Experiments in Fluids*, 53(5):1203–1220, jul 2012. doi: 10.1007/s00348-012-1354-9.

Zhaoyang Shen, Zhanqun Shi, Guoji Shen, Dong Zhen, Fengshou Gu, and Andrew Ball. Informative singular value decomposition and its application in fault detection of planetary gearbox. *Measurement Science and Technology*, 33(8):085010, May 2022. doi: 10.1088/1361-6501/ac69b0.

Moritz Sieber, C. Oliver Paschereit, and Kilian Oberleithner. Spectral proper orthogonal decomposition. *Journal of Fluid Mechanics*, 792:798–828, 2016. doi: 10.1017/jfm.2016.103.

Jaskaran Singh, Moslem Azamfar, Fei Li, and Jay Lee. A systematic review of machine learning algorithms for prognostics and health management of rolling element bearings: fundamentals, concepts and applications. *Measurement Science and Technology*, 32(1):012001, jan 2020. doi: 10.1088/1361-6501/ab8df9.

L. Sirovich. Turbulence and the dynamics of coherent structures: Part i. coherent structures. *Quart. Appl. Math.*, 45(3):561–571, 1987. doi: https://doi.org/10.1090/qam/910462.
L. Sirovich. Chaotic dynamics of coherent structures. *Physica D, 37*(1):126–145, 1989. doi: https://doi.org/10.1016/0167-2789(89)90123-1.

L. Sirovich. Analysis of turbulent flows by means of the empirical eigenfunctions. *Fluid Dyn. Res.*, 8:85–100, 1991. doi: https://doi.org/10.1016/0169-5983(91)90033-F.

P Sullivan and A Pollard. Coherent structure identification from the analysis of hot-wire data. *Measurement Science and Technology, 7*(10):1498–1516, October 1996. doi: 10.1088/0957-0233/7/10/020.

K. Taira, S. L. Brunton, S. T. M. Dawson, C. W. Rowley, T. Colonius, B. J. McKeon, O. T. Schmidt, S. Gordeyev, V. Theofilis, and L. S. Ukeiley. Modal analysis of fluid flows: An overview. *AIAA J.*, 55(12):4013–4041, dec 2017. doi: 10.2514/1.j056060.

Kunihiko Taira, Maziar S. Hemati, Steven L. Brunton, Yi Yang Sun, Karthik Duraisamy, Shervin Bagheri, Scott T. M. Dawson, and Chi-An Yeh. Modal analysis of fluid flows: Applications and outlook. *AIAA Journal*, 58(3):998–1022, mar 2020. doi: 10.2514/1.j058462.

Flavia Tauro, Salvatore Grimaldi, and Maurizio Porfiri. Unraveling flow patterns through nonlinear manifold learning. *PLoS ONE*, 9(3):e91131, mar 2014. doi: 10.1371/journal.pone.0091131.

Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, dec 2000. doi: 10.1126/science.290.5500.2319.

Warren S. Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17(4):401–419, dec 1952. doi: 10.1007/bf02288916.

Aaron Towne, Oliver T. Schmidt, and Tim Colonius. Spectral proper orthogonal decomposition and its relationship to dynamic mode decomposition and resolvent analysis. *Journal of Fluid Mechanics*, 847:821–867, 2018. doi: 10.1017/jfm.2018.283.

Jonathan H. Tu, Clarence W. Rowley, Dirk M. Luchtenburg, Steven L. Brunton, and J. Nathan Kutz. On dynamic mode decomposition: Theory and applications. *Journal of Computational Dynamics*, 1(2):391–421, dec 2014. doi: 10.3934/jcd.2014.1.391.

S. Vellangiri, S. Alagumuthukrishnan, and S Iwin Thankumar joseph. A review of dimensionality reduction techniques for efficient computation. *Procedia Computer Science*, 165:104–111, 2019. doi: 10.1016/j.procs.2020.01.079.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denís Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Hans von Storch and Jinsong Xu. Principal oscillation pattern analysis of the 30- to 60-day oscillation in the tropical troposphere. *Climate Dynamics*, 4(3):175–190, sep 1990. doi: 10.1007/bf00209520.
Zhenjian Yao, Zhongyu Wang, Jeffrey Yi-Lin Forrest, Qiyue Wang, and Jing Lv. Empirical mode decomposition-adaptive least squares method for dynamic calibration of pressure sensors. *Measurement Science and Technology*, 28(4):045010, feb 2017. doi: 10.1088/1361-6501/aa5c25.

Nanning Zheng and Jianru Xue. Manifold learning. In *Statistical Learning and Pattern Analysis for Image and Video Processing*, pages 87–119. Springer London, 2009. doi: 10.1007/978-1-84882-312-9_4.

Wei-Shi Zheng, Jian-Huang Lai, and P.C. Yuen. Weakly supervised learning on pre-image problem in kernel methods. In *18th International Conference on Pattern Recognition (ICPR’06)*. IEEE, 2006. doi: 10.1109/icpr.2006.1187.