Julien Weiss

A Tutorial on the Proper Orthogonal Decomposition

Weiss, Julien: A Tutorial on the Proper Orthogonal Decomposition. In: 2019 AIAA Aviation Forum. 17–21 June 2019, Dallas, Texas, United States.

A Tutorial on the Proper Orthogonal Decomposition

Julien Weiss*

Technical University of Berlin, 10587 Berlin, Germany

This tutorial introduces the Proper Orthogonal Decomposition (POD) to engineering students and researchers interested in its use in fluid dynamics and aerodynamics. The objectives are firstly to give an intuitive feel for the method and secondly to provide example MATLAB codes of common POD algorithms. The discussion is limited to the finite-dimensional case and only requires knowledge of basic statistics and matrix algebra. The POD is first introduced with a two-dimensional example in order to illustrate the different projections that take place in the decomposition. The n-dimensional case is then developed using experimental data obtained in a turbulent separation-bubble flow and numerical results from simulations of a cylinder wake flow.

I. Introduction

The Proper Orthogonal Decomposition (POD) originates from the field of turbulence. It was introduced to the fluid-dynamics community by Lumley in 1967 [1] as an attempt to decompose the random vector field representing turbulent fluid motion into a set of deterministic functions that each capture some portion of the total fluctuating kinetic energy in the flow. The hope was that a limited number of these deterministic functions – the POD modes – can give the analyst some idea about the organization of the flow. In other words, the POD should help us find the elusive coherent structures that populate turbulent flows but yet are surprisingly difficult to define and observe.

Let us write the fluctuating velocity in a flow $\boldsymbol{u'}(\mathbf{x},t)$, where $\boldsymbol{u'}$ is the velocity vector $\boldsymbol{U}$ minus its temporal mean $\overline{\boldsymbol{U}}$ (we are assuming a statistically stationary flow with a defined temporal mean). Our notations are $\mathbf{x}=(x,y,z)$ for the position vector, $\boldsymbol{U}=(U,V,W)$ for the velocity vector, and t for time. The idea behind the POD is to decompose the random vector field $\boldsymbol{u'}(\mathbf{x},t)$ into a set of deterministic spatial functions $\boldsymbol{\Phi}_k(\mathbf{x})$ modulated by random time coefficients $a_k(t)$ so that:

$$\boldsymbol{u'}(\mathbf{x},t)=\sum_{k=1}^{\infty}a_k(t)\boldsymbol{\Phi}_k(\mathbf{x}). \tag{1}$$

The $\boldsymbol{\Phi}_k(\mathbf{x})$ are the POD (spatial) modes and the $a_k(t)$ are their time coefficients.

In practice, there are many ways to write a decomposition like Eq. 1, the familiar Fourier decomposition being another well-know method. The P in POD comes from the fact that the POD is 'proper', or optimal, inasmuch as the sequence $\sum_{k=1}^{n}a_k(t)\boldsymbol{\Phi}_k(\mathbf{x})$ maximizes the kinetic energy that can be captured by the first n spatial modes. The O in POD comes from the fact that the modes are orthonormal, which means that in a suitable function space one can write:

$$\iiint_{\mathbf{x}}\boldsymbol{\Phi}_{k_1}(\mathbf{x})\boldsymbol{\Phi}_{k_2}(\mathbf{x})d\mathbf{x}=\begin{cases}1 \text{ if } k_1=k_2\\ 0 \text{ if } k_1\neq k_2\end{cases}. \tag{2}$$

The orthonormality property is quite useful since it implies that each time coefficient $a_k(t)$ only depends on the spatial mode $\boldsymbol{\Phi}_k(\mathbf{x})$. Indeed, by multiplying Eq. 1 with $\boldsymbol{\Phi}_k(\mathbf{x})$ and integrating over space one obtains:

$$a_k(t)=\iiint_{\mathbf{x}}\boldsymbol{u'}(\mathbf{x},t)\boldsymbol{\Phi}_k(\mathbf{x})d\mathbf{x}. \tag{3}$$

*Professor, Chair of Aerodynamics, Associate Fellow AIAA.

There are several very good overviews of the POD in the fluid-dynamics literature (*e.g.* [2, 3, 4, 5, 6, 7]). With the exception of [3] and [7], these references describe the problem in the infinite-dimensional space of continuous functions of space and time, as in Eqs. 2 and 3. While the mathematics are not new, they are not usually part of the curriculum followed by mechanical or aeronautical engineering students. On the other hand, these students have normally been introduced to finite-dimensional matrix algebra. In the field of statistics, POD is called Principal Component Analysis (PCA) and is typically introduced with finite-dimensional data, first with only two variables for illustrative purposes and then extending to the n-dimensional case (*e.g.* [8]). The author believes that this approach is better suited to engineering students for two reasons: first, because the mathematics are more in line with what they learn in school, and second, because experimental and numerical data are always finite-dimensional. Thus, the purpose of this tutorial is to give an introduction to the POD that follows the usual PCA presentation, but specifically targeted to the fields of fluid mechanics and aerodynamics. The goal is not to demonstrate theoretical results, but to provide an intuitive feel for the method and a description of common POD algorithms. This approach was inspired by the papers of Shlens [9], Smith [10], Chatterjee [3], Chen *et al.* [11], and the books by Jackson [8] and Kutz [12].

II. Example Flow

In this tutorial, we will illustrate the POD through its computation on the velocity fields measured experimentally by particle image velocimetry (PIV) in a turbulent separation-bubble flow (TSB). The general flow geometry is illustrated in Fig. 1 and was investigated in detail in [13]. An incoming turbulent boundary layer separates because of an adverse pressure gradient and reattaches further downstream because of a favorable pressure gradient. What we call a TSB is a recirculation zone that is observed on the ceiling of the wind tunnel and that is bounded by a shear layer. The PIV field of view is a 200 mm $\times$ 80 mm vertical rectangle roughly centered at $x = 1825$ mm and $z = 0$ mm.

The PIV data consists of $N_t = 3580$ vector fields of $129(N_x) \times 45(N_y)$ velocity vectors (U, V) measured on a Cartesian grid in the (x, y) plane. The sampling frequency is 900 Hz. For the sake of simplicity we will only consider the longitudinal velocity U in the calculations. Fig. 2(top) shows an example of instantaneous contour plot of the longitudinal velocity field obtained from PIV (one specific snapshot). It can be seen that the flow is indeed highly unsteady. Fig. 2(bottom) shows the corresponding time-averaged, longitudinal velocity field with a few representative mean streamlines.[a]

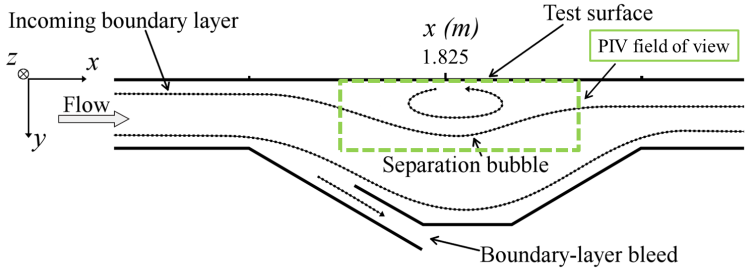

Figure 1. Experimental set-up of the TSB flow.

III. The 2-Dimensional Example

Let's start by analyzing the data obtained at two positions (a) and (b) within our separation bubble. These two points are illustrated by the two small circles in Fig. 2(bottom). Our velocity data thus consists of two arrays of $m = N_t = 3580$ longitudinal velocity values ($U_a(t_i)$ and $U_b(t_i)$) that we can concatenate into one $m \times 2$ matrix $\mathbf{S}$, which we will call the matrix of snapshots:

[a]The PIV data is available as MATLAB workspace at `http://dx.doi.org/10.14279/depositonce-8447`.

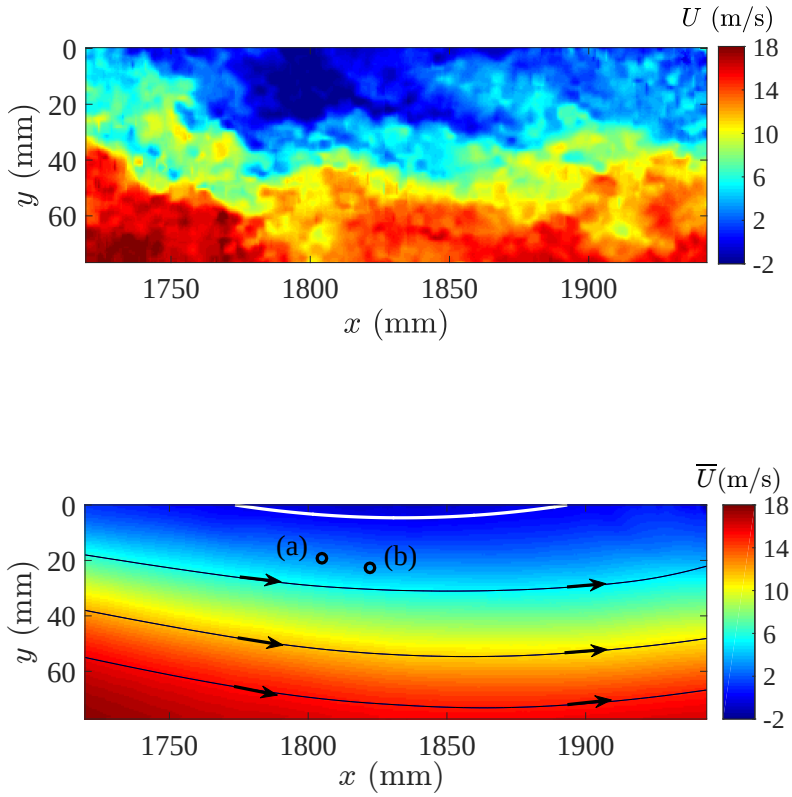

Figure 2. Top: Example of instantaneous longitudinal velocity field in m/s. Bottom: Time-averaged, longitudinal velocity field in m/s. The white line approximately delimits the region of negative mean velocity. The black lines are three representative mean streamlines. The two circles denote the measurement points used in Section III.

$$\mathbf{S} = \begin{pmatrix} U_a(t_1) & U_b(t_1) \\ U_a(t_2) & U_b(t_2) \\ \vdots & \vdots \\ U_a(t_m) & U_b(t_m) \end{pmatrix}. \tag{4}$$

Since we are mostly interested in the flow dynamics, we start by removing the average velocities $\overline{U}_a$ and $\overline{U}_b$ from their respective columns in order to get a new snapshot matrix consisting only of the velocity fluctuations $u'_a(t) = U_a(t) - \overline{U}_a$ and $u'_b(t) = U_b(t) - \overline{U}_b$. We call this new matrix $\mathbf{U}$:

$$\mathbf{U} = \begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \\ \vdots & \vdots \\ u_{m1} & u_{m2} \end{pmatrix} = \begin{pmatrix} u'_a(t_1) & u'_b(t_1) \\ u'_a(t_2) & u'_b(t_2) \\ \vdots & \vdots \\ u'_a(t_m) & u'_b(t_m) \end{pmatrix}. \tag{5}$$

The time traces of u'_a and u'_b are shown in Fig. 3 for an arbitrary time interval. Since we are looking at a turbulent flow, the two signals are essentially random. However, because positions (a) and (b) are not too far apart, the traces tend to follow each other. In other words, there is some correlation between the two velocity signals.

We can also look at our data on a 2D plot where the x-axis represents the magnitude of u'_a and the y-axis that of u'_b. Each row of $\mathbf{U}$ is a point on this plane and when we plot all the rows, what we typically

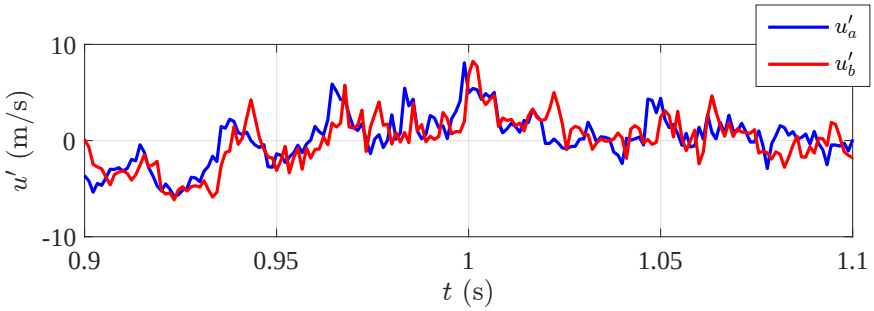

Figure 3. Longitudinal velocity fluctuations $u'(t)$ at positions (a) and (b).

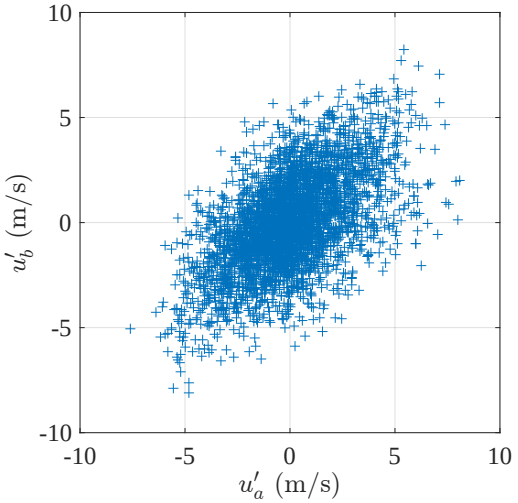

Figure 4. Raw data plotted on a plane.

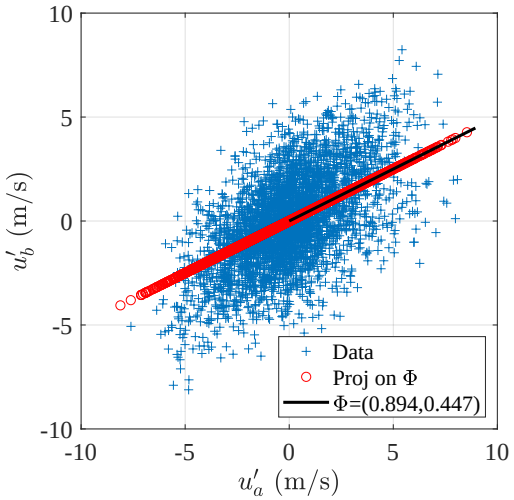

Figure 5. Raw data projected on unit vector ϕ.

see is a sort of ellipse like that shown in Fig. 4. Why do we see an ellipse? Again, this is because our data is *correlated*. Statistically, there is a relationship between u'_a and u'_b: when one moves in one direction, the other also tends to move – on average – in the same direction. Thus, although the velocity fluctuations are essentially random, there is still some order in this randomness, and this order is seen through the correlation between u'_a and u'_b. When we search for coherent structures in a flow, this is precisely what we are interested in: a zone where the fluid moves in sync and the velocity fluctuations are correlated.

We can verify that our data is correlated by computing its covariance matrix $\mathbf{C}$, which in our 2D case is the 2×2 matrix

$$\mathbf{C} = \frac{1}{m-1}\mathbf{U}^T\mathbf{U} = \frac{1}{m-1}\begin{pmatrix} \sum_{i=1}^{m} u'^2_a(t_i) & \sum_{i=1}^{m} u'_a(t_i)u'_b(t_i) \\ \sum_{i=1}^{m} u'_b(t_i)u'_a(t_i) & \sum_{i=1}^{m} u'^2_b(t_i) \end{pmatrix}. \tag{6}$$

The diagonal elements of the covariance matrix are the respective variances of u'_a and u'_b, whereas each off-diagonal element is the covariance between u'_a and u'_b. The covariance matrix is thus necessarily symmetric and if the off-diagonal terms are non-zero, this implies that there is indeed a statistical correlation between u'_a and u'_b. If the covariance matrix were diagonal, this would mean that u'_a and u'_b are perfectly uncorrelated. For the data shown in Fig. 4, the covariance matrix is

$$\mathbf{C} = \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix} = \begin{pmatrix} 4.92 & 2.82 \\ 2.82 & 5.01 \end{pmatrix}. \tag{7}$$

The amount of correlation can be expressed through the correlation coefficient $\rho_{ab} = c_{12}/\sqrt{c_{11}c_{22}} = 0.57$, which shows that u'_a and u'_b are indeed fairly well correlated.[b] The total fluctuating kinetic energy of our data (TKE), which is defined as

$$\text{TKE} = \frac{1}{2}\frac{1}{m-1}\left(\sum_{i=1}^{m} u_a'^2(t_i) + \sum_{i=1}^{m} u_b'^2(t_i)\right) = \frac{1}{2}(c_{11} + c_{22}), \tag{8}$$

is equal to 4.96 m^2/s^2.

Let's go back to Fig. 4. Since the data is plotted on a plane, we can decide to measure the variance of the pair (u'_a, u'_b) along any axis and define this axis as a 'mode' of variation. In Fig. 4 we have used a natural basis to plot our data. On the plane, this natural basis is defined by the two vectors (1,0) and (0,1). So one mode of variation is the horizontal mode (1,0), which physically means the velocity fluctuation at point (a), and the other is the vertical mode (0,1), which means the fluctuation at point (b). The dataset corresponding to fluctuations in the pair of variables (u'_a, u'_b) can naturally be expressed as the fluctuations in u'_a and u'_b taken separately. In the data from Fig. 4, the variances of u'_a and u'_b in the natural basis are 4.92 m^2/s^2 and 5.01 m^2/s^2, respectively (see Eq. 7). Furthermore, the statistical connection between the velocity fluctuations at points (a) and (b) is expressed as the off-diagonal terms of $\mathbf{C}$.

However, we could have also chosen any other basis in the plane to quantify our fluctuations. In fact, to express our data along any direction on the plane, we simply need to project each row of $\mathbf{U}$ onto a unit vector $\boldsymbol{\phi} = (\phi_1, \phi_2)$ pointing in this direction. In other words, we need to compute the dot product of each data point with $\boldsymbol{\phi}$:

$$\mathbf{a} = \mathbf{U}\boldsymbol{\phi} = \begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \\ \vdots & \vdots \\ u_{m1} & u_{m2} \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \end{pmatrix} = \begin{pmatrix} a_1 = u_{11}\phi_1 + u_{12}\phi_2 \\ a_2 = u_{21}\phi_1 + u_{22}\phi_2 \\ \vdots \\ a_m = u_{m1}\phi_1 + u_{m2}\phi_2 \end{pmatrix}. \tag{9}$$

What we obtain is a m-dimensional array $\mathbf{a}$ listing the coordinates a_i of our data points projected on the unit direction vector $\boldsymbol{\phi}$. This is illustrated in Fig. 5 for an exemplary unit vector $\boldsymbol{\phi} = (2/\sqrt{5}, 1/\sqrt{5}) \simeq (0.894, 0.447)$. To compute the variance of our data on this axis, we write

$$\text{var}_{\boldsymbol{\phi}}(\mathbf{a}) = \frac{1}{m-1}\sum_{i=1}^{m} a_i^2(t_i) = \frac{1}{m-1}\mathbf{a}^T\mathbf{a}, \tag{10}$$

and in the case of our example from Fig. 5 we obtain $\text{var}_{\boldsymbol{\phi}}(\mathbf{a}) = 7.19$ m^2/s^2, which is larger than the variances expressed on the horizontal axis ($\text{var}_{(1,0)}(\mathbf{a}) = \text{var}(u'_a) = 4.92$ m^2/s^2) or on the vertical axis ($\text{var}_{(0,1)}(\mathbf{a}) = \text{var}(u'_b) = 5.01$ m^2/s^2).

Now, obviously there is an infinity of ways to do this. Any direction $\boldsymbol{\phi}$ will lead to a new value of the variance in this direction. However, intuitively there are two directions that are particular in Fig. 4, namely the major and minor axes of the ellipse representing our data. It is obvious from Fig. 4 that the maximum variance will be obtained on the major axis and that, once this direction is considered, the 'rest' of the variance will occur on the minor axis. So the projection $\mathbf{U}\boldsymbol{\phi}$ will have maximum variance if $\boldsymbol{\phi}$ is on the major axis of the ellipse. Furthermore, since the major and minor axes are orthogonal, the two unit vectors in their directions form an orthonormal basis for our dataset. These axes are called the *principal axes* or the *proper orthogonal modes* of the dataset.

How do we compute the directions of the major and minor axes? Using tools from linear algebra, one can show that they are the two eigenvectors of the covariance matrix $\mathbf{C}$. There is also an intuitive way to see this: in the proper orthogonal basis, the variance on each axis is maximized and, as a corollary, the covariance between axes should ideally be zero. This means that the covariance matrix should be a 2×2 diagonal

[b] The correlation coefficient ρ between two random variables is defined as their covariance divided by the product of their standard deviations. By construction this coefficient will lie between -1 and 1. When $\rho = 0$ the variables are said to be uncorrelated. When $\rho = \pm 1$ the variables are either perfectly correlated $(+1)$ or perfectly anti-correlated (-1).

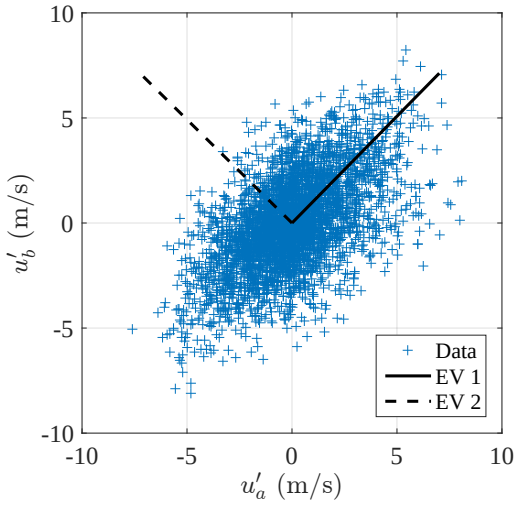

Figure 6. Raw data with directions of the eigenvectors (EVs) of the covariance matrix.

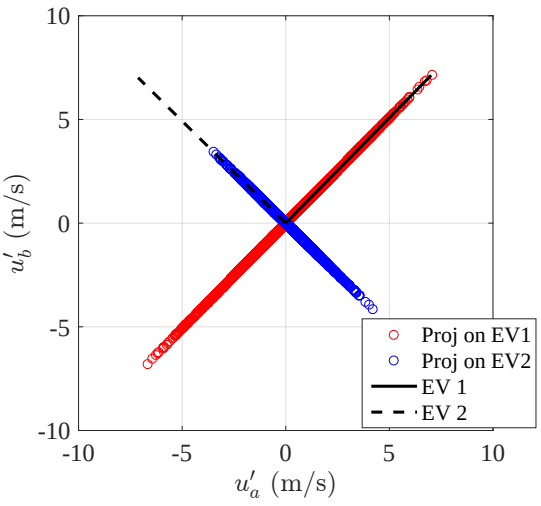

Figure 7. Projection of the data points on the eigenvectors of the covariance matrix.

matrix in the proper orthogonal basis. Now, remember that $\mathbf{C}$ is symmetric and as such its eigenvectors necessarily form an orthonormal basis in which $\mathbf{C}$ can be diagonalized [14]. Therefore, the optimized basis to express our data is simply the set of eigenvectors of the covariance matrix, which are easily computed in MATLAB by: `[PHI LAM] = eig(C)`. Note that, as a convention, we order the eigenvalues from the largest to the smallest, so our first eigenvector is the one that corresponds to the largest eigenvalue and so on. We will see shortly that this ordering is equivalent to ranking the modes based on their contribution to the total fluctuating kinetic energy. Formally, the covariance matrix $\mathbf{C}$ is diagonalized as

$$\begin{aligned}\mathbf{C} &= \mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Phi}^{-1} = \mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Phi}^{T} \\ &= \begin{pmatrix}\phi_{11} & \phi_{12}\\ \phi_{21} & \phi_{22}\end{pmatrix}\begin{pmatrix}\lambda_1 & 0\\ 0 & \lambda_2\end{pmatrix}\begin{pmatrix}\phi_{11} & \phi_{21}\\ \phi_{12} & \phi_{22}\end{pmatrix},\end{aligned} \tag{11}$$

where the two columns of $\mathbf{\Phi}$ are the eigenvectors of $\mathbf{C}$. The second equality in Eq. 11 comes from the fact that $\mathbf{C}$ is symmetric, which implies that its eigenvectors are orthonormal or, equivalently, that $\mathbf{\Phi}$ is an orthogonal matrix ($\mathbf{\Phi}^{-1} = \mathbf{\Phi}^{T}$). Fig. 6 shows our data plotted on the natural basis (u'_a, u'_b) together with the directions of the eigenvectors of the covariance matrix. Clearly, the eigenvectors are indeed on the major and minor axes of the ellipse.

To compute the variance of our data on each principal axis, we need to project the data onto each eigenvector. Thus we write

$$\mathbf{A} = \mathbf{U}\mathbf{\Phi} = \begin{pmatrix}u_{11} & u_{12}\\ u_{21} & u_{22}\\ \vdots & \vdots\\ u_{m1} & u_{m2}\end{pmatrix}\begin{pmatrix}\phi_{11} & \phi_{12}\\ \phi_{21} & \phi_{22}\end{pmatrix}. \tag{12}$$

The projected components are plotted on the natural basis in Fig. 7, where it is obvious that the variance along the first mode (major axis) will be larger than along the second mode (minor axis). In fact, the variance on the major axis is given by the first (largest) eigenvalue λ_1 and the variance on the minor axis is given by the second (smallest) eigenvalue λ_2 since the covariance matrix $\mathbf{C}'$ of the data projected on the proper orthogonal basis is the diagonal matrix $\mathbf{\Lambda}$:

$$\mathbf{C}' = \frac{1}{m-1}\mathbf{A}^{T}\mathbf{A} = \frac{1}{m-1}(\mathbf{U}\mathbf{\Phi})^{T}(\mathbf{U}\mathbf{\Phi}) = \frac{1}{m-1}(\mathbf{\Phi}^{T}\mathbf{U}^{T}\mathbf{U}\mathbf{\Phi}) = \mathbf{\Phi}^{T}\mathbf{C}\mathbf{\Phi} = \mathbf{\Phi}^{T}\mathbf{\Phi}\mathbf{\Lambda}\mathbf{\Phi}^{T}\mathbf{\Phi} = \mathbf{\Lambda}. \tag{13}$$

So why is this useful? Well, the eigenvectors give us an idea of how u'_a and u'_b are correlated, or how they 'move' together. In the proper orthogonal basis, the correlation between u'_a and u'_b is not given by the off-diagonal terms of the covariance matrix anymore since this matrix is diagonal. Rather, the correlation is implicit in the directions of the eigenvectors. In fact, the projected variables (a_{i1}) and (a_{i2}) are specifically constructed to be uncorrelated between each other so that each one can be interpreted as variations on one independent 'mode' of fluctuation. In fluid dynamics, the hope is that these modes can each be linked to an independent coherent structure responsible for the velocity fluctuations. Unfortunately, as we will see later, this link between POD modes and physical coherent structures is far from trivial.

Let's consider some possible directions of the proper orthogonal modes, which are illustrated in Fig. 8.

- In direction $(\sqrt{2}/2, \sqrt{2}/2)$, u'_a and u'_b tend to move in phase with the same amplitude, so we might expect that they are part of the same 'structure' in the flow. This is the case for the first mode in Fig. 7.
- For (1,0), u'_a tends to fluctuate but not u'_b. In this case, the velocity fluctuations are not correlated and the proper orthogonal basis coincides with the natural basis.
- Finally, for $(\sqrt{2}/2, -\sqrt{2}/2)$, u'_a and u'_b tend to fluctuate in opposition of phase with the same amplitude. Here also, this could mean a specific structure in the flow, where the velocity at points (a) and (b) is anti-correlated. This is the case in Fig. 7 for the second mode.

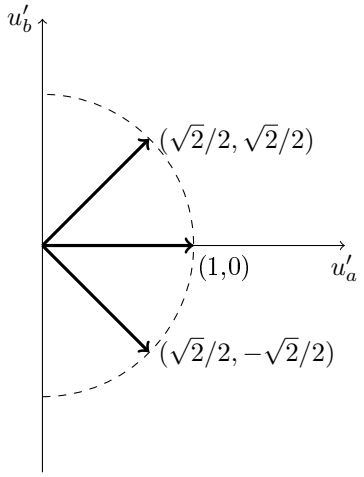

Figure 8. Examples of possible directions of eigenmodes.

Futhermore, the eigenvalues rank the correlation with respect to the variance of the data, which when we consider velocity measurements is the same as saying that they rank the correlation with respect to the kinetic energy of the velocity fluctuations. In the data shown above, $\lambda_1 = 7.78$ m^2/s^2 and $\lambda_2 = 2.15$ m^2/s^2. The TKE is still given by one half of the diagonal elements of the covariance matrix, this time expressed in the proper orthogonal basis.[c] Thus we have again TKE $= \frac{1}{2}(\lambda_1 + \lambda_2) = 4.96$ m^2/s^2. But now we can compute the proportion of TKE produced by each mode: for mode 1 it is $\lambda_1/(\lambda_1 + \lambda_2) \simeq 0.78$ and for mode 2 it is $\lambda_2/(\lambda_1 + \lambda_2) \simeq 0.22$. So approximately 78% of the TKE is accounted for by the first mode and only 22% by the second. From Fig. 7 we see that the first mode is roughly oriented at 45°, which implies a perfect correlation between u'_a and u'_b. Since 78% of the TKE occurs in this first mode, it is not surprising that u'_a and u'_b are well correlated. Looking back at Fig. 7, we conclude that we have essentially decomposed our pair of fluctuating velocities into a large portion of in-phase fluctuations and a smaller portion of out-of-phase fluctuations.

Let's take a break and summarize what we have done so far. We have taken our original matrix of snapshots $\mathbf{S}$ and removed its mean to obtain the matrix $\mathbf{U}$ (note that we typically also call $\mathbf{U}$ a snapshot or data matrix). We have then computed the covariance matrix $\mathbf{C} = \frac{1}{m-1}\mathbf{U}^T\mathbf{U}$ and obtained its matrix of eigenvectors $\mathbf{\Phi}$. Finally, we have projected our matrix $\mathbf{U}$ onto each eigenvector to obtain a new matrix

[c] This is because the trace of an $n \times n$ matrix is equal to the sum of its eigenvalues [14].

$\mathbf{A} = \mathbf{U}\mathbf{\Phi}$ containing the components of our data points on each orthogonal mode. Now, an interesting property of $\mathbf{\Phi}$ is that it is orthogonal, which means that $\mathbf{\Phi}^{-1} = \mathbf{\Phi}^T$ (again, this is because each column of $\mathbf{\Phi}$ is an eigenvector of the symmetric matrix $\mathbf{C}$ [14]). Therefore, we can easily write our original matrix $\mathbf{U}$ in terms of $\mathbf{A}$:

$$\mathbf{A} = \mathbf{U}\mathbf{\Phi} \Rightarrow \mathbf{U} = \mathbf{A}\mathbf{\Phi}^{-1} = \mathbf{A}\mathbf{\Phi}^T, \tag{14}$$

or, written explicitly,

$$\begin{aligned}
\mathbf{U} = \begin{pmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \\ \vdots & \vdots \\ u_{m1} & u_{m2} \end{pmatrix} &= \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ \vdots & \vdots \\ a_{m1} & a_{m2} \end{pmatrix} \begin{pmatrix} \phi_{11} & \phi_{21} \\ \phi_{12} & \phi_{22} \end{pmatrix} \\
&= \begin{pmatrix} a_{11}\phi_{11} + a_{12}\phi_{12} & a_{11}\phi_{21} + a_{12}\phi_{22} \\ a_{21}\phi_{11} + a_{22}\phi_{12} & a_{21}\phi_{21} + a_{22}\phi_{22} \\ \vdots & \vdots \\ a_{m1}\phi_{11} + a_{m2}\phi_{12} & a_{m1}\phi_{21} + a_{m2}\phi_{22} \end{pmatrix} \\
&= \begin{pmatrix} a_{11}\phi_{11} & a_{11}\phi_{21} \\ a_{21}\phi_{11} & a_{21}\phi_{21} \\ \vdots & \vdots \\ a_{m1}\phi_{11} & a_{m1}\phi_{21} \end{pmatrix} + \begin{pmatrix} a_{12}\phi_{12} & a_{12}\phi_{22} \\ a_{22}\phi_{12} & a_{22}\phi_{22} \\ \vdots & \vdots \\ a_{m2}\phi_{12} & a_{m2}\phi_{22} \end{pmatrix} \\
&= \begin{pmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{pmatrix} \begin{pmatrix} \phi_{11} & \phi_{21} \end{pmatrix} + \begin{pmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{m2} \end{pmatrix} \begin{pmatrix} \phi_{12} & \phi_{22} \end{pmatrix} \\
&= \tilde{\mathbf{U}}^{\mathbf{1}} + \tilde{\mathbf{U}}^{\mathbf{2}},
\end{aligned} \tag{15}$$

where by definition for $k = 1, 2$:

$$\tilde{\mathbf{U}}^k = \begin{pmatrix} \tilde{u}^k_{11} & \tilde{u}^k_{12} \\ \tilde{u}^k_{21} & \tilde{u}^k_{22} \\ \vdots & \vdots \\ \tilde{u}^k_{m1} & \tilde{u}^k_{m2} \end{pmatrix} = \begin{pmatrix} a_{1k} \\ a_{2k} \\ \vdots \\ a_{mk} \end{pmatrix} \begin{pmatrix} \phi_{1k} & \phi_{2k} \end{pmatrix}. \tag{16}$$

This brings us to another very interesting property of the decomposition in proper orthogonal modes. The original matrix $\mathbf{U}$ (that is, our original dataset of fluctuating velocities) can be expressed as the sum of two terms: matrix $\tilde{\mathbf{U}}^1$, whose components are the data projected onto the major axis but expressed in the natural basis, plus matrix $\tilde{\mathbf{U}}^2$, whose components are the projections onto the minor axis, also expressed in the natural basis. In other words, we have decomposed our original velocity fluctuations into the sum of a contribution from the first mode $\tilde{\mathbf{U}}^1$ and a contribution from the second mode $\tilde{\mathbf{U}}^2$. To facilitate the geometric interpretation of these matrix operations, the projections involved in Eqs. 12 and 16 are illustrated in Fig. 9.

Now, remember that in our particular case, the contribution from the first mode is largely dominant compared to the second mode. So we could try to simplify our data analysis by looking only at the first term on the right hand side of Eq. 15. In other words, we could look only at the fluctuations of the first mode $\tilde{\mathbf{U}}^1$ and neglect those of the second mode. This is illustrated in Fig. 10, where the time traces of u'_a and u'_b are plotted separately, together with the contributions from mode 1 and 2 ($\tilde{\mathbf{U}}^1$ and $\tilde{\mathbf{U}}^2$). In this example, it is clear that the first mode is a fairly good approximation of the total fluctuations and that matrix $\mathbf{U}$ can be reasonably well approximated by matrix $\tilde{\mathbf{U}}^1$. This is consistent with the fact that, as shown above, about 78% of the TKE can be 'explained' by the first mode. This feature of the POD is called *dimensionality*

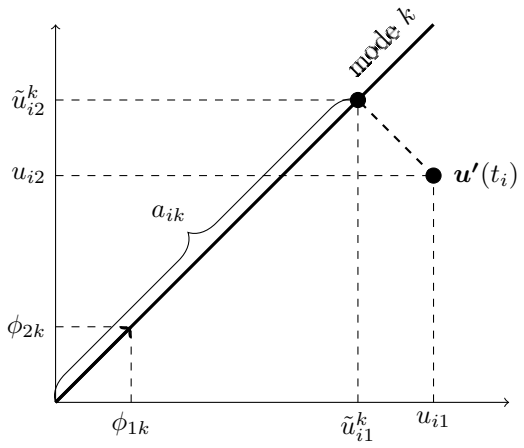

Figure 9. Projections involved in Eqs. 12 and 16.

reduction and although it doesn't really make sense to do this on a 2D problem, it is an extremely useful property for more complicated problems.

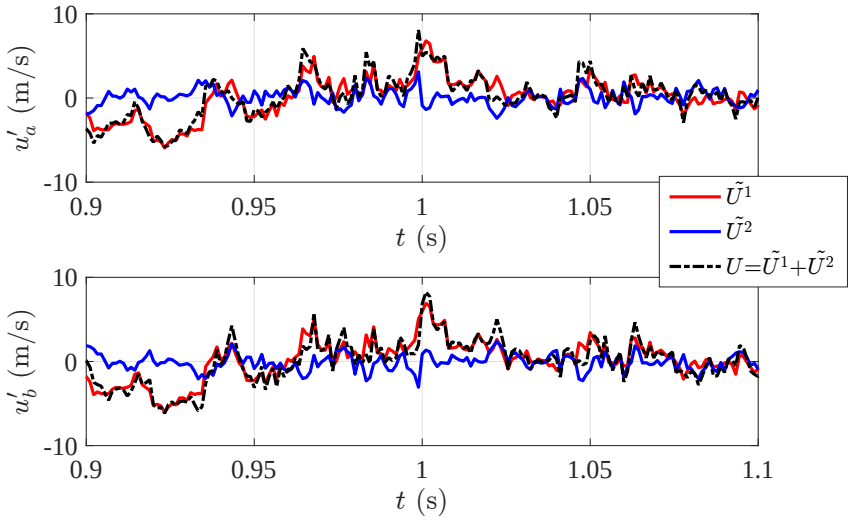

Figure 10. Contributions from modes 1 and 2 to u'_a and u'_b.

IV. The n-Dimensional Example

While the 2D example is quite useful to visualize the different projections occurring in the POD algorithm, the real strength of POD as a decomposition method is on larger-dimensional datasets. To illustrate this, we now consider the POD on the complete longitudinal velocity field measured by PIV. That is, we have an $n = N_x \times N_y = 5805$-dimensional problem.

To create our new snapshot matrix, we start by subtracting the time-averaged velocity field (Fig. 2) from each of our individual vector fields. We obtain a set of $m = N_t = 3580$ fluctuating velocity fields $u'(x_i, y_j, t_k)$. For each instant t_k, we then reorder our dataset by concatenating each individual velocity field into one single $1 \times n$ row and we stack those rows onto each other. All our fluctuating velocity data is then summarized into one $m \times n$ snapshot matrix $\mathbf{U}$:

$$\mathbf{U} = \begin{pmatrix} u_{11} & \dots & u_{1n} \\ u_{21} & \dots & u_{2n} \\ \vdots & & \vdots \\ u_{m1} & \dots & u_{mn} \end{pmatrix} = \begin{pmatrix} u'(x_1, y_1, t_1) & \dots & u'(x_{Nx}, y_{Ny}, t_1) \\ u'(x_1, y_1, t_2) & \dots & u'(x_{Nx}, y_{Ny}, t_2) \\ \vdots & & \vdots \\ u'(x_1, y_1, t_m) & \dots & u'(x_{Nx}, y_{Ny}, t_m) \end{pmatrix}, \tag{17}$$

where each row represents one snapshot measured by the PIV system. In other words, each element (u_{ij}) of $\mathbf{U}$ is the velocity fluctuation at point j measured at time i.

We then follow the same steps as in the 2D example. We start by computing the covariance matrix $\mathbf{C} = \frac{1}{m-1}\mathbf{U}^T\mathbf{U}$, which is now an $n \times n$ matrix (about 33.7 million elements in our example with $n = 5805$). We then compute the eigenvalues and eigenvectors of $\mathbf{C}$ through the MATLAB command `[PHI LAM] = eig(C)` and we order them from the largest eigenvalue to the smallest. We obtain n eigenvalues $\lambda_1 ... \lambda_n$ and a set of n eigenvectors arranged as columns in an $n \times n$ matrix $\mathbf{\Phi}$:

$$\mathbf{\Phi} = \begin{pmatrix} \phi_{11} & \dots & \phi_{1n} \\ \phi_{21} & \dots & \phi_{2n} \\ \vdots & & \vdots \\ \phi_{n1} & \dots & \phi_{nn} \end{pmatrix}. \tag{18}$$

As before, the n eigenvectors (the n columns of $\mathbf{\Phi}$), are the proper orthogonal modes of the dataset. They can be seen as the axes of an n-dimensional ellipsoid that fits the total dataset in n-dimensional space. Furthermore, the modes are ordered according to the variance in their direction.

How can we visualize these modes? Remember that to create our snapshot matrix $\mathbf{U}$, we have reordered our data so that each snapshot is concatenated into a single $1 \times n$ row matrix. We can now follow exactly the inverse procedure and create a synthetic $N_x \times N_y$ field for each of the columns of $\mathbf{\Phi}$. The resulting scalar fields are thus often called *spatial* modes. As an illustration, the left hand side of Fig. 11 shows the three first POD modes (the first three columns of $\mathbf{\Phi}$) of our turbulent separation bubble as contour plots. Technically, the dimension of the contour plots is that of velocity, although the plots are often normalized since we are more interested in the variation of the scalar variable within and between the fields than in their absolute value.

At this point, it is worth mentioning that the procedure of reordering a $N_x \times N_y$ field into a $1 \times n$ row can be done in many ways. The exact procedure is unimportant as long as the same procedure is used in reverse to visualize the modes.[d] Furthermore, we have decided here to compute the POD on the longitudinal velocity data only but we could have used the vertical velocity instead, or even both the longitudinal and vertical velocity fluctuations in the same snapshot matrix. Had we decided to compute the POD with both the u' and v' components, we would simply have extended each row of the snapshot matrix by the v' components, *viz.*

$$\mathbf{U} = \begin{pmatrix} u'(x_1, y_1, t_1) & \dots & u'(x_{Nx}, y_{Ny}, t_1) & v'(x_1, y_1, t_1) & \dots & v'(x_{Nx}, y_{Ny}, t_1) \\ u'(x_1, y_1, t_2) & \dots & u'(x_{Nx}, y_{Ny}, t_2) & v'(x_1, y_1, t_2) & \dots & v'(x_{Nx}, y_{Ny}, t_2) \\ \vdots & & \vdots & \vdots & & \vdots \\ u'(x_1, y_1, t_m) & \dots & u'(x_{Nx}, y_{Ny}, t_m) & v'(x_1, y_1, t_m) & \dots & v'(x_{Nx}, y_{Ny}, t_m) \end{pmatrix}. \tag{19}$$

The snapshot matrix would have been twice as large ($n = 11610$) and the covariance matrix would have been a 11610×11610 matrix (about 135 million elements). The resulting modes would then each be a 2D vector field instead of a scalar field. Furthermore, had we had access to three-dimensional velocity data on a 3D grid (through the use of a tomographic PIV system, for example), we could have used all three components of the velocity fields and obtained volumetric POD modes. The only difference would be the number of elements in the different matrices involved and consequently the storage requirements and the computation time (the number of columns in the snapshot matrix is doubled for two velocity components and tripled for three components).

[d] In fact, we could also have arranged our $N_x \times N_y$ fields into columns instead of rows. In this case the covariance matrix would have been computed as $\mathbf{C} = \frac{1}{m-1}\mathbf{U}\mathbf{U}^T$.

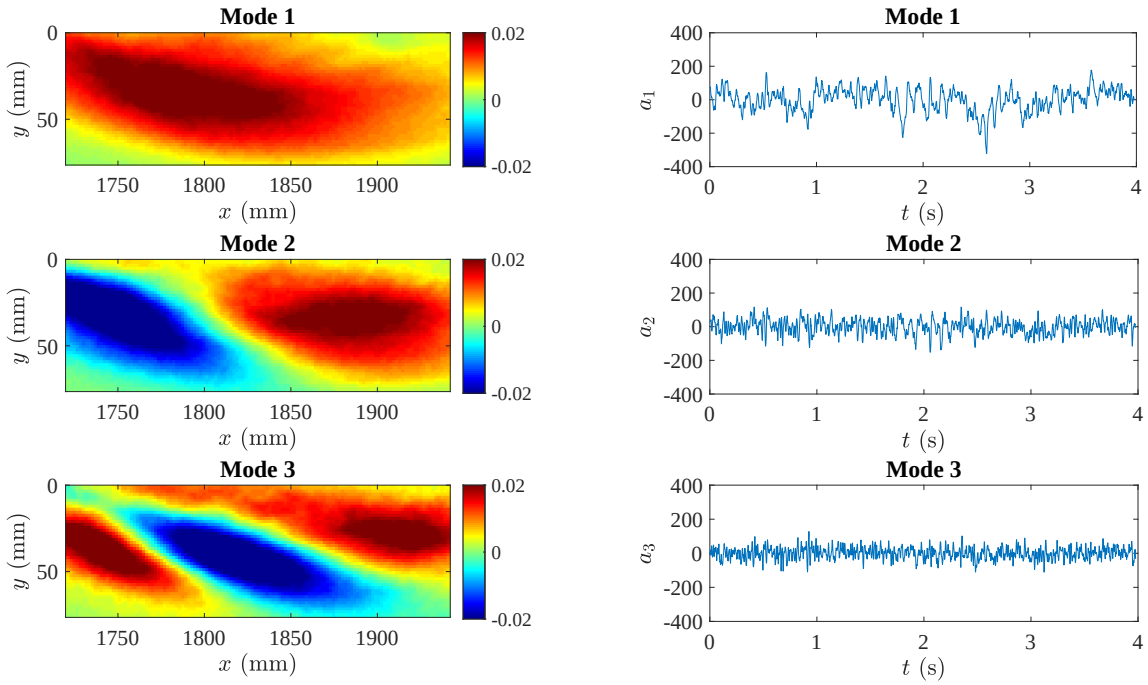

Figure 11. Left column: first, second and third POD modes of longitudinal velocity fluctuations in the TSB. Right column: time coefficients of first, second and third POD modes.

Now, how can we interpret our POD modes? Here it is again useful to remember what was done in the 2D case. In Fig. 8, we interpreted the mode $(\sqrt{2}/2, \sqrt{2}/2)$ as a synchronized fluctuation of u'_a and u'_b. The n-D equivalent of mode $(\sqrt{2}/2, \sqrt{2}/2)$ is a region where the spatial mode has approximately the same value. For example, in the contour plot of mode 1 (Fig. 11), there is a large red region that encompasses roughly the size of the TSB. This means that the longitudinal velocity fluctuations in this region tend to be correlated and this is interpreted as a global fluctuation of the complete TSB (for more details about this specific case, see [13]). Similarly, in the 2D case, we interpreted the mode $(\sqrt{2}/2, -\sqrt{2}/2)$ as anti-correlated velocity fluctuations at both measurement points. In the contour plots of modes 2 and 3, we see red and blue patches with opposite signs. Here also, this means that the longitudinal velocity in these regions is anti-correlated: when the velocity moves up in the red zone, it tends to move down in the blue zone, and vice versa. As we will see later, this type of POD mode is often interpreted as a sign of convected structures.

Now in the n-D case, there are obviously a lot of modes. Is it necessary to interpret all of them? Well, typically we can concentrate on only a few of the low-order modes. This is because the POD ranks the modes according to their contribution to the total variance. As an illustration, the 10 first eigenvalues from our calculation are plotted in Fig. 12. The actual eigenvalues are shown on the left axis and their percent contribution to the TKE is shown on the right (recall that the contribution from mode i is simply $\lambda_i/\Sigma_k\lambda_k$). It can be seen that the first eigenvalue is largely dominant, with approximately 20% of the TKE. The second mode contributes 8% and the contribution from the next ones diminishes slowly. This indicates that a large chunk of the total fluctuations is well represented by the first mode, whereas the remaining turbulent fluctuations are spread among the higher modes. So looking at the first few modes is generally sufficient to identify dominant coherent motions.

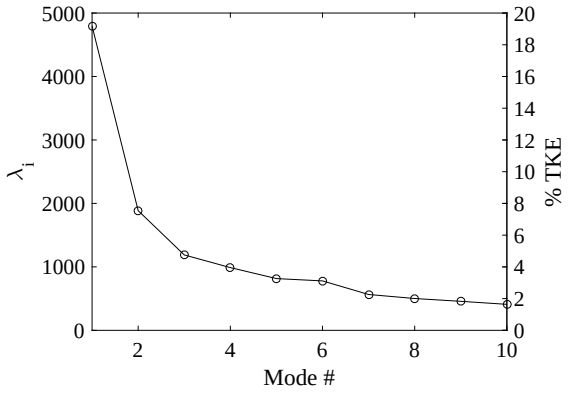

Figure 12. Ten first eigenvalues for the TSB example. Left axis: eigenvalues; right axis: percentage of TKE from each eigenvalue.

Now, like in the 2D case, we can project our original dataset onto each of the n modes by writing $\mathbf{A} = \mathbf{U}\mathbf{\Phi}$. Written explicitly we have:

$$\begin{pmatrix} a_{11} & \dots & a_{1n} \\ a_{21} & \dots & a_{2n} \\ \vdots & & \vdots \\ a_{m1} & \dots & a_{mn} \end{pmatrix} = \begin{pmatrix} u_{11} & \dots & u_{1n} \\ u_{21} & \dots & u_{2n} \\ \vdots & & \vdots \\ u_{m1} & \dots & u_{mn} \end{pmatrix} \begin{pmatrix} \phi_{11} & \dots & \phi_{1n} \\ \phi_{21} & \dots & \phi_{2n} \\ \vdots & & \vdots \\ \phi_{n1} & \dots & \phi_{nn} \end{pmatrix}. \tag{20}$$

Referring back to Fig. 9, we can interpret each a_{ij} as the projection of the data measured at time i on mode j. Here it is important to remember that each data point in our $n = 5805$-dimensional space represents one velocity field measured by PIV. In other words, each point in our n-dimensional space represents the value of the longitudinal velocity at n points in physical space. We refer to the columns of $\mathbf{A}$ as the *time coefficients* of the modes since for each snapshot $k = 1, \dots, m$ there will be one projection on each mode (again it is useful to visualize the projection in the 2D case in Fig. 9). For the case of our turbulent separation bubble, the time coefficients of the first three modes (the first three columns of $\mathbf{A}$ expressed as a function of time), are shown on the right hand side of Fig. 11.

Furthermore, the original snapshot matrix $\mathbf{U}$ can be expressed as the sum of the contributions from the n modes. Indeed, we still have $\mathbf{U} = \mathbf{A}\mathbf{\Phi}^{-1} = \mathbf{A}\mathbf{\Phi}^{T}$:

$$\begin{pmatrix} u_{11} & \dots & u_{1n} \\ u_{21} & \dots & u_{2n} \\ \vdots & & \vdots \\ u_{m1} & \dots & u_{mn} \end{pmatrix} = \begin{pmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{pmatrix} \begin{pmatrix} \phi_{11} & \dots & \phi_{n1} \end{pmatrix} + \dots + \begin{pmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{pmatrix} \begin{pmatrix} \phi_{1n} & \dots & \phi_{nn} \end{pmatrix} \tag{21}$$

$$= \begin{pmatrix} \tilde{u}^1_{11} & \dots & \tilde{u}^1_{1n} \\ \tilde{u}^1_{21} & \dots & \tilde{u}^1_{2n} \\ \vdots & \vdots & \\ \tilde{u}^1_{m1} & \dots & \tilde{u}^1_{mn} \end{pmatrix} + \dots + \begin{pmatrix} \tilde{u}^n_{11} & \dots & \tilde{u}^n_{1n} \\ \tilde{u}^n_{21} & \dots & \tilde{u}^n_{2n} \\ \vdots & \vdots & \\ \tilde{u}^n_{m1} & \dots & \tilde{u}^n_{mn} \end{pmatrix} \tag{22}$$

with

$$\begin{pmatrix} \tilde{u}^k_{11} & \dots & \tilde{u}^k_{1n} \\ \tilde{u}^k_{21} & \dots & \tilde{u}^k_{2n} \\ \vdots & \vdots & \\ \tilde{u}^k_{m1} & \dots & \tilde{u}^k_{mn} \end{pmatrix} = \begin{pmatrix} a_{1k} \\ a_{2k} \\ \vdots \\ a_{mk} \end{pmatrix} \begin{pmatrix} \phi_{1k} & \dots & \phi_{nk} \end{pmatrix} = \tilde{\mathbf{U}}^k. \tag{23}$$

Notice that each matrix $\tilde{\mathbf{U}}^k$ has the same dimension as $\mathbf{U}$ and that:

$$\mathbf{U} = \sum_{k=1}^{n} \tilde{\mathbf{U}}^k. \tag{24}$$

Eq. 24 implies that we have decomposed our original velocity fluctuations into a sum of n contributions from n proper orthogonal modes. This is the finite-dimensional equivalent of the original infinite-dimensional POD theorem (Eq. 1) which states that the fluctuating longitudinal velocity field $\boldsymbol{u'}(\mathbf{x}, t)$ can be decomposed into a sum of deterministic spatial functions $\boldsymbol{\Phi}_k(\mathbf{x})$ each multiplied by a fluctuating (random) time coefficient $a_k(t)$, and which is repeated here for clarity:

$$\boldsymbol{u'}(\mathbf{x}, t) = \sum_{k=1}^{\infty} a_k(t)\boldsymbol{\Phi}_k(\mathbf{x}). \tag{25}$$

Here we have used the traditional notation in which $a_k(t)$ is the time coefficient of POD mode $\boldsymbol{\Phi}_k$. In the finite-dimensional case, and with our previous notation, $a_k(t)$ is the column vector $(a_{1k}a_{2k}\dots a_{mk})^T$ and $\boldsymbol{\Phi}_k(\mathbf{x})$ is the row vector $(\phi_{1k}\phi_{2k}\dots\phi_{nk})$. Again, in our example there are $m = 3580$ time coefficients (one per snapshot) and $n = 5805$ spatial points to represent each mode. The orthonormality of the POD modes, first discussed in Eq. 2, translates into Eq. 21 in finite dimensions: if matrix $\boldsymbol{\Phi}$ were not orthogonal (*i.e.* if $\boldsymbol{\Phi}^{-1} \neq \boldsymbol{\Phi}^T$), it would not be possible to express u_{ij} as a sum of the time coefficients a_{ik} times the eigenvectors ϕ_{jk}.

This is where the notion of dimensionality reduction becomes so powerful. Recall that in our example of a turbulent separation bubble, about 20% of the TKE in the flow is accounted for by the first POD mode (Fig. 12). What is this mode doing, physically, to produce 20% of the fluctuations? In fact, we can visualize the motion that subtends these fluctuations by looking at $\tilde{\mathbf{U}}^1$. All we have to do is reorder matrix $\tilde{\mathbf{U}}^1$ into a series of $m = 3580$ snapshots, add the mean field $\overline{\boldsymbol{U}}$, and look at the resulting velocity fields. This is done in Fig. 13, where it can be seen that in this particular case, since the first mode has a global nature, the motion that it represents is a contraction and expansion of the complete separation bubble. What we have done here is create a *low-order model* (LOM) of the longitudinal velocity using only the first POD mode. Now, it is important to realize that this new series of snapshots does not correspond to any real, physical flow. It is an approximation that accounts for only 20% of the TKE, the other 80% being accounted for by the other modes. In effect, Fig. 13 is the n-dimensional equivalent of the red curves in Fig. 10.

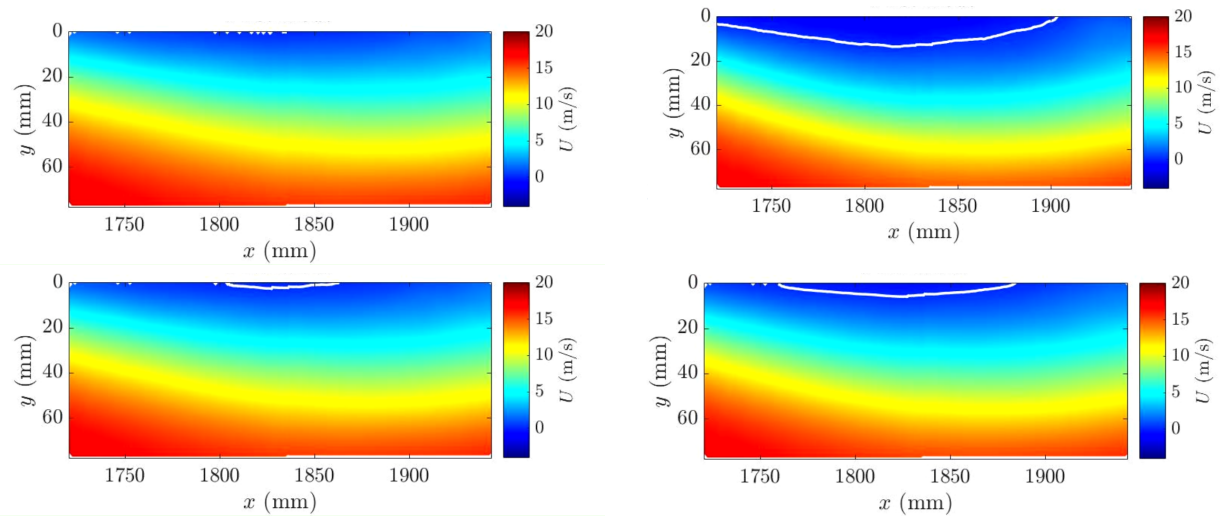

Figure 13. Low-order model $\tilde{\mathbf{U}}^1$ of longitudinal velocity field in the turbulent separation bubble with first POD mode at four random instants. The white line delimits the region of mean backflow.

In this specific example, the first POD mode is idiosyncratic because it contains a fairly large amount of energy and because its shape is different than that of the other modes. In a more general setting, model reduction consists of approximating the complete fluctuating velocity field by a low-order model using a

limited number of POD modes in order to concentrate on the structures responsible for a specific portion of the TKE, *viz.*

$$\boldsymbol{u'}(\mathbf{x},t) \simeq \tilde{\boldsymbol{u'}}(\mathbf{x},t) = \sum_{k=k_1}^{k=k_2} a_k(t)\boldsymbol{\Phi}_k(\mathbf{x}), \tag{26}$$

or, in matrix terms:

$$\mathbf{U} \simeq \tilde{\mathbf{U}} = \sum_{k=k_1}^{k_2} \tilde{\mathbf{U}}^k. \tag{27}$$

Model reduction via POD can also be used to generate a set of ordinary differential equations (a finite-dimensional dynamical system) as a simplification of the partial differential equations normally used to solve fluid-mechanics problems. This is called Galerkin projection but it is out of the scope of the present document.

In summary, we now show the MATLAB code of the n-dimensional POD as described above. The raw velocity data is organized as a three-dimensional array $S(i,j,k)$ where the first index is the y-position in the field of view, the second is the x-position, and the third is time. This particular way of computing the POD is called the *direct* method.

```
% Create snapshot matrix
Nt = length(S(1,1,:));
S = reshape(permute(S, [3 2 1]), Nt, [ ]);  % Reshape data into a matrix S with Nt rows
U = S - repmat(mean(S,1), Nt, 1);           % Subtract the temporal mean from each row

% Create covariance matrix
C = (U'*U)/(Nt-1);

% Solve eigenvalue problem
[PHI LAM] = eig(C,'vector');

% Sort eigenvalues and eigenvectors
[lambda,ilam] = sort(LAM,'descend');
PHI = PHI(:, ilam);                         % These are the spatial modes

% Calculate time coefficients
A = U*PHI;

% Reconstruction on mode k
k = 1;  % for example
Utilde_k = A(:,k)*PHI(:,k)';
```

V. The Snapshot POD

The original POD equation (Eq. 1) is essentially symmetric in t and $\mathbf{x}$ since mathematically there is no fundamental difference between the temporal variable t and the spatial variable $\mathbf{x}$. So instead of seeing it as a decomposition involving deterministic spatial modes and random time coefficients, we could also see it as a decomposition in deterministic temporal modes with random spatial coefficients. In other words, we can interchange t and $\mathbf{x}$ in our algorithm. This is called the *snapshot POD* and it was originally introduced by Sirovich [15].

There are two ways to do this. A first possibility would be to transpose our snapshot matrix and follow the same algorithm as in the previous section. In the second, equivalent method, we start with the same $m \times n$ snapshot matrix $\mathbf{U}$ as before but we build the correlation matrix $\mathbf{C_s} = \frac{1}{m-1}\mathbf{U}\mathbf{U}^T$, which is now $m \times m$ instead of the $n \times n$ matrix $\mathbf{C}$ of the direct method. Recall that $m = N_t$ is the total number of velocity fields acquired from the PIV and $n = N_x \times N_y$ is the number of spatial velocity points measured in each field. Thus,

$\mathbf{C_s}$ is built by averaging in space instead of time.[e] We then compute the eigenvalues and eigenvectors of $\mathbf{C_s}$ through the command `[A_s LAM_s] = eig(C_s)` and, as before, we order them from the largest eigenvalue to the smallest. This time, however, we have performed our decomposition in m-dimensional space and we now have a set of m eigenvalues and eigenvectors. This means that the m eigenvectors of $\mathbf{C_s}$ are *temporal modes* that play the same role as the m time coefficients of the direct method. In order to obtain the spatial coefficients, we have to project our velocity data onto the temporal modes by writing $\mathbf{\Phi_s} = \mathbf{U}^T\mathbf{A_s}$. Note that transposition of $\mathbf{U}$ is necessary from a dimensional point of view and that we now have only m modes instead of n. Matrix $\mathbf{\Phi_s}$ then contains our m spatial coefficients (which are equivalent to the spatial modes of the direct method), ordered from the 'strongest' (first column) to the 'weakest' (last column).

As will be demonstrated below, the eigenvalues obtained from the snapshot method are the same as those obtained from the direct method. The spatial coefficients and temporal modes of the snapshot method, however, differ from the spatial modes and temporal coefficients of the direct method by a multiplicative factor. In the direct method, the spatial modes are orthonormal since they result from the eigendecomposition of the symmetric matrix $\mathbf{C}$ but the time coefficients are not (in fact, the time coefficients are only *orthogonal* since $\mathbf{A}^T\mathbf{A} = (m-1)\mathbf{\Lambda}$, Eq. 13). On the contrary, in the snapshot method, the temporal modes are orthonormal but not the spatial coefficients. Thus, if we want to match the results of both methods, we need to normalize each spatial coefficient of the snapshot POD and scale the temporal modes accordingly. This procedure, which ensures that the results of the snapshot POD are equal to those of the direct method, will be detailed in the example code below.[f] However, this is not strictly necessary since the original snapshot matrix $\mathbf{U}$ can be reconstructed from the spatial coefficients and temporal modes obtained before normalization:

$$\mathbf{\Phi_s} = \mathbf{U}^T\mathbf{A_s} \Rightarrow \mathbf{U}^T = \mathbf{\Phi_s}{\mathbf{A_s}}^{-1} = \mathbf{\Phi_s}{\mathbf{A_s}}^T \Rightarrow \mathbf{U} = \mathbf{A_s}{\mathbf{\Phi_s}}^T. \tag{28}$$

Similarly to Eq. 24, we can therefore write

$$\mathbf{U} = \sum_{k=1}^{m} \tilde{\mathbf{U}}_\mathbf{s}^k, \tag{29}$$

with

$$\tilde{\mathbf{U}}_\mathbf{s}^k = \begin{pmatrix} (a_s)_{1k} \\ (a_s)_{2k} \\ \vdots \\ (a_s)_{mk} \end{pmatrix} \begin{pmatrix} (\phi_s)_{1k} & \dots & (\phi_s)_{nk} \end{pmatrix}. \tag{30}$$

Now why would we bother using the snapshot POD instead of the direct method? Simply because it is often *faster* to run than the direct algorithm. In most practical cases involving planar or volumetric velocity data, the number n of spatial measurement points is larger than the number m of snapshots: $n > m$. This means that the correlation matrix $\mathbf{C_s}$ is smaller and easier to store, and that the corresponding eigenvalue problem is faster to solve. In the example of the turbulent separation bubble, $\mathbf{C}$ has 33.7 million elements but $\mathbf{C_s}$ only has 12.8 million. Furthermore, had we considered both the longitudinal and vertical velocity components, $\mathbf{C}$ would have had 135 million elements but $\mathbf{C_s}$ would still only have 12.8 million elements. The snapshot POD is therefore often used in POD calculations involving PIV or CFD datasets, whereas the direct method is preferred for measurements with a limited number of single-point probes with high temporal resolution. On top of this, the results are essentially the same: when $m < n$, the last $n - m$ eigenvalues of the direct method are zero and the last $n - m$ modes are irrelevant. Thus, when $m \neq n$, either procedure returns $\min(m, n)$ modes. While the modes calculated separately by the two methods may have opposite signs, this is not an issue since the corresponding time coefficients will also have opposite signs.

To conclude this section, the MATLAB code of the snapshot POD is shown here assuming the same $m \times n$ matrix of snapshots $\mathbf{U}$ as before:

```
% Create correlation matrix
C_s = (U*U')/(Nt-1);
```

[e] The reader might wonder why we define $\mathbf{C_s}$ with $m-1$ instead of $n-1$. This will become clear in the next section.

[f] Some modes and coefficients obtained by the two methods may still have opposite sign but this is not an issue since the velocity data is recovered when both are multiplied by one another.

```
% Solve eigenvalue problem
[A_s LAM_s] = eig(C_s,'vector');

% Sort eigenvalues and eigenvectors
[lambda_s,ilam_s] = sort(LAM_s,'descend');
A_s = A_s(:, ilam_s);                    % These are the temporal modes

% Calculate spatial coefficients
PHI_s = U'*A_s;

% Reconstruction on mode k
k = 1;  % for example
Utilde_k_s = A_s(:,k)*PHI_s(:,k)';

% Normalization to match direct and snapshot modes (optional)
PHI = normc(PHI_s);                      % Spatial modes
A = U*PHI;                               % Time coefficients
Utilde_k = A(:,k)*PHI(:,k)';             % Reconstruction on mode k
```

VI. The Singular Value Decomposition

The reason why both direct POD and snapshot POD lead to equivalent results is that both methods are closely related to the *singular value decomposition* (SVD) of the snapshot matrix $\mathbf{U}$. The SVD is a matrix decomposition that factorizes a real $m \times n$ matrix $\mathbf{U}$ into

$$\mathbf{U} = \mathbf{L}\mathbf{\Sigma}\mathbf{R}^T, \tag{31}$$

where $\mathbf{L}$ is an $m \times m$ orthogonal matrix, $\mathbf{\Sigma}$ is an $m \times n$ rectangular diagonal matrix and $\mathbf{R}$ is an $n \times n$ orthogonal matrix [12]. The non-zero diagonal elements of $\mathbf{\Sigma}$ are typically a set of $r = \min(m, n)$ positive numbers arranged in decreasing order, *i.e.* $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r \geq 0$. These are called the *singular values* of $\mathbf{U}$. Essentially, the SVD is a procedure that will diagonalize any rectangular matrix, whereas eigenvalue decomposition only works for square matrices.

What is the connection between SVD and POD? To see this, let us compute the matrices $\mathbf{C}$ and $\mathbf{C_s}$ in light of the SVD. We have

$$\begin{aligned}
\mathbf{C} &= \frac{1}{m-1}\left(\mathbf{U}^T\mathbf{U}\right) \\
&= \frac{1}{m-1}\left(\left(\mathbf{L}\mathbf{\Sigma}\mathbf{R}^T\right)^T\left(\mathbf{L}\mathbf{\Sigma}\mathbf{R}^T\right)\right) \\
&= \frac{1}{m-1}\left(\mathbf{R}\mathbf{\Sigma}^T\mathbf{L}^T\mathbf{L}\mathbf{\Sigma}\mathbf{R}^T\right) \\
&= \frac{1}{m-1}\left(\mathbf{R}\left(\mathbf{\Sigma}^T\mathbf{\Sigma}\right)\mathbf{R}^T\right)
\end{aligned} \tag{32}$$

and similarly

$$\begin{aligned}
\mathbf{C_s} &= \frac{1}{m-1}\left(\mathbf{U}\mathbf{U}^T\right) \\
&= \frac{1}{m-1}\left(\left(\mathbf{L}\mathbf{\Sigma}\mathbf{R}^T\right)\left(\mathbf{L}\mathbf{\Sigma}\mathbf{R}^T\right)^T\right) \\
&= \frac{1}{m-1}\left(\mathbf{L}\mathbf{\Sigma}\mathbf{R}^T\mathbf{R}\mathbf{\Sigma}^T\mathbf{L}^T\right) \\
&= \frac{1}{m-1}\left(\mathbf{L}\left(\mathbf{\Sigma}\mathbf{\Sigma}^T\right)\mathbf{L}^T\right).
\end{aligned} \tag{33}$$

Let us first notice that $\mathbf{\Sigma}^T\mathbf{\Sigma}$ is an $n \times n$ diagonal matrix, that $\mathbf{\Sigma}\mathbf{\Sigma}^T$ is an $m \times m$ diagonal matrix and that the non-zero diagonal elements of these two matrices are exactly the same, namely the squares of the r singular values of $\mathbf{U}$. Furthermore, if we compare Eq. 32 with Eq. 11, we see that $\mathbf{\Lambda} = \mathbf{\Sigma}^T\mathbf{\Sigma}/(m-1)$ and that the matrix $\mathbf{\Phi}$ (the eigenvectors of $\mathbf{C}$) is equal to the matrix $\mathbf{R}$. Similarly, comparing Eq. 33 with the snapshot POD algorithm shows that $\mathbf{\Lambda_s} = \mathbf{\Sigma}\mathbf{\Sigma}^T/(m-1)$ and that $\mathbf{A_s} = \mathbf{L}$. In other words, the eigenvalues λ_i of $\mathbf{C}$ and $\mathbf{C_s}$ are both equal to $\sigma_i^2/(m-1)$. [g]

Therefore, both POD algorithms are equivalent to the SVD of matrix $\mathbf{U}/\sqrt{m-1}$: the spatial modes of the direct POD are given by its right singular vectors $\mathbf{R}$, the temporal modes of the snapshot POD are given by its left singular vectors $\mathbf{L}$ and the eigenvalues of both methods are the squares of its singular values. To be consistent with the direct POD algorithm, it is common to keep the spatial modes as-is and to rescale the temporal modes into the time coefficients of the direct method, as was done for the snapshot POD. Note also that in some POD algorithms, the factor $m-1$ is simply omitted since it only amounts to a scaling of the results.

In MATLAB, the SVD is run by the simple command: `[L,SIG,R] = svd(U)`. Our final POD algorithm is then programmed with the following few lines of code:

```
% Singular value decomposition
[L,SIG,R] = svd(U/sqrt(Nt-1));
PHI = R;                                    % PHI are the spatial modes
A = U*PHI;                                  % A are the time coefficients
lambda = diag(SIG).^2;                      % lambda are the eigenvalues
```

So what is the best algorithm to choose from? Clearly the SVD is most economical in terms of lines of code. However, depending on the dimensions of matrix $\mathbf{U}$, the snapshot POD is often much faster. This is because the SVD algorithm still needs to compute the $n \times n$ matrix $\mathbf{U}^T\mathbf{U}$ to obtain its eigenvectors $\mathbf{R}$. Therefore, for most applications in experimental or numerical fluid dynamics (where $n > m$), the snapshot POD is the method of choice. In fact, the snapshot POD was specifically developed by Sirovich to speed up the computation algorithmically [15].

VII. Physical Interpretation of POD Modes

Wouldn't it be nice if the POD modes were a direct spatial representation of coherent structures in the flow? This was probably the original idea behind the introduction of POD in fluid dynamics. Unfortunately, this is generally not the case. Remember from our 2D example that the spatial modes are a measure of how the velocity is correlated at different points in the flow. While a zone of correlation can indeed represent a 'coherent structure' – especially when the corresponding energy is dominant compared to the other modes, in other words when the ellipsoid representing the data is 'shallow' – there are many POD modes that add up to the complete flow when multiplied by their time coefficients (Eq. 1). Some of these zones of correlation appear randomly often due to the turbulent nature of most practical flows, which means that these modes are simply a manifestation of the randomness of turbulence.

Nevertheless, the POD can be very useful because it enables us to reconstruct a flow with only a few of the most energetic modes in order to educe motions that are sometimes not easily spotted in the raw data. In the example of the turbulent separation bubble above, we saw by visualizing $\tilde{\mathbf{U}}^1$ that the first mode represents a contraction and expansion of the bubble. This is interesting for two reasons, first because this motion cannot easily be observed on the original data $\mathbf{U}$ due to the many scales of turbulence that are superimposed on it, and second because this motion is connected to a fairly large portion (20%) of the TKE, so we expect it to be an important practical characteristic of the flow.

[g] This is precisely why we have normalized $\mathbf{C_s}$ with $m-1$ in the snapshot POD. If $n-1$ had been used, as might have been expected, the eigenvalues of the snapshot method would not have been equal to those of the direct method.

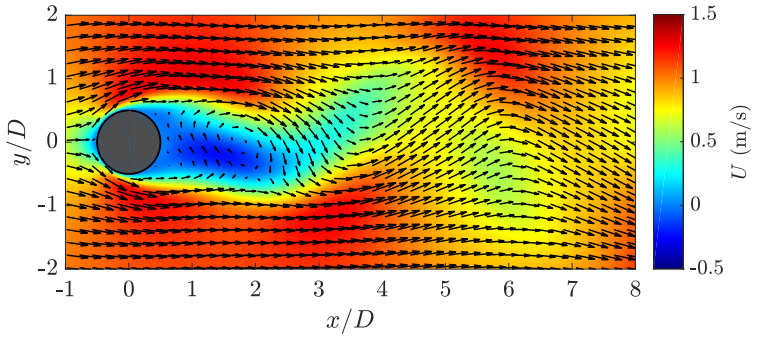

Figure 14. Snapshot of the flow field around a circular cylinder at $Re_D = 100$ [16].

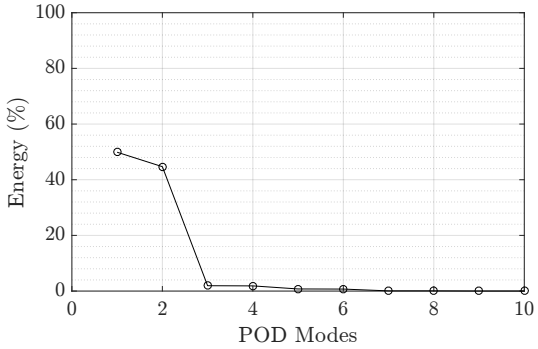

Figure 15. Contribution of the 10 first eigenvalues to the TKE for the 2D cylinder case.

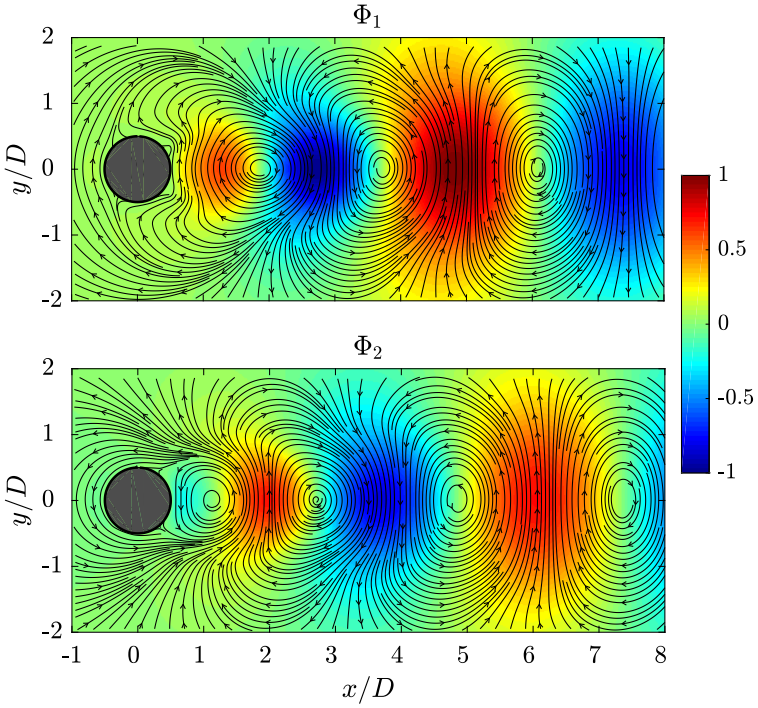

Figure 16. POD modes 1 and 2 for the 2D cylinder case. The color contours indicate the magnitude of the vertical velocity component.

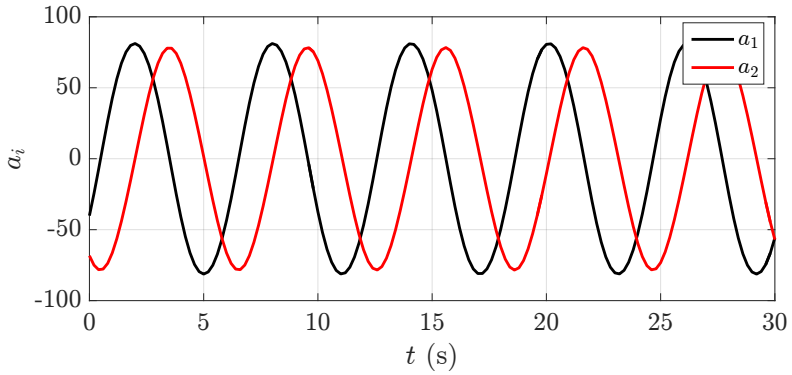

Figure 17. Time coefficients of the first 2 POD modes for the 2D cylinder case.

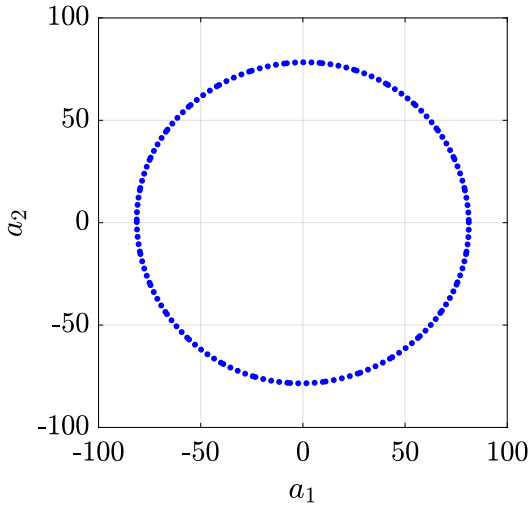

Figure 18. Phase portrait of the first 2 POD modes for the 2D cylinder case.

The POD may also be used to illustrate the convection of vortical structures. Fig. 14 shows a particular snapshot of the flow around a circular cylinder at a Reynolds number $Re_D = 100$. This data, obtained by numerical solution of the two-dimensional Navier-Stokes equations and made publicly available by Kutz *et al.* [16], shows the typical von Kármán street of alternating vortices in its wake. One hundred and fifty of these snapshots, sampled at regular time intervals on a 450×200 grid, are now used to compute the snapshot POD. Both the longitudinal and vertical velocity components are considered in the POD, which results in a matrix $\mathbf{U}$ containing 150 rows and $450 \times 200 \times 2 = 180000$ columns.

The contribution of the first 10 eigenvalues to the TKE (the 'POD spectrum') is shown in Fig. 15, where it can be seen that the first and second modes are largely dominant and contribute to about 50% and 45% of the kinetic energy, respectively. The two first spatial modes, $\mathbf{\Phi}_1$ and $\mathbf{\Phi}_2$ (the first two columns of matrix $\mathbf{\Phi}$), are plotted in Fig. 16. Clearly, both modes show the presence of structures resembling vortices in the flow, which is expected given the predominance of actual vortices in the cylinder's wake. Another interesting point is that mode 2 appears to be almost exactly the same as mode 1, except for a translation of approximately a quarter of a wavelength. The effect of this translation becomes apparent if we look at the time coefficients $a_1(t)$ and $a_2(t)$ of these modes (the first 2 columns of matrix $\mathbf{A}$), which are shown in Fig. 17. These are nearly sinusoidal and offset by a quarter of a period. Because of these offsets, when the two first spatial modes are combined with their time coefficients and added together, they reproduce the translating and alternating character of the vortices in the wake. In fact, to show that a pair of POD modes illustrate a periodic phenomenon, it is common to plot a phase portrait consisting of one time coefficient as a function

of the other. This phase portrait then forms a circular ring, as shown in Fig. 18 for the case of our cylinder wake flow. Treating the plane of Fig. 18 as a the complex plane, we clearly have $\|a_1(t) + ia_2(t)\| = \text{const}$, which implies that $a_1 = r\cos(\theta)$ and $a_2 = r\sin(\theta)$, with r a constant and θ the phase angle of the oscillation. This property can for example be used to obtain the phase information of a periodic flow without performing any time-resolved measurement (*e.g.*, [17]).

In the cylinder example, given that the first two modes account for more than 90% of the total kinetic energy, they are enough to reproduce the original flow quite accurately. In effect, in this particular case, a low-order model containing only the first two POD modes is a very good approximation of the original flow. However, the cylinder example may be seen as 'too simple' since coherent vortices are readily observed in the original snapshots. In fact, this is precisely why only two modes are sufficient to reproduce the salient features of the flow. In a more complex, possibly turbulent flow, many modes are generally necessary to account for 90% of the TKE and their interpretation is generally far from trivial. Nevertheless, the arguments used for the cylinder may be used for other flows as well. Looking back at Fig. 11, we see that the second and third modes in the TSB show the same general features, albeit to a lesser extent, as the first and second modes of the cylinder. Therefore, it is reasonable to expect convection of structures in the TSB as well. Indeed, while a low-order model of the TSB flow containing only modes 2 and 3 does indeed show some signs of convected structures, it is certainly not sufficient to reproduce realistic flow dynamics.

Generally speaking, the interpretation of POD modes is facilitated when a few of them account to a large portion of the TKE. This is similar to looking at peaks in a Fourier power spectrum, for example, except that the POD spectrum is necessarily decreasing and that the most dominant modes are the first ones by construction. Nevertheless, one should remain cautious in one's physical interpretation of the modes since they are nothing more than mathematical objects illustrating spatial zones of correlation.

Finally, it should be mentioned that POD doesn't necessarily have to be applied to velocity fields. In some instances, another variable, like the vorticity field in a flow or the greyscale level of schlieren images (for example) might be better suited to describe the data, or easier to acquire. The algorithms described above can be used directly once a relevant snapshot matrix is constructed. In the case of the vorticity, for example, the POD algorithms will maximize the enstrophy instead of the kinetic energy in order to create proper orthogonal modes of the vorticity field.

VIII. Conclusion

Because of the now ubiquitous availability of planar or volumetric velocity databases, the POD has become a classical tool in experimental and numerical fluid dynamics. Nevertheless, the method still appears to be surrounded by an aura of mystery for the non-specialist. I believe that this is because most texts describe it in terms of abstract functional spaces, which are not part of the usual mathematical toolbox of mechanical and aerospace engineers. In this tutorial, I have tried to demistify the POD by approaching it in the finite-dimensional case, and by restricting the analysis to a separation of variables in terms of space and time. The POD was also first illustrated with a 2D example, as is common in the field of statistics for existing tutorials on Principal Component Analysis, *e.g.* [8, 10]. I believe that this is useful to visualize the different projections that occur in the POD algorithm. While this approach might be seen as too restrictive or too simplistic for the specialist, I hope that it will be useful for students and researchers interested in learning about the method for applications in experimental or numerical fluid dynamics.

Acknowledgements

The author is indebted to Giuseppe Di Labbio for his detailed comments on an earlier version of the manuscript and for his optimization of the original MATLAB codes. Thanks also to Abdelouahab Mohammed-Taifour for his processing of the cylinder case.

References

[1] J. L. Lumley. The structure of inhomogeneous turbulent flows. *Atmospheric turbulence and radio wave propagation*, 1967.

[2] 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, 1993.

[3] A. Chatterjee. An introduction to the proper orthogonal decomposition. *Current Science*, 78(7):808–817, 2000.

[4] L. Cordier and M. Bergmann. Proper orthogonal decomposition: an overview. In *VKI Lecture Series 2003-04*, 2003.

[5] P. Holmes, J. L. Lumley, G. Berkooz, and C. W. Rowley. *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge University Press, 2012.

[6] W. K. George. A 50-year retrospective and the future. In *Whither Turbulence and Big Data in the 21st Century?*, pages 13–43. Springer, 2017.

[7] 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 Journal*, 55(12):4013–40419, 2017.

[8] J. Edward Jackson. *A User's Guide to Principal Components*. Wiley, 2003.

[9] J. Shlens. A tutorial on principal component analysis. *arXiv preprint arXiv:1404.1100*, 2014.

[10] L. Smith. A tutorial on principal component analysis. *http://facepress.net/details.php?id=734*, 2012.

[11] H. Chen, D. L. Reuss, D. LS Hung, and V. Sick. A practical guide for using proper orthogonal decomposition in engine research. *International Journal of Engine Research*, 14(4):307–319, 2013.

[12] J. N. Kutz. *Data-driven modeling & scientific computation: methods for complex systems & big data*. Oxford University Press, 2013.

[13] A. Mohammed-Taifour and J. Weiss. Unsteadiness in a large turbulent separation bubble. *Journal of Fluid Mechanics*, 799:383–412, 2016.

[14] E. Kreyszig. *Advanced Engineering Mathematics*. John Wiley & Sons, 2010.

[15] L. Sirovich. Turbulence and the dynamics of coherent structures. I. coherent structures. *Quarterly of Applied Mathematics*, 45(3):561–571, 1987.

[16] J. N. Kutz, S. L. Brunton, B. W. Brunton, and J. L. Proctor. *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*. SIAM, 2016.

[17] B. W. Van Oudheusden, F. Scarano, N. P. Van Hinsberg, and D. W. Watt. Phase-resolved characterization of vortex shedding in the near wake of a square-section cylinder at incidence. *Experiments in Fluids*, 39(1):86–98, 2005.