Chapter 2
Fluid Dynamics

DMD originated in the fluid dynamics community as a promising new technique to extract spatiotemporal coherent patterns from high-dimensional fluids data [247]. The subsequent connection between DMD modes and eigenvectors of the Koopman operator made the method even more promising as an approach to analyze data from a nonlinear dynamical system, such as the Navier–Stokes equations [235]. In the short time following these two seminal papers, DMD has been used extensively in fluid dynamics to investigate a wide range of flow phenomena.

Here, we will explore many of the advanced applications of DMD in fluid dynamics. In addition, we demonstrate the use of DMD on a large data set that is typical in fluid dynamics: a time series of vorticity field snapshots for the wake behind a circular cylinder at Reynolds number 100. DMD will be compared with POD, which is a workhorse data method in fluids [27, 133].

2.1 ▪ Modal decomposition in fluids

There is a rich history in fluid dynamics of innovations that extract relevant features from large-scale datasets. Although a fluid represents an immensely complex, high-dimensional dynamical system, it has long been observed that dominant patterns exist in the flow field, giving rise to large-scale coherent structures. These coherent structures were famously depicted by Leonardo da Vinci in the drawing shown in Figure 2.1. Efficiently characterizing these energetic structures from data, with either full or limited measurements, has been a central effort in making fluids more tractable to analysis, engineering design, and more recently control [133].

The Navier–Stokes equations represent a highly accurate partial differential equation model for a given fluid system, yet it is typically difficult to use these equations directly for engineering design. The need for an alternative description of fluid systems that represents flow interactions in the aggregate is nicely summarized by Richard Feynman in his lecture on fluid mechanics [93]:

> *The test of science is its ability to predict. Had you never visited the earth, could you predict the thunderstorms, the volcanos, the ocean waves, the auroras, and the colorful sunset?*

Coherent structure analysis has become central in fluid dynamics. Instead of considering every subtle detail in a flow and analyzing equations of motion in isolation,

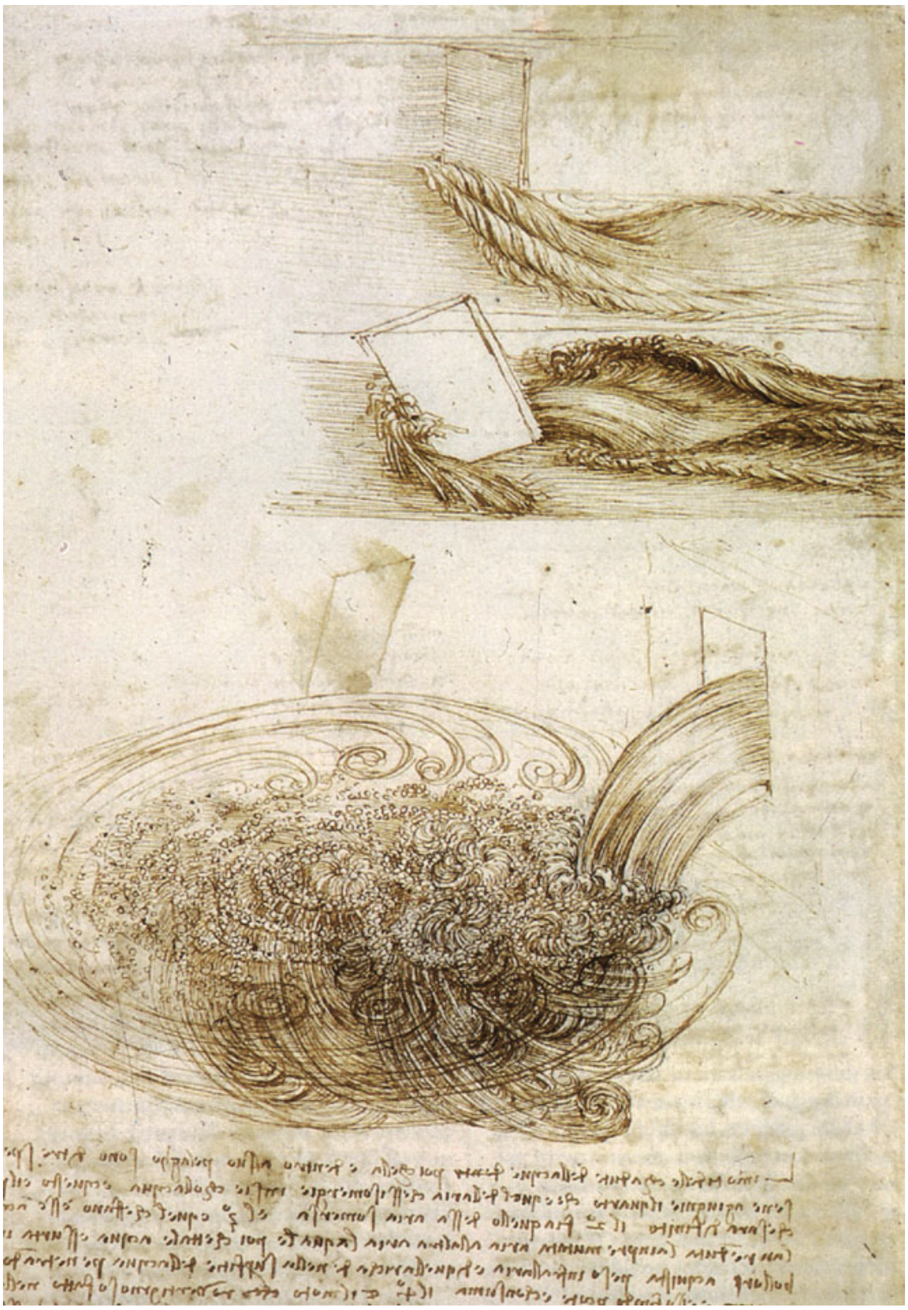


Figure 2.1. *Illustrations of turbulent mixing, by Leonardo da Vinci c.* 1508, *in "Studies of Water Passing Obstacles and Falling." Large-scale coherent structures can be observed in these sketches.*

data is collected from relevant flow configurations and synthesized into representative structures and a hierarchy of models to describe their interactions. POD [27, 133] is one of the earliest data-driven modal decompositions in fluids. It is a form of dimensionality reduction, often computed using SVD [114, 116, 261], that identifies the linear subspace that best spans the data. The method benefits from the fact that important directions in this subspace correspond to high-dimensional vectors that may be thought of as "eigen" flows. POD has been widely accepted because of its ease of interpretation and broad applicability to data from both simulations and experiments. ROMs have been especially important in obtaining computationally efficient representations suitable for closed-loop feedback control of high-dimensional fluid systems [46]. In many ways, these fundamental techniques in dimensionality reduction for fluids are among the first use of data science in complex systems.

2.1.1 ▪ Experimental and numerical flow measurements

In the past half century, the ability to collect and analyze large-scale measurements of fluid systems has rapidly advanced. These advances may roughly be categorized into computational techniques to simulate fluids and experimental techniques to measure flows in the laboratory.

With the advent of scientific computing, computational fluid dynamics (CFD) [125] has transformed the analysis of fluid systems, providing the unprecedented ability to noninvasively probe and measure fully resolved fluid velocity fields. There is a wide variety of numerical methods for flow simulation, including general multiphysics engines as well as methods that are highly tailored for a specific scenario. Direct numerical simulations (DNSs) numerically solve the Navier–Stokes equations, resolving all spatial and temporal scales. The largest DNSs are run on massively parallel computing architectures, and they resolve immensely complex flows. For example, a particularly large simulation of a wall-bounded turbulent channel flow involves over 10^{11} flow states and runs on nearly 10^6 processors in parallel [174]. In addition to providing full spatial flow measurements, CFD enables nonphysical numerical experiments, such as investigating the linearized Navier–Stokes equations or simulating adjoint equations for sensitivity analysis, uncertainty quantification, optimization, and control. These examples involve either simulating equations that are nonphysical or initializing arbitrary flow states, both of which are not possible in experiments.

In experimental fluid dynamics, the growth of laser technology has enabled increasingly detailed measurements, including the tracking of scalar quantities (heat, density, chemicals) as well as full velocity fields. Around the same time as the rise of CFD, laser Doppler velocimetry (LDV) [308, 96] was invented, resulting in noninvasive *point* velocity measurements. Later, the particle image velocimetry (PIV) [4] technique was introduced, resulting in full two-dimensional or three-dimensional velocity field measurements. PIV has quickly become a standard experimental technique, providing large quantities of flow data from real-world engineering fluids.

Both computational and experimental methods have relative strengths and weaknesses. CFD provides extremely high resolution flow data, but investigation of many engineering-scale flows is still decades away, even with the expected exponential increase in computing power over time. Moreover, CFD is often applied to idealized geometries and flow conditions, making it useful as a research and prototyping tool, but limited in the investigation of many physical systems. At the same time, experimental techniques, such as PIV, may be applied to extremely complex and realistic flows, but these methods are limited by data transfer rates as well as laser and camera technology. Thus, the spatial and temporal resolution of PIV systems remains limited.

As more complete measurements become available for more complicated fluid systems, the data associated with these investigations will become increasingly large and unwieldy. Data-driven methods are central to managing and analyzing this data, and, generally, we consider data methods to include techniques that apply equally well to data from numerical or experimental systems. This precludes any nonphysical knowledge of the system, including adjoints, arbitrary initial conditions, iterations through an evolution operator, etc.

2.1.2 ▪ Snapshot-based methods

When analyzing a time series of data containing either velocity or vorticity fields at a grid of spatial locations, it is often beneficial to use snapshot-based methods. First, two-dimensional or three-dimensional vector field data at time t_k is *flattened* into a single tall column vector:

$$\mathbf{x}(\mathbf{r}, t_k) = \begin{bmatrix} x(r_{1,1}, t_k) & x(r_{1,2}, t_k) & \cdots & x(r_{1,p}, t_k) \\ x(r_{2,1}, t_k) & x(r_{2,2}, t_k) & \cdots & x(r_{2,p}, t_k) \\ \vdots & \vdots & \ddots & \vdots \\ x(r_{q,1}, t_k) & x(r_{q,2}, t_k) & \cdots & x(r_{q,p}, t_k) \end{bmatrix} \tag{2.1}$$

$$\implies \quad \mathbf{x}_k = \begin{bmatrix} x(r_{1,1}, t_k) \\ x(r_{1,2}, t_k) \\ \vdots \\ x(r_{2,1}, t_k) \\ \vdots \\ x(r_{q,p}, t_k) \end{bmatrix}. \tag{2.2}$$

For this two-dimensional vector field example, x denotes the flow variable of interest, $\mathbf{r}$ denotes the spatial coordinate, and the index k denotes the kth time step. The vector $\mathbf{x}_k \in \mathbb{R}^n$ is called a *snapshot* of data. This removes the spatial relationship between neighbors, but it also allows snapshots at different times to be compared using vector inner products. The size n of a snapshot depends on the original size of the vector field as well as the number of flow variables of interest, easily consisting of hundreds of thousands, if not millions or billions, of states. These snapshots may be synthesized into a data matrix $\mathbf{X}$:

$$\mathbf{X} = \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_m \\ | & | & & | \end{bmatrix}. \tag{2.3}$$

This data matrix is a common starting place for snapshot-based methods. In fluid systems, the state dimension n is typically much larger than the number of snapshots m, so that $\mathbf{X}$ is a *tall and skinny* matrix. The case where spatial measurements are collected as column vectors differs from the statistics literature, where a collection of measurements in time are typically arranged as row vectors in a *short and fat* matrix.

2.1.3 ▪ POD

POD [27, 133] is a central method for dimensionality reduction in fluids, resulting in a hierarchical decomposition of flow data into an orthogonal basis of spatially correlated modes. POD is often formulated by taking the SVD [114, 54, 116, 119] of the data matrix $\mathbf{X}$:

$$\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*. \tag{2.4}$$

Note that since the state dimension n is much larger than the number of snapshots m, the rank of $\mathbf{X}$ is at most m, so that there is at most an $m \times m$ nonzero block of singular values Σ.

SVD is a powerful and numerically stable method of decomposing a data matrix. In practice, it is common to use the method of snapshots [261] to compute the leading terms in the SVD for high-dimensional data associated with fluid velocity or vorticity fields. In particular, it is possible to construct an $m \times m$ columnwise correlation matrix $\mathbf{X}^*\mathbf{X}$ consisting of inner products of the columns of $\mathbf{X}$. The eigendecomposition of this matrix is related to the SVD:

$$\mathbf{X}^*\mathbf{X} = \mathbf{V}\Sigma\mathbf{U}^*\mathbf{U}\Sigma\mathbf{V}^* = \mathbf{V}\Sigma^2\mathbf{V}^* \tag{2.5}$$

$$\implies \quad \mathbf{X}^*\mathbf{X}\mathbf{V} = \mathbf{V}\Sigma^2. \tag{2.6}$$

After computing $\mathbf{V}$ and Σ from the spectral decomposition of $\mathbf{X}^*\mathbf{X}$, it is possible to construct the leading m columns of $\mathbf{U}$ (i.e., POD modes) by

$$\mathbf{U} = \mathbf{X}\mathbf{V}\Sigma^{-1}. \tag{2.7}$$

In many cases, it is customary to subtract the mean of $\mathbf{X}$ (i.e., the mean velocity or vorticity field) before computing POD modes, in which case POD is equivalent to PCA from statistics [214, 147]. It is interesting to note that the method of snapshots was published by Sirovich in the same year as the breakthrough on eigenfaces by Sirovich and Kirby [262]. This paper provided a method to extract dominant patterns in human faces from data; facial recognition has since become a central machine-learning task.

POD separates the space and time correlations into the matrices $\mathbf{U}$ and $\mathbf{V}$, respectively. The matrix $\mathbf{U}$ only contains the spatial correlations in the data, whereas all temporal information is found in $\mathbf{V}$. Many methods, such as Galerkin projection, disregard the remaining information from Σ and $\mathbf{V}$. In fact, if only $\mathbf{U}$ is desired, POD may be computed on non-time-resolved data $\mathbf{X}$, where the $\mathbf{V}$ matrix is essentially meaningless. For time-resolved data, DMD capitalizes on the time correlations in $\mathbf{V}$ to rearrange the leading column of $\mathbf{U}$, resulting in the dominant patterns in the POD subspace that remain coherent in both space and time.

2.1.4 ▪ Galerkin projection of Navier–Stokes equations

Given an orthogonal basis from POD, it is possible to obtain a ROM of a dynamical system through Galerkin projection. In this approach, the state vector $\mathbf{x}$ is approximated by a finite sum of POD modes, and this expansion is substituted directly into the dynamical system. By the orthogonality of the POD coordinate system, it is possible to obtain an induced dynamical system on the POD mode coefficients. The resulting dynamical system typically has a much smaller dimension r compared with the dimension n of the state-space.

As an example, consider the incompressible Navier–Stokes equations, where $\mathbf{q}(\xi, t)$ represents a continuous vector field of velocity components of the fluid in space and time:

$$\frac{\partial \mathbf{q}}{\partial t} + (\mathbf{q} \cdot \nabla)\mathbf{q} = -\nabla p + \frac{1}{\mathrm{Re}}\nabla^2 \mathbf{q}, \tag{2.8a}$$

$$\nabla \cdot \mathbf{q} = 0. \tag{2.8b}$$

In this nondimensionalized formulation, the only parameter is the Reynolds number $\mathrm{Re} = LU/\nu$, where L is a length scale, U is a velocity scale, and ν is the kinematic fluid viscosity.

Written in coordinates, the Navier–Stokes equations become

$$\frac{\partial q_j}{\partial t} + \sum_i q_i \frac{\partial}{\partial \xi_i} q_j = -\frac{\partial p}{\partial \xi_j} + \frac{1}{\mathrm{Re}} \sum_i \frac{\partial^2}{\partial \xi_i^2} q_j, \tag{2.9a}$$

$$\sum_j \frac{\partial q_j}{\partial \xi_j} = 0. \tag{2.9b}$$

It is possible to write this as a dynamical system in terms of a high-dimensional discretization $\mathbf{x}$ of the continuous variable $\mathbf{q}$:

$$\dot{\mathbf{x}} = \mathbf{L}\mathbf{x} + \mathbf{Q}(\mathbf{x}, \mathbf{x}), \tag{2.10}$$

where $\mathbf{L}$ is a linear operator and $\mathbf{Q}$ is a bilinear operator. The quadratic nonlinearity corresponds to the convection term $(\mathbf{q}\cdot\nabla)\mathbf{q}$ in (2.8a).

The state $\mathbf{x}$ near an equilibrium $\bar{\mathbf{x}}$ may be approximated by expanding the velocity field as a sum of $\bar{\mathbf{x}}$ and the POD modes $\{\psi_i\}_{i=1}^r$ (ψ are the columns of $\mathbf{U}$ from (2.4)):

$$\mathbf{x} \approx \bar{\mathbf{x}} + \sum_{i=1}^r a_i \psi_i. \tag{2.11}$$

Typically, r is chosen to be large enough that the approximation is accurate, yet small enough that $r \ll n$.

Substituting the POD expansion (2.11) into (2.10) yields

$$\begin{aligned} \dot{a}_k \psi_k &= \mathbf{L}(\bar{\mathbf{x}} + a_i\psi_i) + \mathbf{Q}\left(\bar{\mathbf{x}} + a_i\psi_i, \bar{\mathbf{x}} + a_j\psi_j\right) \\ &= \mathbf{L}\bar{\mathbf{x}} + a_i\mathbf{L}\psi_i + \mathbf{Q}(\bar{\mathbf{x}},\bar{\mathbf{x}}) + a_j\mathbf{Q}(\bar{\mathbf{x}},\psi_j) + a_i\mathbf{Q}(\psi_i,\bar{\mathbf{x}}) + a_ia_j\mathbf{Q}(\psi_i,\psi_j) \\ &= \underbrace{\mathbf{L}\bar{\mathbf{x}} + \mathbf{Q}(\bar{\mathbf{x}},\bar{\mathbf{x}})}_{=0} + a_i \underbrace{[\mathbf{L}\psi_i + \mathbf{Q}(\bar{\mathbf{x}},\psi_i) + \mathbf{Q}(\psi_i,\bar{\mathbf{x}})]}_{=\tilde{\mathbf{L}}\psi_i} + a_ia_j\mathbf{Q}(\psi_i,\psi_j), \end{aligned} \tag{2.12}$$

where summation over the indices i and j is assumed.

Finally, because the modes ψ_i are orthonormal, we take the inner product of both sides with ψ_k:

$$\dot{a}_k = a_i\langle\tilde{\mathbf{L}}\psi_i, \psi_k\rangle + a_ia_j\langle\mathbf{Q}(\psi_i,\psi_j),\psi_k\rangle \tag{2.13a}$$

$$= \hat{\mathbf{L}}_{ik}a_i + \hat{\mathbf{Q}}_{ijk}a_ia_j, \tag{2.13b}$$

where $\hat{\mathbf{L}}_{ij}$ and $\hat{\mathbf{Q}}_{ijk}$ are new linear and bilinear operators on mode coefficients. The pressure term in (2.8a) disappears in this POD-Galerkin procedure because each of the POD modes satisfies incompressibility, and so the inner product of the pressure gradient with ψ_k vanishes.

Galerkin projection onto POD modes generates a ROM for a nonlinear dynamical system. These systems may be modified to include both an equilibrium and a mean flow in a mean-field approximation [207]. Despite the benefits, there are challenges associated with the classical POD-Galerkin method:

- The operators $\hat{\mathbf{L}}$ and $\hat{\mathbf{Q}}$ are *dense*, so that even for a relatively small dimension r, these Galerkin systems may be computationally expensive to simulate. In contrast, although a fluid state may be high dimensional, the associated equations are generally sparse, enabling fast numerical schemes such as spectral methods.
- POD-Galerkin systems are often unstable, and analyzing these nonlinear systems is difficult [231].
- Computing the system in (2.10) and the expansion in (2.12) requires significant computational machinery. These operators and inner products are usually computed using a working DNS code.

2.2 ▪ Applications of DMD in fluids

Since the introduction of the DMD algorithm [247] in the fluid dynamics community and its subsequent connection to nonlinear dynamical systems [235], DMD has become a widely used technique in fluid dynamics [248]. Two excellent reviews of the Koopman spectral theory are found in [52, 195]. The original presentation of DMD as a generalization of global stability analysis has framed many of the subsequent studies in fluid dynamics.

2.2.1 ▪ DMD to extract structure from fluids data

DMD has been applied to a wide variety of flow geometries (jets, cavity flow, wakes, channel flow, boundary layers, etc.) to study mixing, acoustics, and combustion, among other phenomena. In the original paper of Schmid [247], both a cavity flow and a jet were considered. In the original paper of Rowley et al. [235], a jet in crossflow was investigated. It is no surprise that DMD has subsequently been used widely in both cavity flows and jets. The following summary of applications of DMD in fluids is not exhaustive, as new examples are constantly being developed.

In a number of jet studies, POD and DMD were compared [23, 254]. In [249], Schlieren images of a helium jet were used with DMD; interestingly, this is an early demonstration that DMD could be used with photographic images in addition to quantitative vector fields. In [251], time-resolved tomographic PIV was used to investigate a jet, and a variant of DMD was introduced using a spatial variable in place of the time variable to analyze coherence with a given spatial wavenumber.

DMD has been used to study various aspects of cavity flows [247, 183, 19]. In particular, DMD has been used to study self-sustained oscillations arising from the unsteady boundary layer separation at the leading edge [253]. Time-resolved PIV measurements of an incompressible cavity [20] have also been used to compute DMD.

Wake flows have also been investigated using DMD. An early example involved a finite-thickness flat plate with elliptic leading edge and active blowing/suction to investigate flow separation. In particular, DMD was used to identify a mode associated with the frequency lock-on of a high-frequency separated shear layer instability associated with the separation bubble and the low-frequency wake modes [288]. The wake past a gurney flap has also been investigated [212]. In [13], the cylinder wake was analyzed using Koopman analysis, strengthening the connection to dynamical systems. Recently, flow past wind turbines has been investigated using DMD, including dynamic stall [87] and wake prediction [141].

DMD has also been applied to boundary layer flows, providing insight into rough walls [173], compressible turbulent boundary layer interaction with a fluttering panel

[210], and near-wall structures in a transitional boundary layer [244]. Turbulent channel flows have also been investigated [199].

In acoustics, DMD has been used to capture the near-field and far-field acoustics that result from instabilities observed in shear flows [264]. DMD has also been used to analyze nonnormal growth mechanisms in thermoacoustic interactions in a Rijke tube [190]. In combustion, DMD has been used to understand the coherent heat release in turbulent swirl flames [200] and to analyze a rocket combustor [138]. DMD has been compared with POD for reacting flows [239].

DMD has also been used to analyze more exotic flows, including a simulated model of a high-speed train [202]. Shock turbulent boundary layer interaction (STBLI) has also been investigated, and DMD was used to identify a pulsating separation bubble that is accompanied by shockwave motion [120]. DMD has also been used to study self-excited fluctuations in detonation waves [191]. Other problems include identifying hairpin vortices [275], decomposing the flow past a surface-mounted cube [203], modeling shallow-water equations [30], studying nanofluids past a square cylinder [243], and measuring the growth rate of instabilities in annular liquid sheets [85].

2.2.2 ▪ Model reduction and system identification

A major promise of DMD in fluids is the ability to synthesize data from simulations or experiments into accurate and computationally tractable ROMs. There have been many important advances in this direction [282], although many avenues are still being developed. In one example, the DMD algorithm was used to identify slow modes to more rapidly compute balanced POD models [287]. DMD was later related concretely to balanced model reduction via the ERA [182, 290]. The publicly available software package MODRED integrates numerous techniques for model reduction, including DMD [24]. Other advances include the use of DMD and POD to create a hybrid ROM for future-state prediction [300] and a DMD error analysis [86].

2.2.3 ▪ Control-oriented methods

Many data-driven methods for model reduction and system identification are specifically tailored for control. The balanced POD (BPOD) [299, 233] combines balanced truncation [201] with POD to yield balanced input-output dynamics and a bi-orthogonal set of modes based on high-dimensional data. This method relies on adjoint simulations and is therefore only applicable to numerical investigation. However, the ERA [151] was recently shown to yield equivalent balanced models purely from data, without the need for adjoint simulations [184]. ERA has deep connections with DMD that will be discussed in Chapter 7.

2.2.4 ▪ Innovations of DMD in fluid dynamics

Numerous innovations to the DMD method have been developed in the context of fluid dynamics. The integration of compressed sensing and fluids has largely centered around applications in DMD [149, 289, 48, 122]; this development will be discussed further in Chapter 9. The optimal mode decomposition [306] is an iterative procedure that simultaneously determines the low-rank dynamics and the optimal subspace to minimize system residual error. Mode selection was also explored in [64]. As mentioned earlier, DMD applications in the jet have resulted in innovations surrounding the use of images for DMD [249] and the substitution of a spatial variable in the role of the time variable in DMD [251].

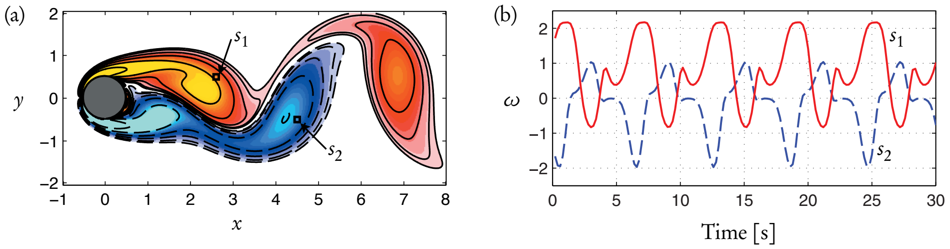


Figure 2.2. *Example of vorticity field* (a) *and two sensors* (b) *for the wake behind a cylinder at Re = 100. Algorithm* **2.3** *plots the vorticity field.*

2.2.5 ▪ Relationship to the Perron–Frobenius operator and almost-invariant sets

The Koopman operator is the dual to the Perron–Frobenius operator [79, 100, 102, 101], and there are deep mathematical connections between these operators. It is no surprise that DMD computations are related to the calculation of almost-invariant sets [102, 103, 274, 303], using set-oriented methods [79, 80] to identify eigenvectors of the Perron–Frobenius operator. Almost-invariant sets have been useful to understand sensitivity and coherence in fluid systems and also have relevance to uncertainty quantification.

2.3 ▪ Example: Re = 100 flow around a cylinder wake

In this example, we demonstrate DMD on a dataset representing a time series of fluid vorticity fields for the wake behind a circular cylinder at Reynolds number Re = 100. The Reynolds number quantifies the ratio of inertial to viscous forces and is defined as $\text{Re} = DU_\infty/\nu$, where D is the cylinder diameter, U_∞ is the free-stream velocity, and ν is the kinematic fluid viscosity. Re = 100 is chosen because it is larger than the critical Reynolds number $\text{Re}_{\text{crit}} \approx 47$ at which point the flow undergoes a supercritical Hopf bifurcation resulting in laminar vortex shedding [142, 313]. This limit cycle is stable and is representative of the three-dimensional flow [208, 207].

The two-dimensional Navier–Stokes equations are simulated using an immersed boundary projection method (IBPM) fluid solver[2] based on the fast multidomain method of Taira and Colonius [272, 70]. The computational domain comprises four grids that are nested so that the finest grid covers a domain of 9×4 and the largest grid covers a domain of 72×32, where lengths are nondimensionalized by the cylinder diameter; each grid contains 450×200 points, so that there are 50 points per cylinder diameter. We use a time step of $\Delta t = 0.02$, which is small enough to satisfy the CFL condition [71].

2.3.1 ▪ Snapshots of data

A snapshot of the cylinder wake data is shown in Figure 2.2. Contours of vorticity are shown on the left, and a time history of vorticity probe sensors s_1 and s_2 is shown on the right. After simulations converge to steady-state vortex shedding, we collect $m = 150$ snapshots at regular intervals in time, $10\Delta t$, sampling five periods of vortex

[2]The IBPM code used to generate data for this chapter is publicly available at https://github.com/cwrowley/ibpm.

shedding.[3] The period of vortex shedding is approximately $300\Delta t$, with $\Delta t = 0.02$, corresponding to a Strouhal number of about $St = fA/U_\infty = 0.16$, which is consistent with experimental results. Algorithms **2.1** and **2.2** load the data for this simulation.

Each vorticity field snapshot from the finest computational domain is reshaped into a large vector $\mathbf{x}_k$, and these vectors comprise columns of the matrices $\mathbf{X}_1^{m-1}$ and $\mathbf{X}_2^m$, as described in Chapter 1. Sampling a whole number of periods is important to recover symmetric solutions. It is also possible to augment the snapshot matrix with the negative vorticity fields, thereby enforcing symmetry.

ALGORITHM 2.1. **Load a single vorticity field from IBPM code (`loadIBPM.m`).**

```
function [X,Y,U,V,VORT] = loadIBPM(fname,nx,ny)

fileID = fopen(fname);  % open file
% remove first 6 lines of text in file
TEXT = textscan(fileID,'%s',6,'delimiter',char(460));

% pull out all data
FDATA = fscanf(fileID,'%f',[5,nx*ny]);
X = reshape(FDATA(1,:),nx,ny)';    % x positions
Y = reshape(FDATA(2,:),nx,ny)';    % y positions
U = reshape(FDATA(3,:),nx,ny)';    % u velocity
V = reshape(FDATA(4,:),nx,ny)';    % v velocity
VORT = reshape(FDATA(5,:),nx,ny)'; % vorticity
fclose all
```

ALGORITHM 2.2. **Load all vorticity fields from IBPM code (`loadDATA.m`).**

```
nx = 199;  % number of grid points in y-direction
ny = 449;  % number of grid points in x-direction

% create space for 150 snapshots
VORTALL = zeros(nx*ny,150); % vorticity

% extract data from 150 snapshots files
for count=1:150
    num = count*10; % load every 10th file
    % load file
    fname = ['ibpm',num2str(num,'%05d'),'.plt'];
    [X,Y,U,V,VORT] = loadIBPM(fname,nx,ny);
    VORTALL(:,count) = reshape(VORT,nx*ny,1);
end
```

ALGORITHM 2.3. **Plot vorticity field for cylinder wake (`plotCylinder.m`).**

```
function f1 = plotCylinder(VORT,ny,nx)

f1 = figure
vortmin = -5;  % only plot what is in -5 to 5 range
vortmax = 5;
VORT(VORT>vortmax) = vortmax;  % cutoff at vortmax
```

[3]The data collected for this example may be downloaded without running the IBPM code at www.siam.org/books/dmd.

```
VORT(VORT<vortmin) = vortmin;  % cutoff at vortmin

imagesc(VORT); % plot vorticity field
load CCcool.mat
colormap(CC);  % use custom colormap

% clean up axes
set(gca,'XTick',[1 50 100 150 200 250 300 350 400 449],'
    XTickLabel',{'-1','0','1','2','3','4','5','6','7','8'})
set(gca,'YTick',[1 50 100 150 199],'YTickLabel',{'2','1','0','-1
    ','-2'});
set(gcf,'Position',[100 100 300 130])
axis equal
hold on

% add contour lines (positive = solid, negative = dotted)
contour(VORT,[-5.5:.5:-.5 -.25 -.125],':k','LineWidth',1.2)
contour(VORT,[.125 .25 .5:.5:5.5],'-k','LineWidth',1.2)

theta = (1:100)/100'*2*pi;
x = 49+25*sin(theta);
y = 99+25*cos(theta);
fill(x,y,[.3 .3 .3])  % place cylinder
plot(x,y,'k','LineWidth',1.2) % cylinder boundary

set(gcf,'PaperPositionMode','auto') %
print('-depsc2', '-loose', 'figures/cylinder'); % eps are vector
     images
```

2.3.2 ▪ POD modes for the cylinder wake

Figure 2.3 shows the POD for the cylinder wake data described above. In this example, the POD modes come in energetic pairs, as shown in panel (b), which depicts the singular values. Although the SVD is a separation of variables resulting in a spatiotemporal decomposition, it is able to approximate the relevant structures required for the traveling wave solution to the cylinder wake. Note that POD may be applied to fluid velocity field or vorticity field data; in this case, we are using vorticity fields.

ALGORITHM **2.4. Compute POD modes for cylinder wake (`computePOD.m`).**

```
X = VORTALL;
Y = [X X];
%% augment matrix with mirror images to enforce symmetry/
    antisymmetry
for k=1:size(X,2)
    xflip = reshape(flipud(reshape(X(:,k),nx,ny)),nx*ny,1);
    Y(:,k+size(X,2)) = -xflip;
end

%% compute mean and subtract;
VORTavg = mean(Y,2);
f1 = plotCylinder(VORTavg,nx,ny);  % plot average wake
```

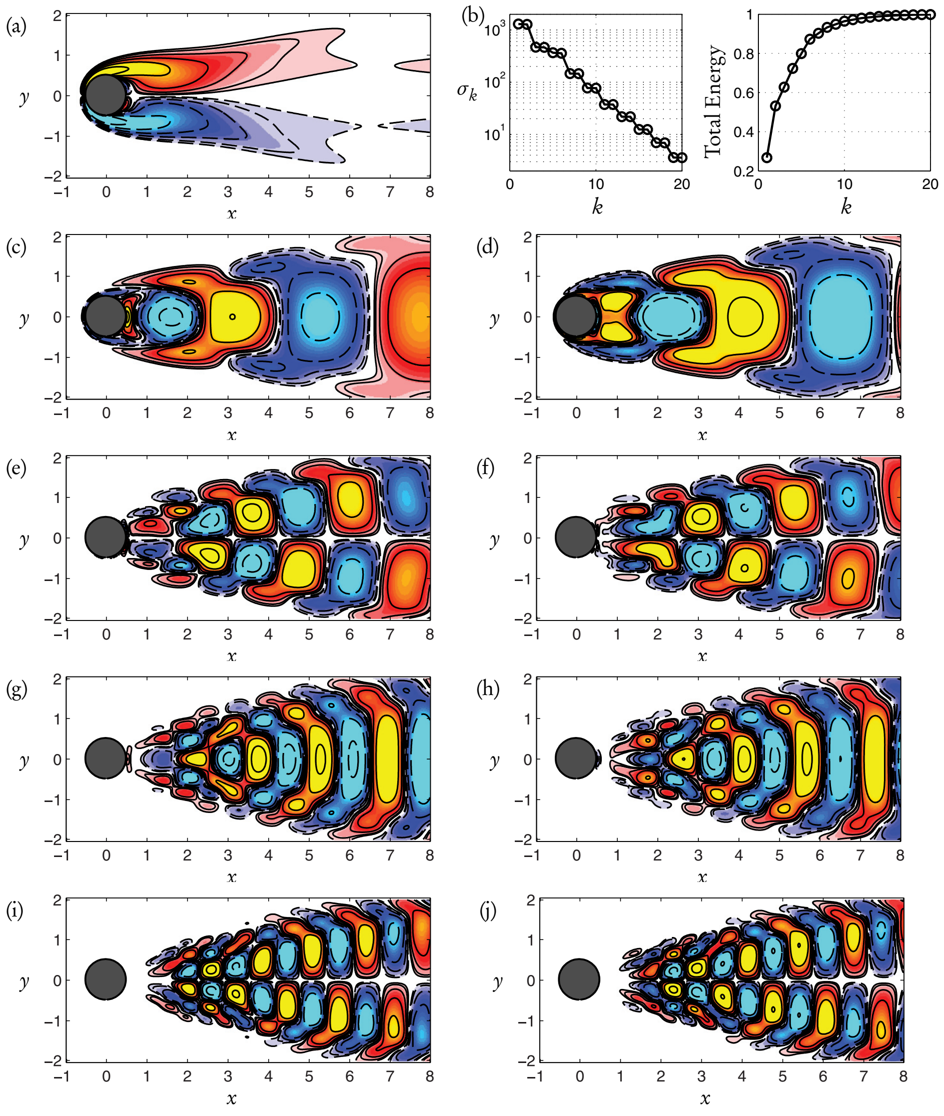


Figure 2.3. *Mean vorticity field* (a) *and POD singular values* (b) *for the wake behind a cylinder at* $Re = 100$. *The first* 8 *POD modes are shown in* (c)–(j).

```
%% compute POD after subtracting mean (i.e., do PCA)
[PSI,S,V] = svd(Y-VORTavg*ones(1,size(Y,2)),'econ');
% PSI are POD modes
semilogy(diag(S)./sum(diag(S))); % plot singular vals
```

2.3.3 ▪ DMD of the cylinder wake

Applying DMD to the cylinder wake requires the same basic snapshot information as POD, making this a data-driven method, since it applies equally well to data from simulations or experiments. Figure 2.4 shows a schematic of the DMD algorithm. A more detailed analysis of the Koopman analysis applied to the cylinder wake may be found in [13].

Unlike POD, DMD provides not only modes but also a set of associated eigenvalues that determine a low-dimensional dynamical system for how the mode amplitudes evolve in time. Moreover, unlike POD, the mean is not subtracted in the DMD calculation, and the first mode (shown as mode 1 in the middle panel), corresponds to a background mode that is not changing (i.e., it has zero eigenvalue). The DMD modes look similar to the POD modes for this example because the POD provides a harmonic decomposition, so that the modal amplitudes are approximately sinusoidal in time at harmonic frequencies of the dominant vortex shedding.

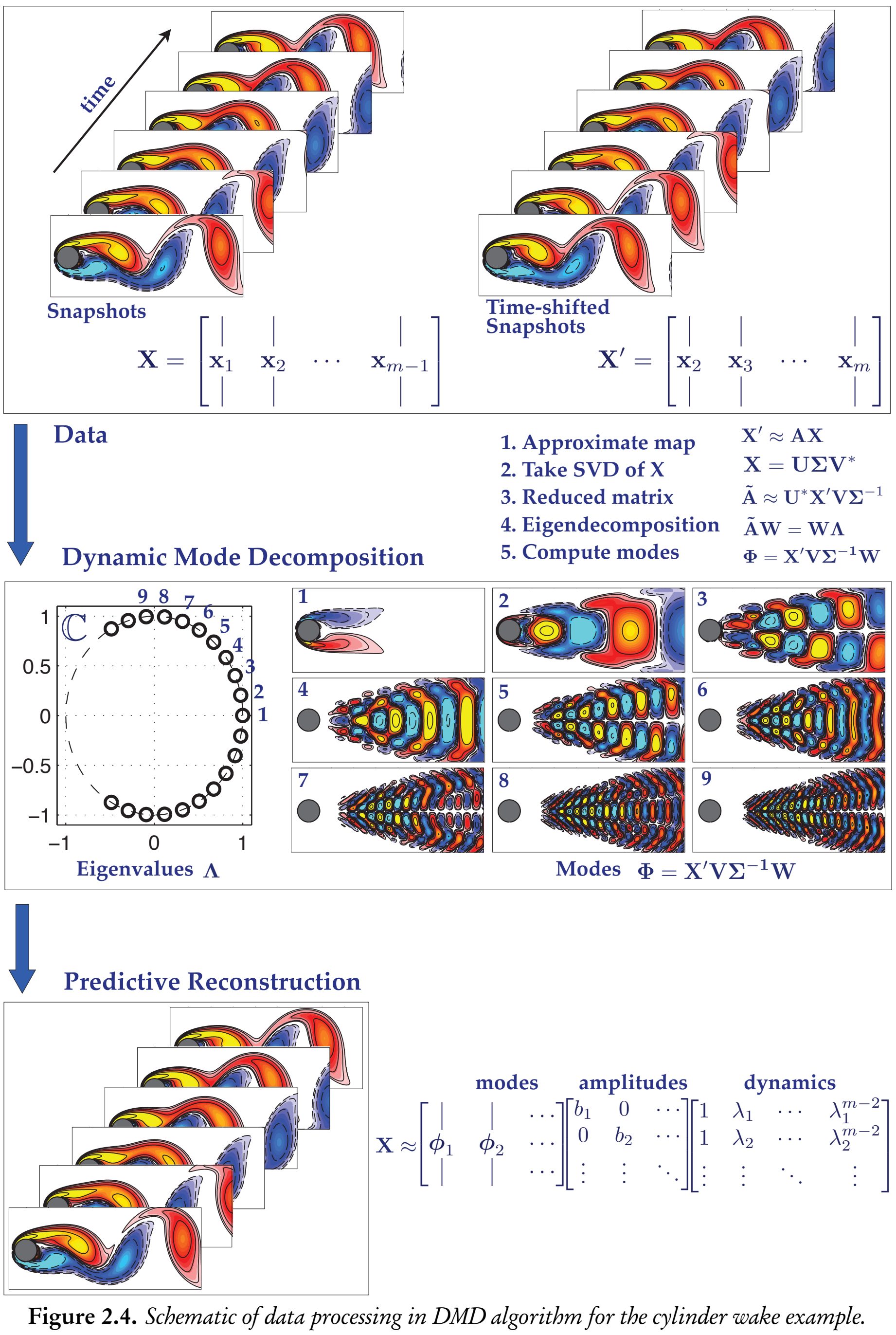
ALGORITHM **2.5. Compute DMD modes for cylinder wake (`computeDMD.m`).**

```
X = VORTALL(:,1:end-1);
X2 = VORTALL(:,2:end);
[U,S,V] = svd(X,'econ');

%%  Compute DMD (Phi are eigenvectors)
r = 21;  % truncate at 21 modes
U = U(:,1:r);
S = S(1:r,1:r);
V = V(:,1:r);
Atilde = U'*X2*V*inv(S);
[W,eigs] = eig(Atilde);
Phi = X2*V*inv(S)*W;

%% Plot DMD modes
for i=10:2:20
    plotCylinder(reshape(real(Phi(:,i)),nx,ny),nx,ny);
    plotCylinder(reshape(imag(Phi(:,i)),nx,ny),nx,ny);
end

%%  Plot DMD spectrum
figure
theta = (0:1:100)*2*pi/100;
plot(cos(theta),sin(theta),'k--') % plot unit circle
hold on, grid on
scatter(real(diag(eigs)),imag(diag(eigs)),'ok')
axis([-1.1 1.1 -1.1 1.1]);
```



time
Snapshots
$\mathbf{X} = \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_{m-1} \\ | & | & & | \end{bmatrix}$
Time-shifted Snapshots
$\mathbf{X}' = \begin{bmatrix} | & | & & | \\ \mathbf{x}_2 & \mathbf{x}_3 & \cdots & \mathbf{x}_m \\ | & | & & | \end{bmatrix}$
Data
1. Approximate map $\mathbf{X}' \approx \mathbf{A}\mathbf{X}$
2. Take SVD of X $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^*$
3. Reduced matrix $\tilde{\mathbf{A}} \approx \mathbf{U}^*\mathbf{X}'\mathbf{V}\mathbf{\Sigma}^{-1}$
4. Eigendecomposition $\tilde{\mathbf{A}}\mathbf{W} = \mathbf{W}\mathbf{\Lambda}$
5. Compute modes $\mathbf{\Phi} = \mathbf{X}'\mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{W}$
Dynamic Mode Decomposition
$\mathbb{C}$
1 2 3 4 5 6 7 8 9
1 0.5 0 −0.5 −1
−1 0 1
Eigenvalues $\mathbf{\Lambda}$
Modes $\mathbf{\Phi} = \mathbf{X}'\mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{W}$
Predictive Reconstruction
modes amplitudes dynamics
$\mathbf{X} \approx \begin{bmatrix} | & | & \cdots \\ \phi_1 & \phi_2 & \cdots \\ | & | & \cdots \end{bmatrix} \begin{bmatrix} b_1 & 0 & \cdots \\ 0 & b_2 & \cdots \\ \vdots & \vdots & \ddots \end{bmatrix} \begin{bmatrix} 1 & \lambda_1 & \cdots & \lambda_1^{m-2} \\ 1 & \lambda_2 & \cdots & \lambda_2^{m-2} \\ \vdots & \vdots & \ddots & \vdots \end{bmatrix}$

Figure 2.4. *Schematic of data processing in DMD algorithm for the cylinder wake example.*