11 Reduced Order Models (ROMs)

The proper orthogonal decomposition (POD) is the SVD algorithm applied to partial differential equations (PDEs). As such, it is one of the most important dimensionality reduction techniques available to study complex, spatio-temporal systems. Such systems are typically exemplified by nonlinear partial differential equations that prescribe the evolution in time and space of the quantities of interest in a given physical, engineering and/or biological system. The success of the POD is related to the seemingly ubiquitous observation that in most complex systems, meaningful behaviors are encoded in low-dimensional patterns of dynamic activity. The POD technique seeks to take advantage of this fact in order to produce low-rank dynamical systems capable of accurately modeling the full spatio-temporal evolution of the governing complex system. Specifically, *reduced order models* (ROMs) leverage POD modes for projecting PDE dynamics to low-rank subspaces where simulations of the governing PDE model can be more readily evaluated. Importantly, the low-rank models produced by the ROM allow for significant improvements in computational speed, potentially enabling prohibitively expensive Monte-Carlo simulations of PDE systems, optimization over parametrized PDE systems, and/or real-time control of PDE-based systems. POD has been extensively used in the fluids dynamics community [251]. It has also found a wide variety of applications in structural mechanics and vibrational analysis [287, 23, 232, 329], optical and MEMS technologies [333, 488], atmospheric sciences (where it is called empirical orthogonal functions (EOFs)) [116, 117], wind engineering applications [494], acoustics [181], and neuroscience [33, 519, 284]. The success of the method relies on its ability to provide physically interpretable spatio-temporal decompositions of data [316, 57, 181, 286, 126, 333].

11.1 POD for Partial Differential Equations

Throughout the engineering, physical and biological sciences, many systems are known to have prescribed relationships between time and space that drive patterns of dynamical activity. Even simple spatio-temporal relationships can lead to highly complex, yet coherent, dynamics that motivate the main thrust of analytic and computational studies. Modeling efforts seek to derive these spatio-temporal relationships either through first principle laws or through well-reasoned conjectures about existing relationships, thus leading generally to an underlying partial differential equation (PDE) that constrains and governs the complex system. Typically, such PDEs are beyond our ability to solve analytically. As a result, two primary solution strategies are pursued: computation and/or asymptotic reduction. In the former, the complex system is discretized in space and time to artificially produce an extremely high-dimensional system of equations which can be solved

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

to a desired level of accuracy, with higher accuracy requiring a larger dimension of the discretized system. In this technique, the high-dimensionality is artificial and simply a consequence of the underlying numerical solution scheme. In contrast, asymptotic reduction seeks to replace the complex system with a simpler set of equations, preferably that are *linear* so as to be amenable to analysis. Before the 1960s and the rise of computation, such asymptotic reductions formed the backbone of applied mathematics in fields such a fluid dynamics. Indeed, asymptotics form the basis of the earliest efforts of dimensionality reduction. Asymptotic methods are not covered in this book, but the computational methods that enable reduced order models are.

To be more mathematically precise about our study of complex systems, we consider generically a system of nonlinear PDEs of a single spatial variable that can be modeled as

$$\mathbf{u}_t = \mathbf{N}\left(\mathbf{u}, \mathbf{u}_x, \mathbf{u}_{xx}, \cdots, x, t; \boldsymbol{\beta}\right) \tag{11.1}$$

where the subscripts denote partial differentiation and $\mathbf{N}(\cdot)$ prescribes the generically nonlinear evolution. The parameter $\boldsymbol{\beta}$ will represent a bifurcation parameter for our later considerations. Further, associated with (11.1) are a set of initial and boundary conditions on a domain $x \in [-L, L]$. Historically, a number of analytic solution techniques have been devised to study (11.1). Typically the aim of such methods is to reduce the PDE (11.1) to a set of ordinary differential equations (ODEs). The standard PDE methods of *separation of variables* and *similarity solutions* are constructed for this express purpose. Once in the form of an ODE, a broader variety of analytic methods can be applied along with a *qualitative theory* in the case of nonlinear behavior [252]. This again highlights the role that *asymptotics* can play in characterizing behavior.

Although a number of potential solution strategies have been mentioned, (11.1) does not admit a closed form solution in general. Even the simplest nonlinearity or a spatially dependent coefficient can render the standard analytic solution strategies useless. However, computational strategies for solving (11.1) are abundant and have provided transformative insights across the physical, engineering and biological sciences. The various computational techniques devised lead to a approximate numerical solution of (11.1), which is of high-dimension. Consider, for instance, a standard spatial discretization of (11.1) whereby the spatial variable x is evaluated at $n \gg 1$ points

$$\mathbf{u}(x_k, t) \quad \text{for} \quad k = 1, 2, \cdots, n \tag{11.2}$$

with spacing $\Delta x = x_{k+1} - x_k = 2L/n$. Using standard finite-difference formulas, spatial derivatives can be evaluated using neighboring spatial points so that, for instance,

$$\mathbf{u}_x = \frac{\mathbf{u}(x_{k+1}, t) - \mathbf{u}(x_{k-1}, t)}{2\Delta x} \tag{11.3a}$$

$$\mathbf{u}_{xx} = \frac{\mathbf{u}(x_{k+1}, t) - 2\mathbf{u}(x_k, t) + \mathbf{u}(x_{k-1}, t)}{\Delta x^2}\,. \tag{11.3b}$$

Such spatial discretization transforms the governing PDE (11.1) into a set of n ODEs

$$\frac{d\mathbf{u}_k}{dt} = \mathbf{N}\left(\mathbf{u}(x_{k+1}, t), \mathbf{u}(x_k, t), \mathbf{u}(x_{k-1}, t), \cdots, x_k, t, \boldsymbol{\beta}\right)\,, \quad k = 1, 2, \cdots, n. \tag{11.4}$$

This process of discretization produces a more manageable system of equations at the expense of rendering (11.1) high-dimensional. It should be noted that as accuracy requirements become more stringent, the resulting dimension n of the system (11.4) also increases,

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

since $\Delta x = 2L/n$. Thus, the dimension of the underlying computational scheme is artificially determined by the accuracy of the finite-difference differentiation schemes.

The spatial discretization of (11.1) illustrates how high-dimensional systems are rendered. The artificial production of high-dimensional systems is ubiquitous across computational schemes and presents significant challenges for scientific computing efforts. To further illustrate this phenomenon, we consider a second computational scheme for solving (11.1). In particular, we consider the most common technique for analytically solving PDEs: separation of variables. In this method, a solution is assumed, whereby space and time are independent, so that

$$\mathbf{u}(x,t) = \mathbf{a}(t)\psi(x) \tag{11.5}$$

where the variable $\mathbf{a}(t)$ subsumes all the time dependence of (11.1) and $\psi(x)$ characterizes the spatial dependence. Separation of variables is only guaranteed to work analytically if (11.1) is linear with constant coefficients. In that restrictive case, two differential equations can be derived that separately characterize the spatial and temporal dependences of the complex system. The differential equations are related by a constant parameter that is present in each.

For the general form of (11.1), separation of variables can be used to yield a computational algorithm capable of producing accurate solutions. Since the spatial solutions are not known *a priori*, it is typical to assume a set of basis modes which are used to construct $\psi(x)$. Indeed, such assumptions on basis modes underlies the critical ideas of the method of *eigenfunction expansions*. This yields a separation of variables solution ansatz of the form

$$\mathbf{u}(x,t) = \sum_{k=1}^{n} \mathbf{a}_k(t)\psi_k(x) \tag{11.6}$$

where $\psi_k(x)$ form a set of $n \gg 1$ basis modes. As before, this expansion artificially renders a high dimensional system of equations since n modes are required. This separation of variables solution approximates the true solution, provided n is large enough. Increasing the number of modes n is equivalent to increasing the spatial discretization in a finite-difference scheme.

The orthogonality properties of the basis functions $\psi_k(x)$ enable us to make use of (11.6). To illustrate this, consider a scalar version of (11.1) with the associated scalar separable solution $u(x,t) = \sum_{k=1}^{n} a_k(t)\psi_k(x)$. Inserting this solution into the governing equations gives

$$\sum \psi_k \frac{da_k}{dt} = \mathbf{N}\left(\sum a_k\psi_k, \sum a_k(\psi_k)_x, \sum a_k(\psi_k)_{xx}, \cdots, x, t, \boldsymbol{\beta}\right) \tag{11.7}$$

where the sums are from $k = 1, 2, \cdots, n$. Orthogonality of our basis functions implies that

$$\langle \psi_k, \psi_j \rangle = \delta_{kj} = \begin{cases} 0 & j \neq k \\ 1 & j = k \end{cases} \tag{11.8}$$

where δ_{kj} is the Kronecker delta function and $\langle \psi_k, \psi_j \rangle$ is the inner product defined as:

$$\langle \psi_k, \psi_j \rangle = \int_{-L}^{L} \psi_k \psi_j^* dx \tag{11.9}$$

where * denotes complex conjugation.

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

Once the modal basis is decided on, the governing equations for the $a_k(t)$ can be determined by multiplying (11.7) by $\psi_j(x)$ and integrating from $x \in [-L, L]$. Orthogonality then results in the temporal governing equations, or Galerkin projected dynamics, for each mode

$$\frac{da_k}{dt} = \left\langle \mathbf{N}\left(\sum a_j\psi_j, \sum a_j(\psi_j)_x, \sum a_j(\psi_j)_{xx}, \cdots, x, t, \boldsymbol{\beta}\right), \psi_k \right\rangle \quad k = 1, 2, \cdots, n. \tag{11.10}$$

The given form of $\mathbf{N}(\cdot)$ determines the mode-coupling that occurs between the various n modes. Indeed, the hallmark feature of nonlinearity is the production of modal mixing from (11.10).

Numerical schemes based on the Galerkin projection (11.10) are commonly used to perform simulations of the full governing system (11.1). Convergence to the true solution can be accomplished by both judicious choice of the modal basis elements ψ_k as well as the total number of modes n. Interestingly, the separation of variables strategy, which is rooted in *linear* PDEs, works for *nonlinear* and *nonconstant coefficient* PDEs, provided enough modal basis functions are chosen in order to accommodate all the nonlinear mode mixing that occurs in (11.10). A good choice of modal basis elements allows for a smaller set of n modes to be chosen to achieve a desired accuracy. The POD method is designed to specifically address the data-driven selection of a set of basis modes that are tailored to the particular dynamics, geometry, and parameters.

Fourier Mode Expansion

The most prolific basis used for the Galerkin projection technique is Fourier modes. More precisely, the fast Fourier transform (FFT) and its variants have dominated scientific computing applied to the engineering, physical, and biological sciences. There are two primary reasons for this: (i) There is a strong intuition developed around the meaning of Fourier modes as it directly relates to spatial wavelengths and frequencies, and more importantly, (ii) the algorithm necessary to compute the right-hand side of (11.10) can be executed in $O(n \log n)$ operations. The second fact has made the FFT one of the top ten algorithms of the last century and a foundational cornerstone of scientific computing.

The Fourier mode basis elements are given by

$$\psi_k(x) = \frac{1}{L}\exp\left(i\frac{2\pi kx}{L}\right) \quad x \in [0, L] \text{ and } k = -n/2, \cdots, -1, 0, 1, \cdots, n/2 - 1\,. \tag{11.11}$$

It should be noted that in most software packages, including Matlab, the FFT command assumes that the spatial interval is $x \in [0, 2\pi]$. Thus one must rescale a domain of length L to 2π before using the FFT.

Obviously the Fourier modes (11.11) are complex periodic functions on the interval $x \in [0, L]$. However, they are applicable to a much broader class of functions that are not necessarily periodic. For instance, consider a localized Gaussian function

$$u(x, t) = \exp\left(-\sigma x^2\right) \tag{11.12}$$

whose Fourier transform is also a Gaussian. In representing such a function with Fourier modes, a large number of modes are often required since the function itself isn't periodic. Fig. 11.1 shows the Fourier mode representation of the Gaussian for three values of σ. Of note is the fact that a large number of modes is required to represent this simple function,

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

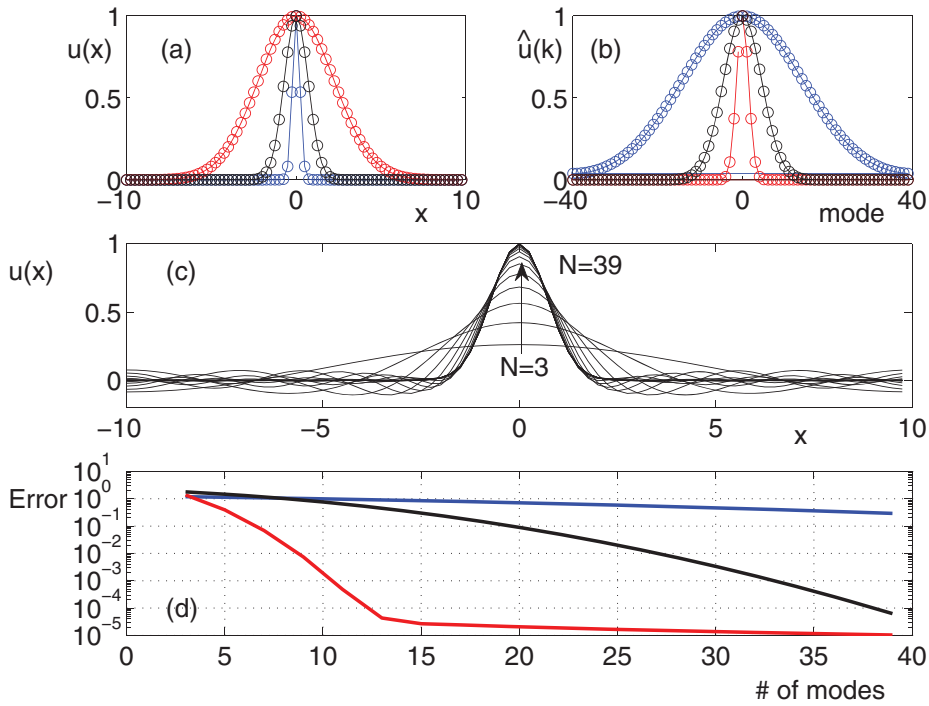

Figure 11.1 Illustration of Fourier modes for representing a localized Gaussian pulse. (a) $n = 80$ Fourier modes are used to represent the Gaussian $u(x) = \exp(-\sigma x^2)$ in the domain $x \in [-10, 10]$ for $\sigma = 0.1$ (red), $\sigma = 1$ (black) and $\sigma = 10$ (blue). (b) The Fourier mode representation of the Gaussian, showing the modes required for an accurate representation of the localized function. (c) The convergence of the n mode solution to the actual Gaussian ($\sigma = 1$) with the (d) L^2 error from the true solution for the three values of σ.

especially as the Gaussian width is decreased. Although the FFT algorithm is extremely fast and widely applied, one can see immediately that a large number of modes are generically required to represent simple functions of interest. Thus, solving problems using the FFT often requires high-dimensional representations (i.e., $n \gg 1$) to accommodate generic, localized spatial behaviors. Ultimately, our aim is to move away from artificially creating such high-dimensional problems.

Special Functions and Sturm-Liouville Theory

In the 1800s and early 1900s, mathematical physics developed many of the governing principles behind heat flow, electromagnetism and quantum mechanics, for instance. Many of the hallmark problems considered were driven by *linear* dynamics, allowing for analytically tractable solutions. And since these problems arose before the advent of computing, nonlinearities were typically treated as perturbations to an underlying linear equation. Thus one often considered complex systems of the form

$$\mathbf{u}_t = \mathbf{L}\mathbf{u} + \epsilon \mathbf{N}\left(\mathbf{u}, \mathbf{u}_x, \mathbf{u}_{xx}, \cdots, x, t, \boldsymbol{\beta}\right) \tag{11.13}$$

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

where $\mathbf{L}$ is a linear operator and $\epsilon \ll 1$ is a small parameter used for perturbation calculations. Often in mathematical physics, the operator $\mathbf{L}$ is a Sturm-Liouville operator which guarantees many advantageous properties of the eigenvalues and eigenfunctions.

To solve equations of the form in (11.13), special modes are often used that are ideally suited for the problem. Such modes are eigenfunctions of the underlying linear operator L in (11.13):

$$\mathbf{L}\psi_k = \lambda_k \psi_k \tag{11.14}$$

where $\psi_k(x)$ are orthonormal eigenfunctions of the operator $\mathbf{L}$. The eigenfunctions allow for an *eigenfunction expansion* solution whereby $\mathbf{u}(x,t) = \sum a_k(t)\psi_k(x)$. This leads to the following solution form

$$\frac{da_k}{dt} = \langle \mathbf{Lu}, \psi_k \rangle + \epsilon \langle \mathbf{N}, \psi_k \rangle \ . \tag{11.15}$$

The key idea in such an expansion is that the eigenfunctions presumably are ideal for modeling the spatial variations particular to the problem under consideration. Thus, they would seem to be ideal, or perfectly suited, modes for (11.13). This is in contrast to the Fourier mode expansion, as the sinusoidal modes may be unrelated to the particular physics or symmetries in the geometry. For example, the Gaussian example considered can be potentially represented more efficiently by Gauss-Hermite polynomials. Indeed, the wide variety of special functions, including the Sturm-Liouville operators of *Bessel*, *Laguerre*, *Hermite*, *Legendre*, for instance, are aimed at making the representation of solutions more efficient and much more closely related to the underlying physics and geometry. Ultimately, one can think of using such functions as a way of doing *dimensionality reduction* by using an ideally suited set of basis functions.

Dimensionality Reduction

The examples above and solution methods for PDEs illustrate a common problem of scientific computing: the generation of n degree, high-dimensional systems. For many complex PDEs with several spatial dimensions, it is not uncommon for discretization or modal expansion techniques to yield systems of differential equations with millions or billions of degrees of freedom. Such large systems are extremely demanding for even the latest computational architectures, limiting accuracies and run-times in the modeling of many complex systems, such as high Reynolds number fluid flows.

To aid in computation, the selection of a set of optimal basis modes is critical, as it can greatly reduce the number of differential equations generated. Many solution techniques involve the solution of a linear system of size n, which generically involves $O(n^3)$ operations. Thus, reducing n is of paramount importance. One can already see that even in the 1800s and early 1900s, the special functions developed for various problems of mathematical physics were an analytic attempt to generate an ideal set of modes for representing the dynamics of the complex system. However, for strongly nonlinear, complex systems (11.1), even such special functions rarely give the best set of modes. In the next section, we show how one might generate modes ψ_k that are tailored specifically for the dynamics and geometry in (11.1). Based on the SVD algorithm, the *proper orthogonal decomposition* (POD) generates a set of modes that are *optimal* for representing either simulation or

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

measurement data, potentially allowing for significant reduction of the number of modes n required to model the behavior of (11.1) for a given accuracy [57, 542, 543].

11.2 Optimal Basis Elements: The POD Expansion

As illustrated in the previous section, the selection of a good modal basis for solving (11.1) using the Galerkin expansion in (11.6) is critical for efficient scientific computing strategies. Many algorithms for solving PDEs rely on choosing basis modes *a priori* based on (i) computational speed, (ii) accuracy, and/or (iii) constraints on boundary conditions. All these reasons are justified and form the basis of computationally sound methods. However, our primary concern in this chapter is in selecting a method that allows for maximal computational efficiency via *dimensionality reduction*. As already highlighted, many algorithms generate artificially large systems of size n. In what follows, we present a data-driven strategy, whereby optimal modes, also known as POD modes, are selected from numerical and/or experimental observations, thus allowing for a minimal number of modes $r \ll n$ to characterize the dynamics of (11.1).

Two options exist for extracting the optimal basis modes from a given complex system. One can either collect data directly from an experiment, or one can simulate the complex system and sample the state of the system as it evolves according to the dynamics. In both cases, snapshots of the dynamics are taken and optimal modes identified. In the case when the system is simulated to extract modes, one can argue that no computational savings are achieved. However, much like the LU decomposition, which has an initial one-time computational cost of $O(n^3)$ before further $O(n^2)$ operations can be applied, the costly modal extraction process is performed only once. The optimal modes can then be used in a computationally efficient manner thereafter.

To proceed with the construction of the optimal POD modes, the dynamics of (11.1) are sampled at some prescribed time interval. In particular, a snapshot $\mathbf{u}_k$ consists of samples of the complex system, with subscript k indicating sampling at time t_k: $\mathbf{u}_k := \begin{bmatrix}\mathbf{u}(x_1,t_k) & \mathbf{u}(x_2,t_k) & \cdots & \mathbf{u}(x_n,t_k)\end{bmatrix}^T$. Now, the continuous functions and modes will be evaluated at n discrete spatial locations, resulting in a high-dimensional vector representation; these will be denoted by bold symbols. We are generally interested in analyzing the computationally or experimentally generated large data set $\mathbf{X}$:

$$\mathbf{X} = \begin{bmatrix} | & | & & | \\ \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_m \\ | & | & & | \end{bmatrix} \tag{11.16}$$

where the columns $\mathbf{u}_k = \mathbf{u}(t_k) \in \mathbb{C}^n$ may be measurements from simulations or experiments. $\mathbf{X}$ consists of a *time-series* of data, with m distinct measurement instances in time. Often the *state-dimension* n is very large, on the order of millions or billions in the case of fluid systems. Typically $n \gg m$, resulting in a *tall-skinny* matrix, as opposed to a *short-fat* matrix when $n \ll m$.

As discussed previously, the singular value decomposition (SVD) provides a unique matrix decomposition for any complex valued matrix $\mathbf{X} \in \mathbb{C}^{n\times m}$:

$$\mathbf{X} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^* \tag{11.17}$$

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

where $\mathbf{U} \in \mathbb{C}^{n\times n}$ and $\mathbf{V} \in \mathbb{C}^{m\times m}$ are *unitary* matrices and $\mathbf{\Sigma} \in \mathbb{C}^{n\times m}$ is a matrix with nonnegative entries on the diagonal. Here * denotes the complex conjugate transpose. The columns of $\mathbf{U}$ are called *left singular vectors* of $\mathbf{X}$ and the columns of $\mathbf{V}$ are *right singular vectors*. The diagonal elements of $\mathbf{\Sigma}$ are called *singular values* and they are ordered from largest to smallest. The SVD provides critical insight into building an optimal basis set tailored to the specific problem. In particular, the matrix $\mathbf{U}$ is guaranteed to provide the best set of modes to approximate $\mathbf{X}$ in an ℓ_2 sense. Specifically, the columns of this matrix contain the orthogonal modes necessary to form the ideal basis. The matrix $\mathbf{V}$ gives the time-history of each of the modal elements and the diagonal matrix $\mathbf{\Sigma}$ is the weighting of each mode relative to the others. Recall that the modes are arranged with the most dominant first and the least dominant last.

The total number of modes generated is typically determined by the number of snapshots m taken in constructing $\mathbf{X}$ (where normally $n \gg m$). Our objective is to determine the minimal number of modes necessary to accurately represent the dynamics of (11.1) with a Galerkin projection (11.6). Thus we are interested in a rank-r approximation to the true dynamics where typically $r \ll m$. The quantity of interest is then the low-rank decomposition of the SVD given by

$$\tilde{\mathbf{X}} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^* \tag{11.18}$$

where $\|\mathbf{X} - \tilde{\mathbf{X}}\| < \epsilon$ for a given small value of epsilon. This low-rank truncation allows us to construct the modes of interest $\boldsymbol{\psi}_k$ from the columns of the truncated matrix $\tilde{\mathbf{U}}$. In particular, the optimal basis modes are given by

$$\tilde{\mathbf{U}} = \mathbf{\Psi} = \begin{bmatrix} | & | & & | \\ \boldsymbol{\psi}_1 & \boldsymbol{\psi}_2 & \cdots & \boldsymbol{\psi}_r \\ | & | & & | \end{bmatrix} \tag{11.19}$$

where the truncation preserves the r most dominant modes used in (11.6). The truncated r modes $\{\boldsymbol{\psi}_1, \boldsymbol{\psi}_2, \cdots, \boldsymbol{\psi}_r\}$ are then used as the low-rank, orthogonal basis to represent the dynamics of (11.1).

The above snapshot based method for extracting the low-rank, r-dimensional subspace of dynamic evolution associated with (11.1) is a data-driven computational architecture. Indeed, it provides an equation-free method, i.e. the governing equation (11.1) may actually be unknown. In the event that the underlying dynamics are unknown, then the extraction of the low-rank space allows one to build potential models in an r-dimensional subspace as opposed to remaining in a high-dimensional space where $n \gg r$. These ideas will be explored further in what follows. However, it suffices to highlight at this juncture that an optimal basis representation does not require an underlying knowledge of the complex system (11.1).

Galerkin Projection onto POD Modes

It is possible to approximate the state $\mathbf{u}$ of the PDE using a Galerkin expansion:

$$\mathbf{u}(t) \approx \mathbf{\Psi}\mathbf{a}(t) \tag{11.20}$$

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

where $\mathbf{a}(t) \in \mathbb{R}^r$ is the time-dependent coefficient vector and $r \ll n$. Plugging this modal expansion into the governing equation (11.13) and applying orthogonality (multiplying by $\mathbf{\Psi}^T$) gives the dimensionally reduced evolution

$$\frac{d\mathbf{a}(t)}{dt} = \mathbf{\Psi}^T \mathbf{L} \mathbf{\Psi} \mathbf{a}(t) + \mathbf{\Psi}^T \mathbf{N}(\mathbf{\Psi} \mathbf{a}(t), \boldsymbol{\beta}). \tag{11.21}$$

By solving this system of much smaller dimension, the solution of a high-dimensional nonlinear dynamical system can be approximated. Of critical importance is evaluating the nonlinear terms in an efficient way using the gappy POD or DEIM mathematical architecture in Chapter 12. Otherwise, the evaluation of the nonlinear terms still requires calculation of functions and inner products with the original dimension n. In certain cases, such as the quadratic nonlinearity of Navier-Stokes, the nonlinear terms can be computed once in an off-line manner. However, parametrized systems generally require repeated evaluation of the nonlinear terms as the POD modes change with $\boldsymbol{\beta}$.

Example: The Harmonic Oscillator

To illustrate the POD method for selecting optimal basis elements, we will consider a classic problem of mathematical physics: the *quantum harmonic oscillator*. Although the ideal basis functions (Gauss-Hermite functions) for this problem are already known, we would like to infer these special functions in a purely data-driven way. In other words, can we deduce these special functions from snapshots of the dynamics alone? The standard harmonic oscillator arises in the study of spring-mass systems. In particular, one often assumes that the restoring force F of a spring is governed by the linear Hooke's law:

$$F(t) = -kx \tag{11.22}$$

where k is the spring constant and $x(t)$ represents the displacement of the spring from its equilibrium position. Such a force gives rise to a potential energy for the spring of the form $V = kx^2/2$.

In considering quantum mechanical systems, such a restoring force (with $k = 1$ without loss of generality) and associated potential energy gives rise to the Schrödinger equation with a parabolic potential

$$iu_t + \frac{1}{2}u_{xx} - \frac{x^2}{2}u = 0 \tag{11.23}$$

where the second term in the partial differential equation represents the kinetic energy of a quantum particle while the last term is the parabolic potential associated with the linear restoring force.

The solution for the quantum harmonic oscillator can be easily computed in terms of special functions. In particular, by assuming a solution of the form

$$u(x,t) = a_k \psi_k(x) \exp\left[-i(k + 1/2)t\right] \tag{11.24}$$

with a_k determined from initial conditions, one finds the following boundary value problem for the eigenmodes of the system

$$\frac{d^2\psi_k}{dx^2} + (2k + 1 - x^2)\psi_k \tag{11.25}$$

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

with the boundary conditions $\psi_k \to 0$ as $x \to \pm\infty$. Normalized solutions to this equation can be expressed in terms of *Hermite polynomials*, $H_k(x)$ or the Gaussian-Hermite functions

$$\psi_k = \left(2^k k\sqrt{\pi}\right)^{-1/2} \exp(-x^2/2) H_k(x) \tag{11.26a}$$

$$= (-1)^k \left(2^k k\sqrt{\pi}\right)^{-1/2} \exp(-x^2/2) \frac{d^k}{dx^k} \exp(-x^2) . \tag{11.26b}$$

The Gauss-Hermite functions are typically thought of as the optimal basis functions for the harmonic oscillator as they naturally represent the underlying dynamics driven by the Schrödinger equation with parabolic potential. Indeed, solutions of the complex system (11.23) can be represented as the sum

$$u(x,t) = \sum_{k=0}^{\infty} a_k \left(2^k k\sqrt{\pi}\right)^{-1/2} \exp(-x^2/2) H_k(x) \exp\left[-i(k+1/2)t\right] . \tag{11.27}$$

Such a solution strategy is ubiquitous in mathematical physics as is evidenced by the large number of special functions, often of Sturm-Liouville form, for different geometries and boundary conditions. These include Bessel functions, Laguerre polynomials, Legendre polynomials, parabolic cylinder functions, spherical harmonics, etc.

A numerical solution to the governing PDE (11.23) based on the fast Fourier transform is easy to implement [316]. The following code executes a full numerical solution with the initial conditions $u(x,0) = \exp(-0.2(x-x_0)^2)$, which is a Gaussian pulse centered at $x = x_0$. This initial condition generically excites a number of Gauss-Hermite functions. In particular, the initial projection onto the eigenmodes is computed from the orthogonality conditions so that

$$a_k = \langle u(x,0), \psi_k \rangle . \tag{11.28}$$

This inner product projects the initial condition onto each mode ψ_k.

Code 11.1 Harmonic oscillator code.

```
L=30; n=512; x2=linspace(-L/2,L/2,n+1); x=x2(1:n); % spatial
    discretization
k=(2*pi/L)*[0:n/2-1 -n/2:-1].';     % wavenumbers for FFT
V=x.^2.';                 % potential
t=0:0.2:20;               % time domain collection points

u=exp(-0.2*(x-1).^2); % initial conditions
ut=fft(u);                % FFT initial data
[t,utsol]=ode45('pod_harm_rhs',t,ut,[],k,V); % integrate PDE
for j=1:length(t)
  usol(j,:)=ifft(utsol(j,:));       % transforming back
end
```

The right-hand side function, **pod_harm_rhs.m** associated with the above code contains the governing equation (11.23) in a three-line MATLAB code:

Code 11.2 Harmonic oscillator right-hand side.

```
function rhs=pod_harm_rhs(t,ut,dummy,k,V)
u=ifft(ut);
rhs=-(i/2)*(k.^2).*ut - 0.5*i*fft(V.*u);
```

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

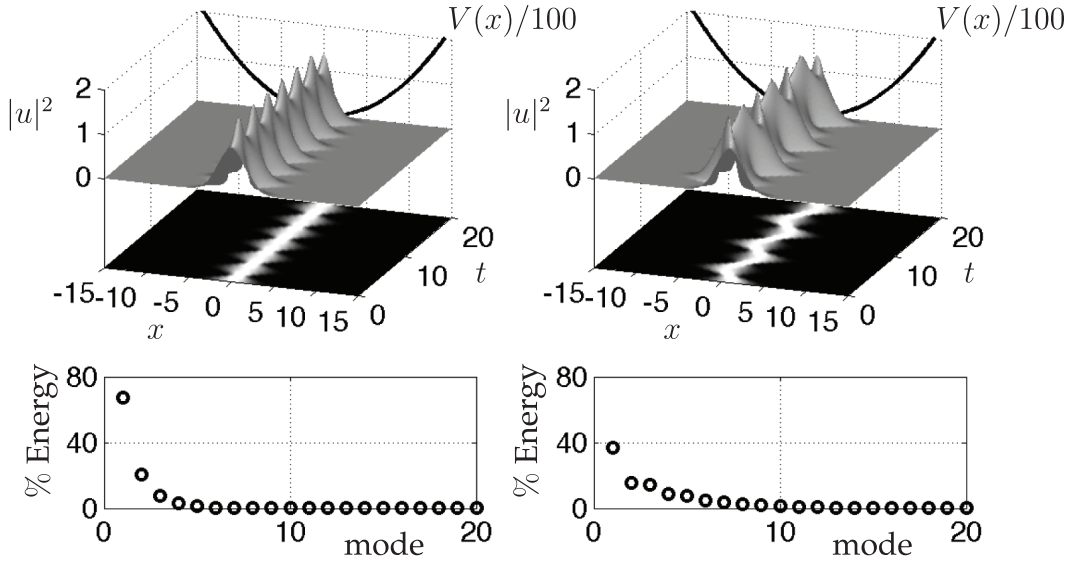

Figure 11.2 Dynamics of the quantum harmonic oscillator (11.23) given the initial condition $u(x,0) = \exp(-0.2(x-x_0)^2)$ for $x_0 = 0$ (left panel) and $x_0 = 1$ (right panel). The symmetric initial data elicits a dominant five mode response while the initial condition with initial offset $x_0 = 1$ activates ten modes. The bottom panels show the singular values of the SVD of their corresponding top panels along with the percentage of energy (or L^2 norm) in each mode. The dynamics are clearly low-rank given the rapid decay of the singular values.

The two codes together produce dynamics associated with the quantum harmonic oscillator. Fig. 11.2 shows the dynamical evolution of an initial Gaussian $u(x,0) = \exp(-0.2(x-x_0)^2)$ with $x_0 = 0$ (left panel) and $x_0 = 1$ (right panel). From the simulation, one can see that there are a total of 101 snapshots (the initial condition and an additional 100 measurement times). These snapshots can be organized as in (11.16) and the singular value decomposition performed. The singular values of the decomposition are suggestive of the underlying dimensionality of the dynamics. For the dynamical evolution observed in the top panels of Fig. 11.2, the corresponding singular values of the snapshots are given in the bottom panels. For the symmetric initial condition (symmetric about $x = 0$), five modes dominate the dynamics. In contrast, for an asymmetric initial condition, twice as many modes are required to represent the dynamics with the same precision.

The singular value decomposition not only gives the distribution of energy within the first set of modes, but it also produces the optimal basis elements as columns of the matrix $\mathbf{U}$. The distribution of singular values is highly suggestive of how to truncate with a low-rank subspace of r modes, thus allowing us to construct the dimensionally reduced space (11.19) appropriate for a Galerkin-POD expansion.

The modes of the quantum harmonic oscillator are illustrated in Fig. 11.3. Specifically, the first five modes are shown for (i) the Gauss-Hermite functions representing the special function solutions, (ii) the modes of the SVD for the symmetric ($x_0 = 0$) initial conditions, and (iii) the modes of the SVD for the offset (asymmetric, $x_0 = 1$) initial conditions. The Gauss-Hermite functions, by construction, are arranged from lowest eigenvalue

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

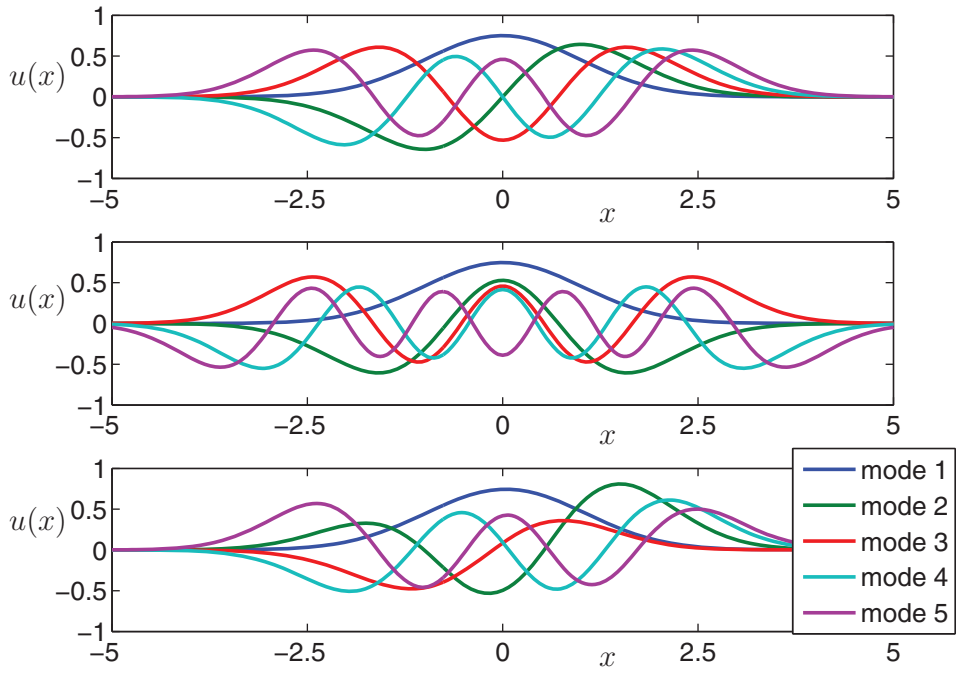

Figure 11.3 First five modes of the quantum harmonic oscillator. In the top panel, the first five Gauss-Hermite modes (11.26), arranged by their Sturm-Liouville eigenvalue, are illustrated. The second panel shows the dominant modes computed from the SVD of the dynamics of the harmonic oscillator with $u(x, 0) = \exp(-0.2x^2)$, illustrated in Fig. 11.2 left panel. Note that the modes are all symmetric since no asymmetric dynamics was actually manifested. For the bottom panel, where the harmonic oscillator was simulated with the offset Gaussian $u(x, 0) = \exp(-0.2(x - 1)^2)$, asymmetry is certainly observed. This also produces modes that are very similar to the Gauss-Hermite functions. Thus a purely snapshot based method is capable of reproducing the nearly ideal basis set for the harmonic oscillator.

of the Sturm-Liouville problem (11.25). The eigenmodes alternate between symmetric and asymmetric modes. For the symmetric (about $x = 0$) initial conditions given by $u(x, 0) = \exp(-0.2x^2)$, the first five modes are all symmetric as the snapshot based method is incapable of producing asymmetric modes since they are actually not part of the dynamics, and thus they are not observable, or manifested in the evolution. In contrast, with a slight offset, $u(x, 0) = \exp(-0.2(x - 1)^2)$, snapshots of the evolution produce asymmetric modes that closely resemble the asymmetric modes of the Gauss-Hermite expansion. Interestingly, in this case, the SVD arranges the modes by the amount of energy exhibited in each mode. Thus the first asymmetric mode (bottom panel in red – third mode) is equivalent to the second mode of the exact Gauss-Hermite polynomials (top panel in green – second mode). The key observation here is that the snapshot based method is capable of generating, or nearly so, the known optimal Gauss-Hermite polynomials characteristic of this system. Importantly, the POD-Galerkin method generalizes to more complex physics and geometries where the solution is not known *a priori*.

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

11.3 POD and Soliton Dynamics

To illustrate a full implementation of the Galerkin-POD method, we will consider an illustrative complex system whose dynamics are strongly nonlinear. Thus, we consider the nonlinear Schrödinger (NLS) equation

$$iu_t + \frac{1}{2}u_{xx} + |u|^2 u = 0 \tag{11.29}$$

with the boundary conditions $u \to 0$ as $x \to \pm\infty$. If not for the nonlinear term, this equation could be solved easily in closed form. However, the nonlinearity mixes the eigenfunction components in the expansion (11.6), and it is impossible to derive a simple analytic solution.

To solve the NLS computationally, a Fourier mode expansion is used. Thus the standard fast Fourier transform may be leveraged. Rewriting (11.29) in the Fourier domain, i.e. taking the Fourier transform, gives the set of differential equations

$$\hat{u}_t = -\frac{i}{2}k^2\hat{u} + i\widehat{|u|^2 u} \tag{11.30}$$

where the Fourier mode mixing occurs due to the nonlinear mixing in the cubic term. This gives the system of differential equations to be solved in order to evaluate the NLS behavior.

The following code formulates the PDE solution as an eigenfunction expansion (11.6) of the NLS (11.29). The first step in the process is to define an appropriate spatial and temporal domain for the solution along with the Fourier frequencies present in the system. The following code produces both the time and space domains of interest:

Code 11.3 Nonlinear Schrödinger equation solver.

```
L=40; n=512; x2=linspace(-L/2,L/2,n+1); x=x2(1:n); % spatial
    discretization
k=(2*pi/L)*[0:n/2-1 -n/2:-1].';  % wavenumbers for FFT
t=linspace(0,2*pi,21);           % time domain collection points

N=1;
u=N*sech(x);              % initial conditions
ut=fft(u);                % FFT initial data
[t,utsol]=ode45('pod_sol_rhs',t,ut,[],k); % integrate PDE
for j=1:length(t)
  usol(j,:)=ifft(utsol(j,:));          % transforming back
end
```

The right-hand side function, **pod_sol_rhs.m** associated with the above code contains the governing equation (11.29) in a three-line MATLAB code:

Code 11.4 NLS right-hand side.

```
function rhs=pod_sol_rhs(t,ut,dummy,k)
u=ifft(ut);
rhs=-(i/2)*(k.^2).*ut + i*fft( (abs(u).^2).*u );
```

It now remains to consider a specific spatial configuration for the initial condition. For the NLS, there are a set of special initial conditions called solitons where the initial conditions are given by

$$u(x,0) = N\text{sech}(x) \tag{11.31}$$

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

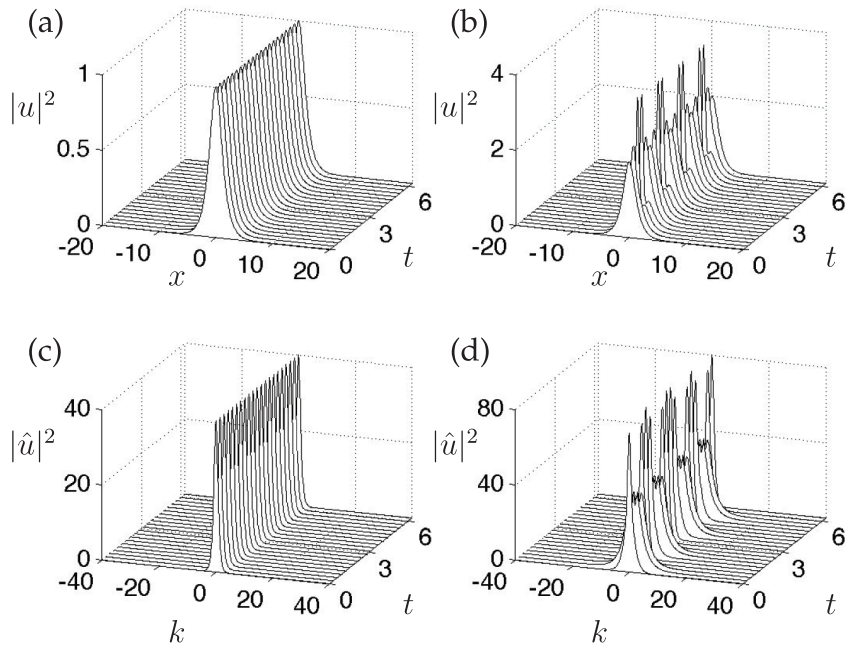

Figure 11.4 Evolution of the (a) $N = 1$ and (b) $N = 2$ solitons. Here steady-state ($N = 1$, left panels (a) and (c)) and periodic ($N = 2$, right panels (b) and (d)) dynamics are observed and approximately 50 and 200 Fourier modes, respectively, are required to model the behaviors.

where N is an integer. We will consider the soliton dynamics with $N = 1$ and $N = 2$. First, the initial condition is projected onto the Fourier modes with the fast Fourier transform.

The dynamics of the $N = 1$ and $N = 2$ solitons are demonstrated in Fig. 11.4. During evolution, the $N = 1$ soliton only undergoes phase changes while its amplitude remains stationary. In contrast, the $N = 2$ soliton undergoes periodic oscillations. In both cases, a large number of Fourier modes, about 50 and 200 respectively, are required to model the simple behaviors illustrated.

The obvious question to ask in light of our dimensionality reduction thinking is this: is the soliton dynamics really a 50 or 200 degrees-of-freedom system as required by the Fourier mode solution technique. The answer is no. Indeed, with the appropriate basis, i.e. the POD modes generated from the SVD, it can be shown that the dynamics is a simple reduction to 1 or 2 modes respectively. Indeed, it can easily be shown that the $N = 1$ and $N = 2$ solitons are truly low dimensional by computing the singular value decomposition of the evolutions shown in Fig. 11.4.

Fig. 11.5 explicitly demonstrates the low-dimensional nature of the numerical solutions by computing the singular values, along with the modes to be used in our new eigenfunction expansion. For both of these cases, the dynamics are truly low dimensional with the $N = 1$ soliton being modeled well by a single POD mode while the $N = 2$ dynamics are modeled quite well with two POD modes. Thus, in performing an eigenfunction expansion, the modes chosen should be the POD modes generated from the simulations themselves. In the next section, we will derive the dynamics of the modal interaction for these two cases, which are low-dimensional and amenable to analysis.

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

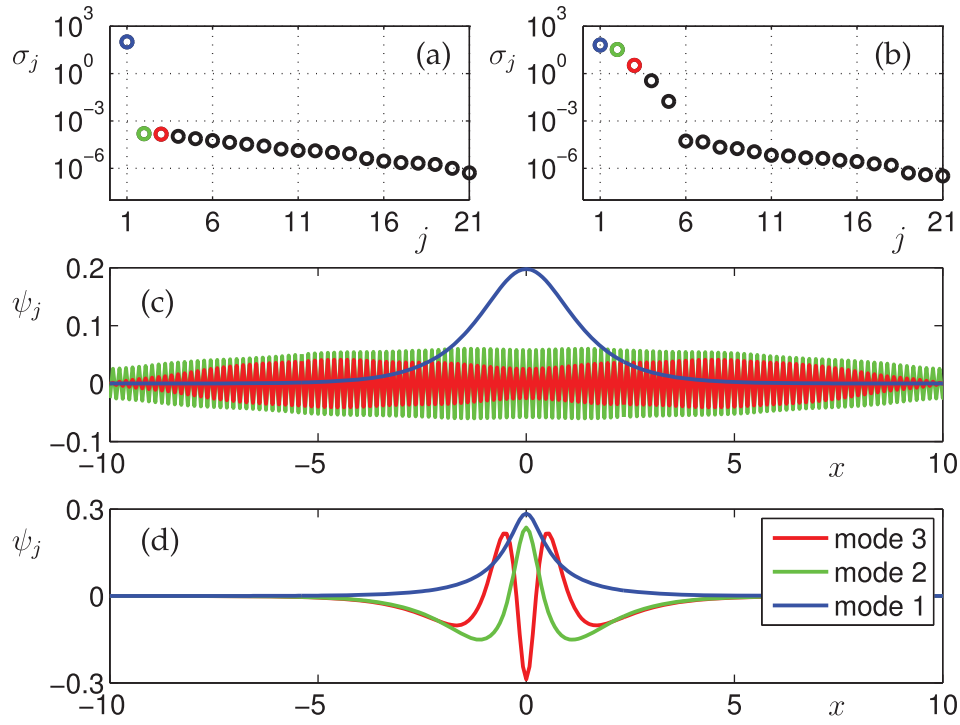

Figure 11.5 Projection of the $N = 1$ and $N = 2$ evolutions onto their POD modes. The top two figures (a) and (b) are the singular values σ_j on a logarithmic scale of the two evolutions demonstrated in (11.4). This demonstrates that the $N = 1$ and $N = 2$ soliton dynamics are primarily low-rank, with the $N = 1$ being a single mode evolution and the $N = 2$ being dominated by two modes that contain approximately 95% of the evolution variance. The first three modes in both cases are shown in the bottom two panels (c) and (d).

Soliton Reduction ($N = 1$)

To take advantage of the low dimensional structure, we first consider the $N = 1$ soliton dynamics. Fig. 11.5 shows that a single mode in the SVD dominates the dynamics. This is the first column of the $\mathbf{U}$ matrix. Thus the dynamics are recast in a single mode so that

$$u(x,t) = a(t)\psi(x). \tag{11.32}$$

Plugging this into the NLS equation (11.29) yields the following:

$$ia_t\psi + \frac{1}{2}a\psi_{xx} + |a|^2 a|\psi|^2\psi = 0\,. \tag{11.33}$$

The inner product is now taken with respect to ψ which gives

$$ia_t + \frac{\alpha}{2}a + \beta|a|^2 a = 0 \tag{11.34}$$

where

$$\alpha = \frac{\langle \psi_{xx}, \psi \rangle}{\langle \psi, \psi \rangle} \tag{11.35a}$$

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

$$\beta = \frac{\langle |\psi|^2\psi, \psi\rangle}{\langle \psi, \psi\rangle}. \tag{11.35b}$$

This is the low-rank approximation achieved by the POD-Galerkin method.

The differential equation (11.34) for $a(t)$ can be solved explicitly to yield

$$a(t) = a(0)\exp\left(i\frac{\alpha}{2}t + \beta|a(0)|^2 t\right) \tag{11.36}$$

where $a(0)$ is the initial condition for $a(t)$. To find the initial condition, recall that

$$u(x,0) = \text{sech}(x) = a(0)\psi(x)\,. \tag{11.37}$$

Taking the inner product with respect to $\psi(x)$ gives

$$a(0) = \frac{\langle \text{sech}(x), \psi\rangle}{\langle \psi, \psi\rangle}\,. \tag{11.38}$$

Thus the one mode expansion gives the approximate PDE solution

$$u(x,t) = a(0)\exp\left(i\frac{\alpha}{2}t + \beta|a(0)|^2 t\right)\psi(x)\,. \tag{11.39}$$

This solution is the low-dimensional POD approximation of the PDE expanded in the best basis possible, i.e. the SVD basis.

For the $N = 1$ soliton, the spatial profile remains constant while its phase undergoes a nonlinear rotation. The POD solution (11.39) can be solved exactly to characterize this phase rotation.

Soliton Reduction ($N = 2$)

The $N = 2$ soliton case is a bit more complicated and interesting. In this case, two modes clearly dominate the behavior of the system, as they contain 96% of the energy. These two modes, ψ_1 and ψ_2, are the first two columns of the matrix $\mathbf{U}$ and are now used to approximate the dynamics observed in Fig. (11.4). In this case, the two mode expansion takes the form

$$u(x,t) = a_1(t)\psi_1(x) + a_2(t)\psi_2(x). \tag{11.40}$$

Inserting this approximation into the governing equation (11.29) gives

$$i\,(a_{1t}\psi_1 + a_{2t}\psi_2) + \frac{1}{2}\left(a_1\psi_{1xx} + a_2\psi_{2xx}\right) + (a_1\psi_1 + a_2\psi_2)^2(a_1^*\psi_1^* + a_2^*\psi_2^*) = 0. \tag{11.41}$$

Multiplying out the cubic term gives

$$\begin{aligned} &i\,(a_{1t}\psi_1 + a_{2t}\psi_2) + \frac{1}{2}\left(a_1\psi_{1xx} + a_2\psi_{2xx}\right) \\ &\quad + \Big(|a_1|^2a_1|\psi_1|^2\psi_1 + |a_2|^2a_2|\psi_2|^2\psi_2 + 2|a_1|^2a_2|\psi_1|^2\psi_2 + 2|a_2|^2a_1|\psi_2|^2\psi_1 \\ &\quad + a_1^2a_2^*\psi_1^2\psi_2^* + a_2^2a_1^*\psi_2^2\psi_1^*\Big)\,. \end{aligned} \tag{11.42}$$

All that remains is to take the inner product of this equation with respect to both $\psi_1(x)$ and $\psi_2(x)$. Recall that these two modes are orthogonal, resulting in the following 2×2 system of nonlinear equations:

$$ia_{1t} + \alpha_{11}a_1 + \alpha_{12}a_2 + \left(\beta_{111}|a_1|^2 + 2\beta_{211}|a_2|^2\right)a_1 \tag{11.43a}$$

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

$$+\left(\beta_{121}|a_1|^2+2\beta_{221}|a_2|^2\right)a_2+\sigma_{121}a_1^2a_2^*+\sigma_{211}a_2^2a_1^*=0$$
$$ia_{2t}+\alpha_{21}a_1+\alpha_{22}a_2+\left(\beta_{112}|a_1|^2+2\beta_{212}|a_2|^2\right)a_1 \tag{11.43b}$$
$$+\left(\beta_{122}|a_1|^2+2\beta_{222}|a_2|^2\right)a_2+\sigma_{122}a_1^2a_2^*+\sigma_{212}a_2^2a_1^*=0$$

where

$$\alpha_{jk}=\langle\psi_{j_{xx}},\psi_k\rangle/2 \tag{11.44a}$$
$$\beta_{jkl}=\langle|\psi_j|^2\psi_k,\psi_l\rangle \tag{11.44b}$$
$$\sigma_{jkl}=\langle\psi_j^2\psi_k^*,\psi_l\rangle \tag{11.44c}$$

and the initial values of the two components are given by

$$a_1(0)=\frac{\langle 2\text{sech}(x),\psi_1\rangle}{\langle\psi_1,\psi_1\rangle} \tag{11.45a}$$
$$a_2(0)=\frac{\langle 2\text{sech}(x),\psi_2\rangle}{\langle\psi_2,\psi_2\rangle}\,. \tag{11.45b}$$

This gives a complete description of the two mode dynamics predicted from the SVD analysis.

The two mode dynamics accurately approximates the solution. However, there is a phase drift that occurs in the dynamics that would require higher precision in both the time series of the full PDE and more accurate integration of the inner products for the coefficients. Indeed, the most simple trapezoidal rule has been used to compute the inner products and its accuracy is somewhat suspect; this issue will be addressed in the following section. Higher-order schemes could certainly help improve the accuracy. Additionally, incorporating the third or higher modes could also help. In either case, this demonstrates how one would use the low dimensional structures to approximate PDE dynamics in practice.

11.4 Continuous Formulation of POD

Thus far, the POD reduction has been constructed to accommodate discrete data measurement snapshots $\mathbf{X}$ as given by (11.16). The POD reduction generates a set of low-rank basis modes $\mathbf{\Psi}$ so that the following least-squares error is minimized:

$$\underset{\mathbf{\Psi}\text{ s.t. rank}(\mathbf{\Psi})=r}{\text{argmin}} \|\mathbf{X}-\mathbf{\Psi}\mathbf{\Psi}^T\mathbf{X}\|_F\,. \tag{11.46}$$

Recall that $\mathbf{X}\in\mathbb{C}^{n\times m}$ and $\mathbf{\Psi}\in\mathbb{C}^{n\times r}$ where r is the rank of the truncation.

In many cases, measurements are performed on a continuous time process over a prescribed spatial domain, thus the data we consider are constructed from trajectories

$$u(x,t)\quad t\in[0,T],\ x\in[-L,L]. \tag{11.47}$$

Such data require a *continuous* time formulation of the POD reduction. In particular, an equivalent of (11.46) must be constructed for these continuous time trajectories. Note that instead of a spatially dependent function $u(x,t)$, one can also consider a vector of trajectories $\mathbf{u}(t)\in\mathbb{C}^n$. This may arise when a PDE is discretized so that the infinite dimensional spatial variable x is finite dimensional. Wolkwein [542, 543] gives an excellent, technical overview of the POD method and its continuous formulation.

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

To define the continuous formulation, we prescribe the inner product

$$\langle f(x), g(x)\rangle = \int_{-L}^{L} f(x)g^*(x)dx\,. \tag{11.48}$$

To find the best fit function through the entire temporal trajectory $u(x,t)$ in (11.47), the following minimization problem must be solved

$$\min_{\psi} \frac{1}{T}\int_0^T \|u(x,t) - \langle u(x,t), \psi(x)\rangle\,\psi\|^2\,dt \quad \text{subject to} \quad \|\psi\|^2 = 1 \tag{11.49}$$

where the normalization of the temporal integral by $1/T$ averages the difference between the data and its low-rank approximation using the function ψ over the time $t \in [0,T]$. Equation (11.49) is equivalent to maximizing the inner product between the data $u(x,t)$ and the function $\psi(x)$, i.e. they are maximally parallel in function space. Thus the minimization problem can be restated as

$$\max_{\psi} \frac{1}{T}\int_0^T |\langle u(x,t), \psi(x)\rangle|^2 dt \quad \text{subject to} \quad \|\psi\|^2 = 1\,. \tag{11.50}$$

The constrained optimization problem in (11.50) can be reformulated as a Lagrangian functional

$$\mathcal{L}(\psi,\lambda) = \frac{1}{T}\int_0^T |\langle u(x,t), \psi(x)\rangle|^2 dt + \lambda\left(1 - \|\psi\|^2\right) \tag{11.51}$$

where λ is the Lagrange multiplier that enforces the constraint $\|\psi\|^2 = 1$. This can be rewritten as

$$\begin{aligned}\mathcal{L}(\psi,\lambda) = \frac{1}{T}\int_0^T \left(\int_{-L}^{L} u(\xi,t)\psi^*(\xi)d\xi \int_{-L}^{L} u^*(x,t)\psi(x)dx\right)dt \\ +\lambda\left(1 - \|\psi\|^2\right) + \lambda\left(1 - \int_{-L}^{L}\psi(x)\psi^*(x)dx\right).\end{aligned} \tag{11.52}$$

The Lagrange multiplier problem requires that the functional derivative be zero:

$$\frac{\partial \mathcal{L}}{\partial \psi^*} = 0. \tag{11.53}$$

Applying this derivative constraint to (11.52) and interchanging integrals yields

$$\frac{\partial \mathcal{L}}{\partial \psi^*} = \int_{-L}^{L} d\xi\left[\frac{1}{T}\int_0^T \left(u(\xi,t)\int_{-L}^{L} u^*(x,t)\psi(x)dx\right)dt - \lambda\psi(x)\right] = 0\,. \tag{11.54}$$

Setting the integrand to zero, the following eigenvalue problem is derived

$$\langle R(\xi,x), \psi\rangle = \lambda\psi \tag{11.55}$$

where $R(\xi,x)$ is a two-point correlation tensor of the continuous data $u(x,t)$ which is averaged over the time interval where the data is sampled

$$R(\xi,x) = \frac{1}{T}\int_0^T u(\xi,t)u^*(x,t)dt\,. \tag{11.56}$$

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

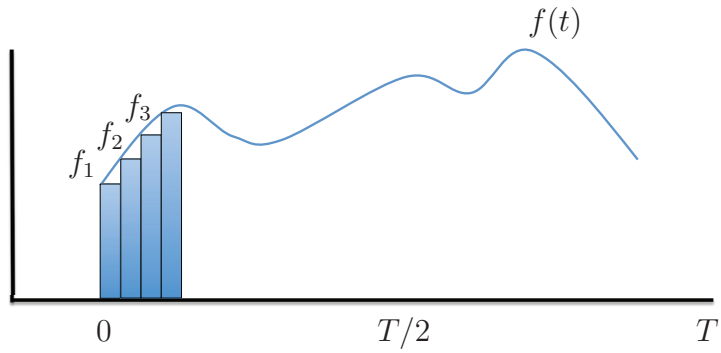

Figure 11.6 Illustration of an implementation of the quadrature rule to evaluate the integrals $\int_0^T f(t)dt$. The rectangles of height $f(t_j) = f_j$ and width δt are summed to approximate the integral.

If the spatial direction x is discretized, resulting in a high-dimensional vector $\mathbf{u}(t) = \begin{bmatrix} u(x_1,t) & u(x_2,t) & \cdots & u(x_n,t) \end{bmatrix}^T$, then $R(\xi, x)$ becomes:

$$\mathbf{R} = \frac{1}{T}\int_0^T \mathbf{u}(t)\mathbf{u}^*(t)dt\,. \tag{11.57}$$

In practice, the function $\mathbf{R}$ is evaluated using a quadrature rule for integration. This will allow us to connect the method to the snapshot based method discussed thus far.

Quadrature Rules for R: Trapezoidal Rule

The evaluation of the integral (11.57) can be performed by numerical quadrature [316]. The simplest quadrature rule is the trapezoidal rule which evaluates the integral via summation of approximating rectangles. Fig. 11.6 illustrates a version of the trapezoidal rule where the integral is approximated by a summation over a number of rectangles. This gives the approximation of the two-point correlation tensor:

$$\begin{aligned} \mathbf{R} &= \frac{1}{T}\int_0^T \mathbf{u}(t)\mathbf{u}^*(t)dt \\ &\approx \frac{\Delta t}{T}\left[\mathbf{u}^*(t_1)\mathbf{u}(t_1) + \mathbf{u}^*(t_2)\mathbf{u}(t_2) + \cdots + \mathbf{u}^*(t_m)\mathbf{u}(t_m)\right] \\ &= \frac{\Delta t}{T}\left[\mathbf{u}_1^*\mathbf{u}_1 + \mathbf{u}_2^*\mathbf{u}_2 + \cdots + \mathbf{u}_m^*\mathbf{u}_m\right] \end{aligned} \tag{11.58}$$

where we have assumed $u(x,t)$ is discretized into a vector $\mathbf{u}_j = \mathbf{u}(t_j)$, and there are m rectangular bins of width Δt so that $(m)\Delta t = T$. Defining a data matrix

$$\mathbf{X} = [\mathbf{u}_1 \;\; \mathbf{u}_2 \;\; \cdots \;\; \mathbf{u}_m] \tag{11.59}$$

we can then rewrite the two-point correlation tensor as

$$\mathbf{R} \approx \frac{1}{m}\mathbf{X}^*\mathbf{X} \tag{11.60}$$

which is exactly the definition of the covariance matrix in (1.27), i.e. $\mathbf{C} \approx \mathbf{R}$. Note that the role of $1/T$ is to average over the various trajectories so that the average is subtracted out, giving rise to a definition consistent with the covariance.

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

Higher-order Quadrature Rules

Numerical integration simply calculates the area under a given curve. The basic ideas for performing such an operation come from the definition of integration

$$\int_a^b f(t)dt = \lim_{\Delta t \to 0} \sum_{j=0}^{m-1} f(t_j)\Delta t \tag{11.61}$$

where $b - a = (m-1)\Delta t$. The area under the curve is a limiting process of summing up an ever-increasing number of rectangles. This process is known as numerical quadrature. Specifically, any sum can be represented as follows:

$$Q[f] = \sum_{j=0}^{m-1} w_j f(t_j) = w_0 f(t_0) + w_1 f(t_1) + \cdots + w_{m-1} f(t_{m-1}) \tag{11.62}$$

where $a = t_0 < t_1 < t_2 < \cdots < t_{m-1} = b$. Thus the integral is evaluated as

$$\int_a^b f(t)dt = Q[f] + E[f] \tag{11.63}$$

where the term $E[f]$ is the error in approximating the integral by the quadrature sum (11.62). Typically, the error $E[f]$ is due to truncation error. To integrate, we will use polynomial fits to the y-values $f(t_j)$. Thus we assume the function $f(t)$ can be approximated by a polynomial

$$P_n(t) = a_n t^n + a_{n-1} t^{n-1} + \cdots + a_1 t + a_0 \tag{11.64}$$

where the truncation error in this case is proportional to the $(n+1)^{th}$ derivative $E[f] = Af^{(n+1)}(c)$ and A is a constant. This process of polynomial fitting the data gives the *Newton-Cotes Formulas*.

The following integration approximations result from using a polynomial fit through the data to be integrated. It is assumed that

$$t_k = t_0 + \Delta t k \qquad\qquad f_k = f(t_k)\,. \tag{11.65}$$

This gives the following integration algorithms:

$$\text{Trapezoid Rule} \int_{t_0}^{t_1} f(t)dt = \frac{\Delta t}{2}(f_0 + f_1) - \frac{\Delta t^3}{12} f''(c) \tag{11.66a}$$

$$\text{Simpson's Rule} \int_{t_0}^{t_2} f(t)dt = \frac{\Delta t}{3}(f_0 + 4f_1 + f_2) - \frac{\Delta t^5}{90} f''''(c) \tag{11.66b}$$

$$\text{Simpson's 3/8 Rule} \int_{t_0}^{t_3} f(t)dt = \frac{3\Delta t}{8}(f_0 + 3f_1 + 3f_2 + f_3) - \frac{3\Delta t^5}{80} f''''(c) \tag{11.66c}$$

$$\text{Boole's Rule} \int_{t_0}^{t_4} f(t)dt = \frac{2\Delta t}{45}(7f_0 + 32f_1 + 12f_2 + 32f_3 + 7f_4) - \frac{8\Delta t^7}{945} f^{(6)}(c). \tag{11.66d}$$

These algorithms have varying degrees of accuracy. Specifically, they are $O(\Delta t^2)$, $O(\Delta t^4)$, $O(\Delta t^4)$ and $O(\Delta t^6)$ accurate schemes respectively. The accuracy condition is determined from the truncation terms of the polynomial fit. Note that the *trapezoidal rule* uses a sum of simple trapezoids to approximate the integral. *Simpson's rule* fits a quadratic curve through three points and calculates the area under the quadratic curve. *Simpson's 3/8 rule* uses four

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

points and a cubic polynomial to evaluate the area, while *Boole's rule* uses five points and a quartic polynomial fit to generate an evaluation of the integral.

The integration methods (11.66) give values for the integrals over only a small part of the integration domain. The trapezoidal rule, for instance, only gives a value for $t \in [t_0, t_1]$. However, our fundamental aim is to evaluate the integral over the entire domain $t \in [a, b]$. Assuming once again that our interval is divided as $a = t_0 < t_1 < t_2 < \cdots < t_{m-1} = b$, then the trapezoidal rule applied over the interval gives the total integral

$$\int_a^b f(t)dt \approx Q[f] = \sum_{j=1}^{m} \frac{\Delta t}{2} \left(f_j + f_{j+1}\right) . \tag{11.67}$$

Writing out this sum gives

$$\begin{aligned} \sum_{j=1}^{m} \frac{\Delta t}{2} \left(f_j + f_{j+1}\right) &= \frac{\Delta t}{2}(f_0 + f_1) + \frac{\Delta t}{2}(f_1 + f_2) + \cdots + \frac{\Delta t}{2}(f_m + f_{m-1}) \\ &= \frac{\Delta t}{2}(f_0 + 2f_1 + 2f_2 + \cdots + 2f_m + f_{m-1}) \\ &= \frac{\Delta t}{2}\left(f_0 + f_{m-1} + 2\sum_{j=1}^{m} f_j\right). \end{aligned} \tag{11.68}$$

The final expression no longer double counts the values of the points between f_0 and f_{m-1}. Instead, the final sum only counts the intermediate values once, thus making the algorithm about twice as fast as the previous sum expression. These are computational savings which should always be exploited if possible.

POD Modes from Quadrature Rules

Any of these algorithms could be used to approximate the two-point correlation tensor $\mathbf{R}(\xi, x)$. The method of snapshots implicitly uses the trapezoidal rule to produce the snapshot matrix $\mathbf{X}$. Specifically, recall that

$$\mathbf{X} = \begin{bmatrix} | & | & & | \\ \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_m \\ | & | & & | \end{bmatrix} \tag{11.69}$$

where the columns $\mathbf{u}_k \in \mathbb{C}^n$ may be measurements from simulations or experiments. The SVD of this matrix produces the modes used to produce a low-rank embedding $\boldsymbol{\Psi}$ of the data.

One could alternatively use a higher-order quadrature rule to produce a low-rank decomposition. Thus the matrix (11.69) would be modified to

$$\mathbf{X} = \begin{bmatrix} | & | & | & | & | & & | & | \\ \mathbf{u}_1 & 4\mathbf{u}_2 & 2\mathbf{u}_3 & 4\mathbf{u}_4 & 2\mathbf{u}_5 & \cdots & 4\mathbf{u}_{m-1} & \mathbf{u}_m \\ | & | & | & | & | & & | & | \end{bmatrix} \tag{11.70}$$

where the Simpson's rule quadrature formula is used. Simpson's rule is commonly used in practice as it is simple to execute and provides significant improvement in accuracy over the trapezoidal rule. Producing this matrix simply involves multiplying the data matrix on the

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

right by $\begin{bmatrix}1 & 4 & 2 & 4 & \cdots & 2 & 4 & 1\end{bmatrix}^T$. The SVD can then be used to construct a low-rank embedding $\boldsymbol{\Psi}$. Before approximating the low-rank solution, the quadrature weighting matrix must be undone. To our knowledge, very little work has been done in quantifying the merits of various quadrature rules. However, the interested reader should consider the optimal snapshot sampling strategy developed by Kunisch and Volkwein [315].

11.5 POD with Symmetries: Rotations and Translations

The POD method is not without its shortcomings. It is well known in the POD community that the underlying SVD algorithm does handle invariances in the data in an optimal way. The most common invariances arise from translational or rotational invariances in the data. Translational invariance is observed in the simple phenomenon of wave propagation, making it difficult for correlation to be computed since critical features in the data are no longer aligned snapshot to snapshot.

In what follows, we will consider the effects of both translation and rotation. The examples are motivated from physical problems of practical interest. The important observation is that unless the invariance structure is accounted for, the POD reduction will give an artificially inflated dimension for the underlying dynamics. This challenges our ability to use the POD as a diagnostic tool or as the platform for reduced order models.

Translation: Wave Propagation

To illustrate the impact of translation on a POD analysis, consider a simple translating Gaussian propagating with velocity c.

$$u(x,t) = \exp\left[-(x - ct + 15)^2\right]. \tag{11.71}$$

We consider this solution on the space and time intervals $x \in [-20, 20]$ and $t \in [0, 10]$. The following code produces the representative translating solution and its low-rank representation.

Code 11.5 Translating wave for POD analysis.

```
n=200; L=20; x=linspace(-L,L,n); y=x;   % space
m=41; T=10; t=linspace(0,T,m);          % time
c=3;   % wave speed

X=[];
for j=1:m
    X(:,j)=exp(-(x+15-c*t(j)).^2).';   % data snapshots
end
[U,S,V]=svd(X);   % SVD decomposition
```

Figure 11.7(a) demonstrates the simple evolution to be considered. As is clear from the figure, the translation of the pulse will clearly affect the correlation at a given spatial location. Naive application of the SVD does not account for the translating nature of the data. As a result, the singular values produced by the SVD decay slowly as shown in Fig. 11.7(b) and (c). In fact, the first few modes each contain approximately 8% of the variance.

The slow decay of singular values suggests that a low-rank embedding is not easily constructed. Moreover, there are interesting issues interpreting the POD modes and their

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

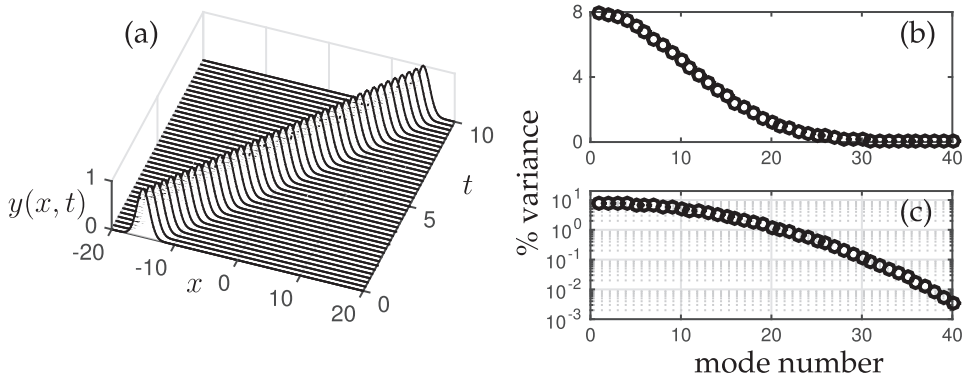

Figure 11.7 (a) Translating Gaussian with speed $c = 3$. The singular value decomposition produces a slow decay of the singular values which is shown on a (b) normal and (c) logarithmic plot.

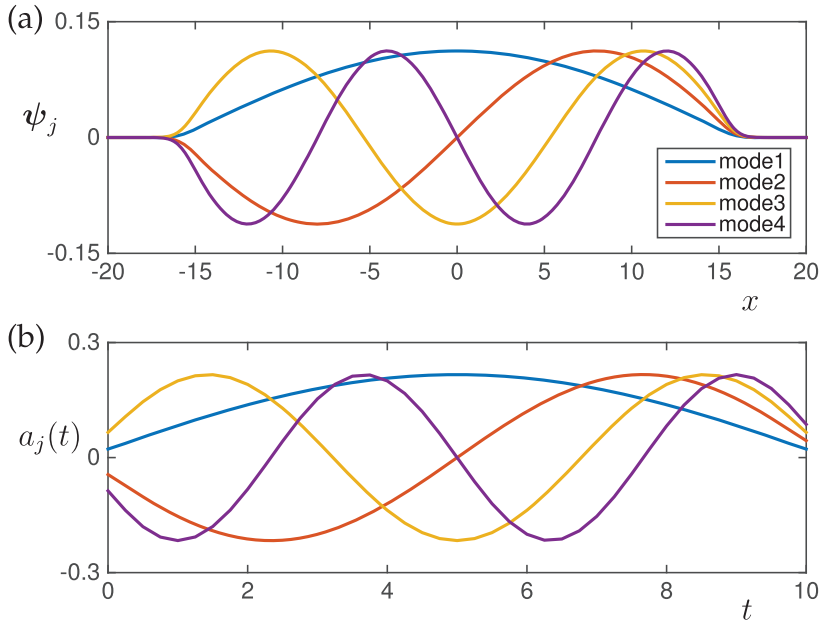

Figure 11.8 First four spatial modes (a) (first four columns of the **U** matrix) and temporal modes (b) (first four columns of the **V** matrix). A wave translating at a constant speed produces Fourier mode structures in both space and time.

time dynamics. Fig. 11.8 shows the first four spatial (**U**) and temporal (**V**) modes generated by the SVD. The spatial modes are global in that they span the entire region where the pulse propagation occurred. Interestingly, they appear to be Fourier modes over the region where the pulse propagated. The temporal modes illustrate a similar Fourier mode basis for this specific example of a translating wave propagating at a constant velocity.

The failure of POD in this case is due simply to the translational invariance. If the invariance is *removed*, or factored out [457], before a data reduction is attempted, then the POD method can once again be used to produce a low-rank approximation. In order

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

Figure 11.9 Spiral waves (a) $u(x, y)$, (b) $|u(x, y)|$ and (c) $u(x, y)^5$ on the domain $x \in [-20, 20]$ and $y \in [-20, 20]$. The spirals are made to spin clockwise with angular velocity ω.

to remove the invariance, the invariance must first be identified and an auxiliary variable defined. Thus we consider the dynamics rewritten as

$$u(x, t) \to u(x - c(t)) \tag{11.72}$$

where $c(t)$ corresponds to the translational invariance in the system responsible for limiting the POD method. The parameter c can be found by a number of methods. Rowley and Marsden [457] propose a template based technique for factoring out the invariance. Alternatively, a simple center-of-mass calculation can be used to compute the location of the wave and the variable $c(t)$ [316].

Rotation: Spiral Waves

A second invariance commonly observed in simulations and data is associated with rotation. Much like translation, rotation moves a coherent, low-rank structure in such a way that correlations, which are produced at specific spatial locations, are no longer produced. To illustrate the effects of rotational invariance, a localized spiral wave with rotation will be considered.

A spiral wave centered at the origin can be defined as follows

$$u(x, y) = \tanh\left[\sqrt{x^2 + y^2}\cos\left(A\angle(x + iy) - \sqrt{x^2 + y^2}\right)\right] \tag{11.73}$$

where A is the number of arms of the spiral, and the $\angle$ denotes the phase angle of the quantity $(x + iy)$. To localize the spiral on a spatial domain, it is multiplied by a Gaussian centered at the origin so that our function of interest is given by

$$f(x, y) = u(x, y)\exp\left[-0.01(x^2 + y^2)\right]. \tag{11.74}$$

This function can be produced with the following code.

Code 11.6 Spiral wave for POD analysis.

```
n=100;
L=20; x=linspace(-L,L,n); y=x;
[X,Y]=meshgrid(x,y);

Xd=[];
for j=1:100
    u=tanh(sqrt(X.^2+Y.^2)).*cos(angle(X+i*Y)-(sqrt(X.^2+Y.^2))+
        j/10);
    f=exp(-0.01*(X.^2+Y.^2));
    uf=u.*f;
    Xd(:,j)=reshape(uf,n^2,1);
```

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

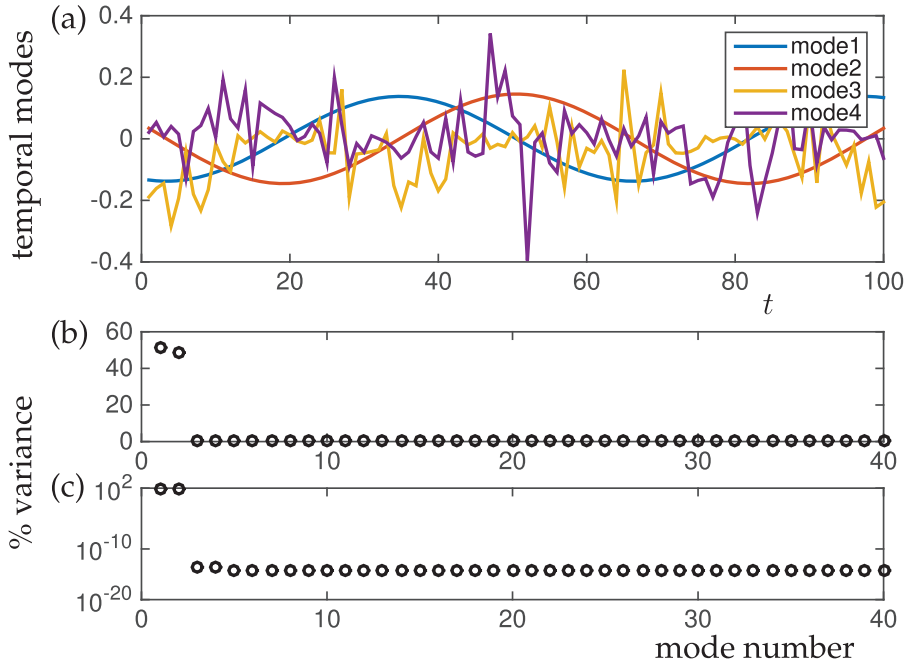

Figure 11.10 (a) First four temporal modes of the matrix $\mathbf{V}$. To numerical precision, all the variance is in the first two modes as shown by the singular value decay on a normal (b) and logarithmic (c) plot. Remarkably, the POD extracts exactly two modes (See Fig. 11.11) to represent the rotating spiral wave.

```
    pcolor(x,y,uf), shading interp, colormap(hot)
end
```

Note that the code produces snapshots which advance the phase of the spiral wave by $j/10$ each pass through the for loop. This creates the *rotation* structure we wish to consider. The rate of spin can be made faster or slower by lowering or raising the value of the denominator respectively.

In addition to considering the function $u(x, y)$, we will also consider the closely related functions $|u(x, y)|$ and $u(x, y)^5$ as shown in Fig. 11.9. Although these three functions clearly have the same underlying function that rotates, the change in functional form is shown to produce quite different low-rank approximations for the rotating waves.

To begin our analysis, consider the function $u(x, y)$ illustrated in Fig. 11.9(a). The SVD of this matrix can be computed and its low-rank structure evaluated using the following code.

Code 11.7 SVD decomposition of spiral wave.

```
[U,S,V]=svd(Xd,0);

figure(2)
subplot(4,1,3)
plot(100*diag(S)/sum(diag(S)),'ko','Linewidth',[2])
subplot(4,1,4)
semilogy(100*diag(S)/sum(diag(S)),'ko','Linewidth',[2])
subplot(2,1,1)
```

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

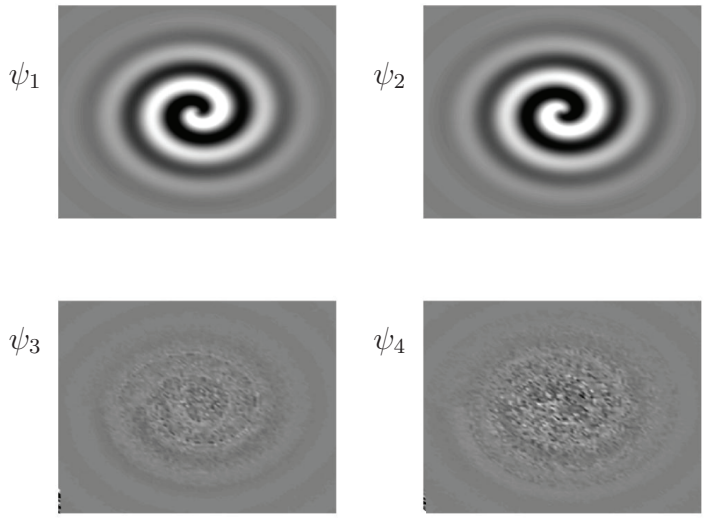

Figure 11.11 First four POD modes associated with the rotating spiral wave $u(x, y)$. The first two modes capture all the variance to numerical precision while the third and fourth mode are noisy due to numerical round-off. The domain considered is $x \in [-20, 20]$ and $y \in [-20, 20]$.

```
plot(V(:,1:4),'Linewidth',[2])
figure(3)
for j=1:4
    subplot(4,4,j)
    mode=reshape(U(:,j),n,n);
    pcolor(X,Y,mode), shading interp,caxis([-0.03 0.03]),
        colormap(gray)
end
```

Two figures are produced. The first assesses the rank of the observed dynamics and the temporal behavior of the first four modes in $\mathbf{V}$. Figs. 11.10 (b) and (c) show the decay of singular values on a regular and logarithmic scale respectively. Remarkably, the first two modes capture *all* the variance of the data to numerical precision. This is further illustrated in the time dynamics of the first four modes. Specifically, the first two modes of Fig. 11.10(a) have a clear oscillatory signature associated with the rotation of modes one and two of Fig. 11.11. Modes three and four resemble noise in both time and space as a result of numerical round off.

The spiral wave (11.74) allows for a two-mode truncation that is accurate to numerical precision. This is in part due to the sinusoidal nature of the solution when circumnavigating the solution at a fixed radius. Simply changing the data from $u(x, t)$ to either $|u(x, t)|$ or $u(x, t)^5$ reveals that the low-rank modes and their time dynamics are significantly different. Figs. 11.12 (a) and (b) show the decay of the singular values for these two new functions and demonstrate the significant difference from the two mode evolution previously considered. The dominant time dynamics computed from the matrix $\mathbf{V}$ are also demonstrated. In the case of the absolute value of the function $|u(x, t)|$, the decay of the singular values is slow and never approaches numerical precision. The quintic function suggests a rank $r = 6$ truncation is capable of producing an approximation to numerical precision. This highlights the fact that rotational invariance complicates the POD reduction procedure.

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

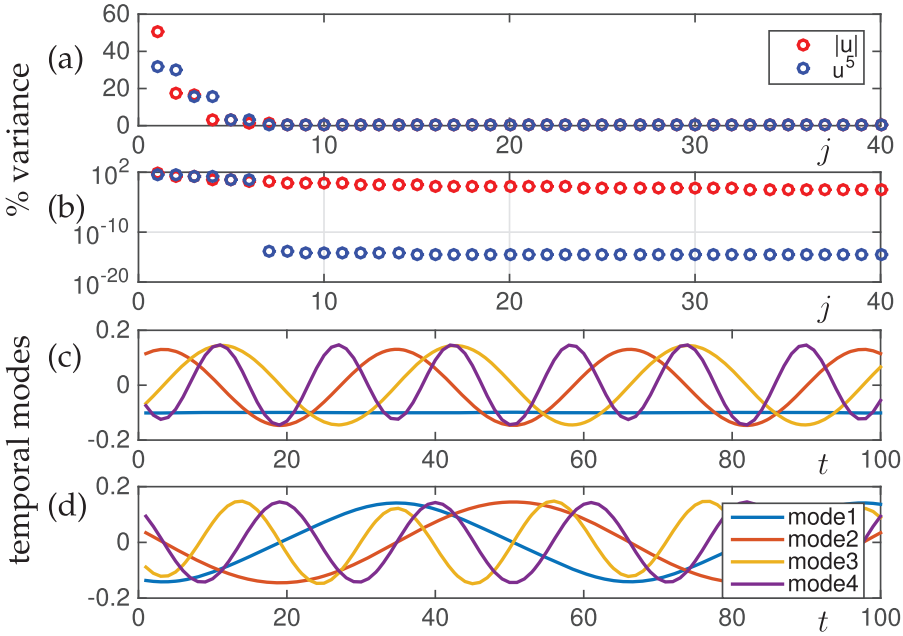

Figure 11.12 Decay of the singular values on a normal (a) and logarithmic (b) scale showing that the function $|u(x,t)|$ produces a slow decay while $u(x,t)^5$ produces an $r = 6$ approximation to numerical accuracy. The first four temporal modes of the matrix $\mathbf{V}$ are shown for these two functions in (c) and (d) respectively.

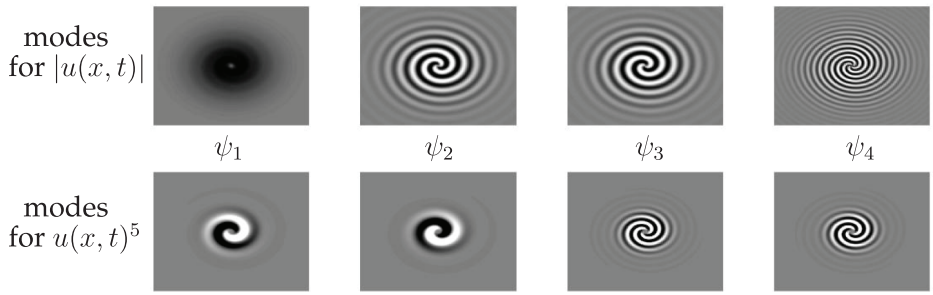

Figure 11.13 First four POD modes associated with the rotating spiral wave $|u(x, y)|$ (top row) and $u(x,t)^5$ (bottom row). Unlike our previous example, the first four modes do not capture all the variance to numerical precision, thus requiring more modes for accurate approximation. The domain considered is $x \in [-20, 20]$ and $y \in [-20, 20]$.

After all, the only difference between the three rotating solutions is the actual shape of the rotating function as they are all rotating with the same speed.

To conclude, invariances can severely limit the POD method. Most notably, it can artificially inflate the dimension of the system and lead to compromised interpretability. Expert knowledge of a given system and its potential invariances can help frame mathematical strategies to remove the invariances, i.e. re-aligning the data [316, 457]. But this strategy also has limitations, especially if two or more invariant structures are present. For instance,

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press

if two waves of different speeds are observed in the data, then the methods proposed for removing invariances will fail to capture both wave speeds simultaneously. Ultimately, dealing with invariances remains an open research question.

Suggested Reading

Texts

(1) **Certified reduced basis methods for parametrized partial differential equations**, by J. Hesthaven, G. Rozza and B. Stamm, 2015 [244].

(2) **Reduced basis methods for partial differential equations: An introduction**, by A. Quarteroni, A. Manzoni and N. Federico, 2015 [442].

(3) **Model reduction and approximation: Theory and algorithms**, by P. Benner, A. Cohen, M. Ohlberger and K. Willcox, 2017 [54].

(4) **Turbulence, coherent structures, dynamical systems and symmetry**, by P. Holmes, J. L. Lumley, G. Berkooz and C. W. Rowley, 2012 [251].

Papers and Reviews

(1) **A survey of model reduction methods for parametric systems**, by P. Benner, S. Gugercin and K. Willcox, *SIAM Review*, 2015 [53].

(2) **Model reduction using proper orthogonal decomposition**, by S. Volkwein, *Lecture Notes, Institute of Mathematics and Scientific Computing, University of Graz*, 2011 [542].

(3) **The proper orthogonal decomposition in the analysis of turbulent flows**, by G. Berkooz, P. Holmes and J. L. Lumley, *Annual Review of Fluid Mechanics*, 1993 [57].

https://doi.org/10.1017/9781108380690.012 Published online by Cambridge University Press