An introduction to the proper orthogonal decomposition

Anindya Chatterjee*

Department of Engineering Science and Mechanics, Penn State University, University Park, Pennsylvania, PA, 16802, USA

A tutorial is presented on the Proper Orthogonal Decomposition (POD), which finds applications in computationally processing large amounts of high-dimensional data with the aim of obtaining low-dimensional descriptions that capture much of the phenomena of interest. The discrete version of the POD, which is the singular value decomposition (SVD) of matrices, is described in some detail. The continuous version of the POD is outlined. Low-rank approximations to data using the SVD are discussed. The SVD and the eigenvalue decomposition are compared. Two geometric interpretations of the SVD/POD are given. Computational strategies (using standard software) are mentioned. Two numerical examples are provided: one shows low-rank approximations of a surface, and the other demonstrates simple *a posteriori* analysis of data from a simulated vibroimpact system. Some relevant computer code is supplied.

1. Introduction

COMPUTERS have increased our capacity to not only simulate complicated systems, but also to collect and analyse large amounts of data. Using personal computers, it is now unremarkable to record data at, say, 20 kHz for a few hours. One might then process hundreds of millions of data points to obtain a few quantities of final interest. For example, expensive machinery might be instrumented and monitored over days, with the sole objective of efficiently scheduling maintenance.

This article provides an introduction to the Proper Orthogonal Decomposition (POD) which is a powerful and elegant method of data analysis aimed at obtaining low-dimensional approximate descriptions of high-dimensional processes. The POD was developed by several people (among the first was Kosambi[1]), and is also known as *Principal Component Analysis*, the *Karhunen–Loéve Decomposition*, and the *single value decomposition*. The POD has been used to obtain approximate, low-dimensional descriptions of turbulent fluid flows[2], structural vibrations[3,4], and insect gait[5], and has been used for damage detection[6], to name a few applications in dynamic systems. It has also been extensively used in image processing, signal analysis and data compression. For references to the many sources of the POD, for applications of the POD in a variety of fields, as well as for a nice treatment that complements this tutorial, the reader is encouraged to read Holmes, Lumley and Berkooz[2] (chapter 3).

Data analysis using the POD is often conducted to extract 'mode shapes' or basis functions, from experimental data or detailed simulations of high-dimensional systems, for subsequent use in Galerkin projections that yield low-dimensional dynamical models (see ref. 2). This article concentrates on the data analysis aspect, and subsequent reduced-order modelling is not discussed.

2. Motivation

Suppose we wish to approximate a function $z(x, t)$ over some domain of interest as a finite sum in the variables-separated form

$$z(x,t) \approx \sum_{k=1}^{M} a_k(t)\mathbf{f}_k(x), \tag{1}$$

with the reasonable expectation that the approximation becomes exact in the limit as M approaches infinity, except possibly on a set of measure zero (readers unfamiliar with measure theory may ignore it if they deal with finite-dimensional calculations; and consult, e.g. Rudin[7] otherwise).

While in eq. (1) there is no fundamental difference between t and x, we usually think of x as a spatial coordinate (possibly vector-valued) and think of t as a temporal coordinate.

The representation of eq. (1) is not unique. For example, if the domain of x is a bounded interval $\mathbf{X}$ on the real line, then the functions $\mathbf{f}_k(x)$ can be chosen as a Fourier series, or Legendre polynomials, or Chebyshev polynomials, and so on. For each such choice of a sequence $\mathbf{f}_k(x)$ that forms a basis for some suitable class of functions $z(x, t)$ (see note 1), the sequence of time-functions $a_k(t)$ is different. That is, for sines and cosines we get one sequence of functions $a_k(t)$ is different. That is, for sines and cosines we get one sequence of functions $a_k(t)$, for Legendre polynomials we get another, and so on. The POD is concerned with one possible choice of the functions $\mathbf{f}_k(x)$.

e-mail: anindya@crash.esm.psu.edu

If we have chosen orthonormal basis functions, i.e.

$$\int_{\mathbf{x}} \mathbf{f}_{k_1}(x)\mathbf{f}_{k_2}(x)\,\mathrm{d}x = \begin{cases} 1 \text{ if } k_1 = k_2 \\ 0 \text{ otherwise} \end{cases},$$

then

$$a_k(t) = \int_{\mathbf{x}} z(x,t)\mathrm{f}_k(x)\,\mathrm{d}x, \tag{2}$$

i.e. for orthonormal basis functions, the determination of the coefficient function $a_k(t)$ depends only on $\mathbf{f}_k(x)$ and not on the other $\mathbf{f}$'s.

What criteria should we use for selecting the functions $\mathbf{f}_k$? Orthonormality would be useful. Moreover, while an approximation to any desired accuracy in eq. (1) can always be obtained if M can be chosen large enough, we may like to choose the $\mathbf{f}_k(x)$ in such a way that the approximation for each M is as good as possible in a least squares sense. That is, we would try to find, once and for all, a sequence of orthonormal functions $\mathbf{f}_k(x)$ such that the first two of these functions give the best possible two-term approximation, the first seven give the best possible seven-term approximation, and so on. These special, ordered, orthonormal functions are called the *proper orthogonal nodes* for the function $z(x, t)$. With these functions, the expression in eq. (1) is called the POD of $z(x, t)$.

3. Theory: Finite-dimensional case

Consider a system where we take measurements of m state variables (these could be from m strain gauges on a structure, or m velocity probes in a fluid, or a mixture of two kinds of probes in a system with flow-induced vibrations, etc.). Assume that at N instants of time, we take N sets of m simultaneous measurements at these m locations. We arrange out data in an $N \times m$ matrix A, such that element A_{ij} is the measurement from the jth probe taken at the ith time instant.

The m state variables are *not* assumed to be measured by transducers that are arranged in some straight line in physical space. We merely assume that the transducers have been numbered for identification; and that their outputs have been placed side by side in the matrix A. In the actual physical system these measurements might represent one spatial dimension (e.g. accelerometers on a beam), or more than one spatial dimension (e.g. pressure probes in a three-dimensional fluid flow experiment). Each physical transducer may itself measure more than one scalar quantity (e.g. triaxial accelerometers); in such cases, the different scalar time series from the same physical transducer are arranged in different columns of A. The final result of the data collection is assumed here to be the $N \times m$ matrix A.

It is common to subtract, from each column of A, the mean value of that column. Whether or not this is done does not affect the basic calculation, though it affects the interpretation of the results (see section 3.5 below).

3.1 *The singular value decomposition*

We now compute the singular value decomposition (SVD) of the matrix A, which is of the form (for a discussion of the SVD, see ref. 8):

$$A = U\Sigma V^T, \tag{3}$$

where U is an $N \times N$ orthogonal matrix, V is an $m \times m$ orthogonal matrix, the superscript T indicates matrix transpose, and Σ is an $N \times m$ matrix with all elements zero except along the diagonal. The diagonal elements Σ_{ii} consist of $r = \min(N, m)$ nonnegative numbers s_i, which are arranged in decreasing order, i.e. $\mathsf{s}_1 \geq \mathsf{s}_2 \geq \ldots \geq \mathsf{s}_r \geq 0$. The s's are called the singular values of A (and also of A^T) and are unique. The rank of A equals the number of nonzero singular values it has. In the presence of noise, the number of singular values larger than some suitably small fraction of the largest singular value might be taken as the 'numerical rank'.

Since the singular values are arranged in a specific order, the index k of the kth singular value will be called the *singular value number* (see Figures 1 *e* and 2 *c*).

3.2 *Correspondence with eqs (1) and (2)*

In eq. (3), let $U\Sigma = Q$. Then the matrix Q is $N \times m$, and $A = QV^T$. Letting q_k be the kth column of Q and v_k be the kth column of V, we write out the matrix product as

$$A = QV^T = \sum_{k=1}^{m} q_k v_k^T. \tag{4}$$

Equation (4) is the discrete form of eq. (1). The function $z(x, t)$ is represented here by the matrix A. The function $a_k(t)$ is represented by the column matrix q_k. The function $\mathbf{f}_k(x)$ is represented by the row matrix v_k^T. The approximation of eq. (1) is now exact because the dimension is finite. Due to the orthonormality of the columns of V, eq. (2) corresponds to multiplying eq. (4) by one of the v's on the right.

3.3 *Lower-rank approximations to A*

For any $k < r$, the matrix Σ_k obtained by setting $\mathsf{s}_{k+1} = \mathsf{s}_{k+2} = \ldots = \mathsf{s}_r = 0$ in Σ can be used to calculate an *optimal* rank k approximation (see note 2) to A, given by

$$A_k = U\Sigma_k V^T. \tag{5}$$

In computations, one would actually replace U and V with the matrices of their first k columns; and replace Σ_k by its leading $k \times k$ principal minor, the submatrix consisting of Σ's first k rows and first k columns (see the computer code in Appendix A.1).

The optimality of the approximation in eq. (5) lies in the fact that no other rank k matrix can be closer to A in the Frobenius norm (square root of the sums of squares of all the elements), which is a discrete version of the L_2 norm; or in the 2-norm (the 2-norm of a matrix is its largest singular value). Thus, the first k columns of the matrix V (for any k) give an optimal orthonormal basis for approximating the data. Note that V is determined once and for all: the rank k of the approximating matrix can be chosen afterwards, and arbitrarily, with guaranteed optimality for each k.

The columns of V are the proper orthogonal modes.

3.4 *SVD vs eigenvalue decomposition*

Consider the differences between the SVD and eigenvalue decomposition. The SVD can be computed for non-square matrices, while the eigenvalue decomposition is only defined for square matrices; the SVD remains within real arithmetic whenever A is real, while eigenvalues and eigenvectors of unsymmetric real matrices can be complex; the left and right singular vectors (columns of U and of V respectively) are each orthogonal, while eigenvectors of

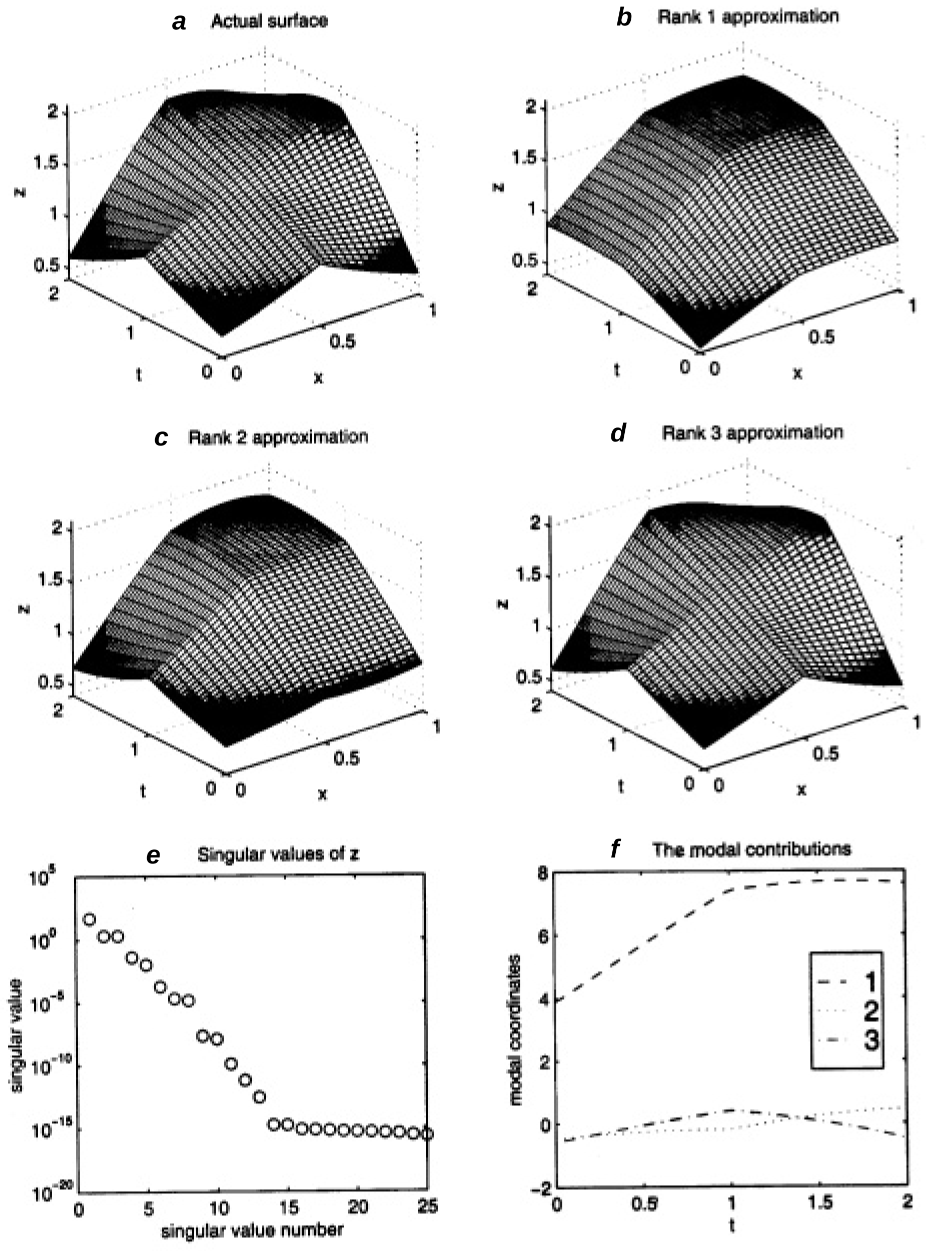

Figure 1. Approximation of surface example.

unsymmetric matrices need not be orthogonal even when a full set exists; and finally, while an eigenvector $\mathbf{y}$ (say) and its image $A\mathbf{y}$ are in the same direction, a right-singular vector v_k (kth column of V) and its image Av_k need not be in the same direction or even in spaces of the same dimension.

However, the SVD does have strong connections with the eigenvalue decomposition. On premultiplying eq. (3) with its transpose and noting that $V^{-1} = V^T$, we see that V is the matrix of eigenvectors of the symmetric $m \times m$ A^TA matrix A^TA, while the squares of the singular values are the $r = \min(N, m)$ largest eigenvalues (see note 3) of A^TA.

Similarly, on premultiplying the transposed eq. (3) with itself and noting that $U^{-1} = U^T$, we see that U is the matrix of eigenvectors of the symmetric $N \times N$ matrix AA^T, and the squares of the singular values are the $r = \min(N, m)$ largest eigenvalues of AA^T.

If A is symmetric and positive definite, then its eigenvalues are also its singular values, and $U = V$. If A is symmetric with some eigenvalues negative (they will be real, because A is symmetric), then the singular values are the magnitudes of the eigenvalues, and U and V are the same up to multiplication by minus one for the columns corresponding to negative eigenvalues.

3.5 *Geometric interpretations*

The SVD of a matrix has a nice geometric interpretation. An $N \times m$ matrix A is a linear operator that maps vectors from an m-dimensional space, say S_1, to an N-dimensional space, say S_2. Imagine the unit sphere in S_1, the set of vectors of unit magnitude (square root of sum of squares of elements). This unit sphere gets mapped to an ellipsoid in S_2. The singular values $\sigma_1, \sigma_2, \ldots$ are the lengths of the principal radii of that ellipsoid. The directions of these principal radii are given by the columns of U. The pre-images of these principal radii are the columns of V.

A second geometric interpretation may be more illuminative for POD applications. We now view the $N \times m$ matrix A as a list of the coordinates of N points in an m-dimensional space. For any $k \le m$, we seek a k-dimensional subspace for which the mean square distance of the points, from the subspace, is minimized. A basis for this subspace is given by the first k columns of V.

Recall that it is common in POD applications to subtract from each column of A the mean of that column. This mean-shift ensures that the N-point 'cloud' is centered around the origin. Figure 3 shows how the one-dimensional optimal subspace basis vector, indicated in each case by a grey arrow, depends on where the point cloud is centered (by

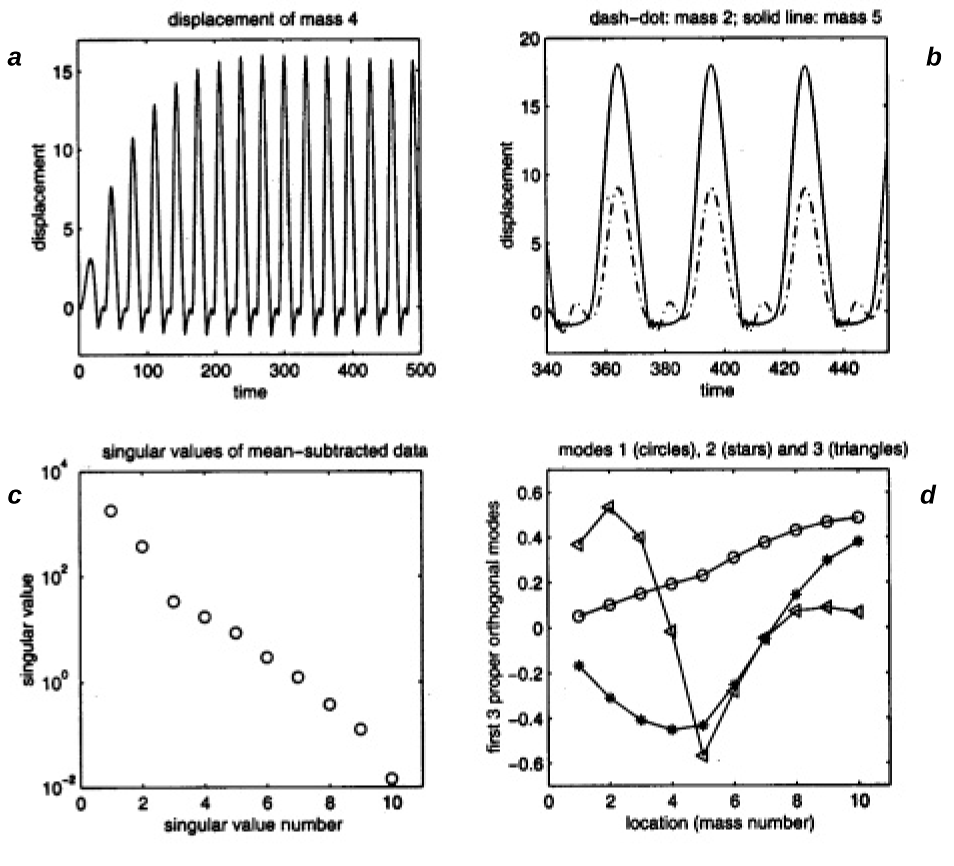

Figure 2. Vibrioimpact example results.

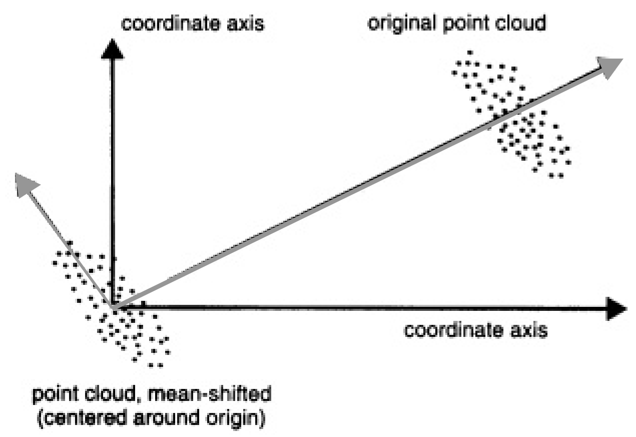

Figure 3. Effect of shifting the mean of the point cloud to the origin.

definition, a subspace must pass through the origin).

3.6 *Computations*

The matrix V can be found by computing the SVD directly using commercial software like Matlab. Alternatively, the calculation can be indirectly carried out using eigenvalue/eigenvector routines, using A^TA as mentioned in section 3.4.

If $m \gg N$, it is more efficient to first compute the matrix U as the matrix of eigenvectors of AA^T. This method is called the 'method of snapshots' in the POD literature. For example, in trying to find low-rank representations of a digital image that evolves over time, one might have a 1000×1000 pixel image ($m = 10^6$), but only on the order of $N \approx 10^3$ images; in such cases, the method of snapshots might be used. Once U is known, premultiplying eq. (3) by U^T gives

$$U^TA = \Sigma V^T \quad . \tag{6}$$

The product in eq. (6) is obviously still $N \times m$. The last $m - N$ columns of Σ are zero; the bottom $m - N$ rows of V^T, or the last $m - N$ columns of V, are multiplied by zeros and are indeterminate and irrelevant. If there are k nonzero singular values, then the first k rows of ΣV^T are nonzero and orthogonal. Their norms are the singular values. Normalizing them to unit magnitude gives the corresponding proper orthogonal modes v_i.

4. Theory: Infinite-dimensional versions

For most users of the POD, the simpler theory for the discrete case suffices. Experimental data are always discrete, and in any case integral equations (which arise in the infinite-dimensional case) are usually solved numerically by discretization. However, those requiring the infinite-dimensional versions may wish to consult, e.g. Holmes, Lumley and Berkooz[2] as well as a text on integral equations, such as Porter and Stirling[9]. For completeness, the main points are outlined here.

Infinite-dimensional PODs are solved as eigenvalue problems. The issue to resolve is what A^TA should mean for infinite-dimensional problems.

To this end, note that in the finite-dimensional version discussed in preceding subsections, element (i, j) of the matrix $B = A^TA$ is

$$B_{ij} = \sum_{k=1}^{N} A_{k_i} A_{k_j} \, , \tag{7}$$

with the sum being carried out on the 'time' variable or the row-index of A.

In the finite-dimensional-in-space but continuous-time version of the POD, the vector inner products involved in computing A^TA become integrals. We simply take

$$B_{ij} = \frac{1}{T}\int_{t_0}^{t_0+T} z(x_i, t) z(x_j, t) \, \mathrm{d}t, \tag{8}$$

where for any finite T the correspondence between the integral (eq. (8)) and the sum (eq. (7)) is clear. Since B is still just a matrix, it still has only a finite number of eigenvalues.

In the discrete case, with finite data, the factor of $1/T$ can be ignored since it affects only the absolute values of the σ's and leaves their relative magnitudes unchanged; the matrices U and V are unaffected as well.

Before going on to the fully infinite-dimensional case, we briefly consider the significance of the averaging time duration T in eq. (8). In applications of the continuous-time POD to steady state problems, one usually assumes that in eq. (8) a well-defined limit is attained, independent of t_0, as T approaches infinity. In practical terms this means that *if* the POD is being used to obtain a low-dimensional description of the long-term or steady state behaviour, then the data should be collected over a time period much longer than the time scale of the inherent dynamics of the system. However, applications of the POD need not only be to steady state problems. In studying the impulse response, say, of a complicated structure, the steady state solution may be the zero solution. However, a finite-time POD may well yield useful insights into the transient behaviour of the structure.

As was the case with subtracting vs not subtracting the means from the columns of A, whether or not the data collection time T was long enough to accurately capture the steady state behaviour does not affect the basic calculation but affects the interpretation of results.

We now consider the fully infinite-dimensional version of the POD. Now the rows of A are replaced by functions of space, and there is an infinite sequence of eigenvalues (with associated eigenfunctions). B is not a matrix anymore, but a function of two variables (say x_1 and x_2).

$$B(x_1, x_2) = \frac{1}{T}\int_{t_0}^{t_0+T} z(x_1, t) z(x_2, t)\, \mathrm{d}t, \tag{9}$$

where x_1 and x_2 are continuous variables both defined on some domain $\mathbf{X}$. How should we interpret the eigenvalues of B as given by eq. (9)? The eigenvalue problem in the discrete $m \times m$ case, $B\psi = \lambda\psi$, can be written out in component form as

$$\sum_{j=1}^{m} B_{ij}\psi_j = \lambda\psi_i. \tag{10}$$

For the B of eq. (9), the sum of eq. (10) becomes an integral, ψ becomes a function of x, and we have the integral equation

$$\int_{\mathbf{X}} B(x_1, x_2)\psi(x_2)\, \mathrm{d}x_2 = \lambda\psi(x_1).$$

It is clear that the above integral equation has the same form whether the space variable x is *scalar* valued or vector valued.

5. Numerical examples

5.1 *Approximation of a surface*

Let z be given by

$$z(x, t) = \mathrm{e}^{-|(x-0.5)(t-1)|} + \sin(xt), \quad 0 \le x \le 1, \quad 0 \le t \le 2. \tag{11}$$

Imagine that we 'measure' this function at 25 equally spaced x points, and 50 equally spaced instants of t. The surface $z(x, t)$ is shown in Figure 1 *a*.

Arranging the data in a matrix Z, we compute the SVD of Z, and then compute (see section 3.3) rank 1, rank 2, and rank 3 approximations to Z, as shown in Figure 1 *b–d*, The computer code used for this calculation is provided in the appendix. (The means were *not* subtracted from the columns for this example.)

The rank 3 approximation (Figure 1 *d*) looks indistinguishable from the actual surface (Figure 1 *a*). This is explained by Figure 1 *e*, which shows the singular values of Z. Note how the singular values decrease rapidly in magnitude, with the fourth one significantly smaller than the third. (The numerical values are 47.5653, 2.0633, 2.0256, 0.0413, 0.0106, ...)

Note that in this example without noise, the computed singular values beyond number 14 flatten out at the numerical roundoff floor around 10^{-15}. The actual singular values beyond number 14 should be smaller, and an identical computation with more digests of precision should show the computed singular values flattening out at a smaller magnitude. Conversely, perturbing the data matrix by zero-mean random numbers of typical magnitude (say) 10^{-8}, causes the graph of singular values to develop an obvious elbow at about that value. For experimental data with noise, the SVD of the data matrix can sometimes provide an empirical estimate of where the noise floor is.

So far in this example, we have merely computed lower-rank approximations to the data, and the use of the SVD in the calculation may be considered incidental. Now, suppose we wish to interpret the results in terms of mode shapes, i.e. in the context of the POD. The first 3 columns of V provide the 3 dominant x-direction mode shapes, and on projecting the data onto these mode shapes we can obtain the time histories of the corresponding modal 'coordinates'.

The calculation of the modal coordinates is straightforward. Using eq. (4), the kth modal coordinate q_k is simply $u_k\sigma_k$, where u_k is the kth column of U (assuming U is available from the SVD). Alternatively, if only the proper orthogonal modes V are available, then the projection calculation is simply $q_k = Av_k$, where v_k is the kth column of V.

The modal coordinates for the surface given by eq. (11) are plotted in Figure 1 *f*. The first coordinate is obviously dominant (the first singular value is dominant), while the second and third have comparable magnitude (singular values 2 and 3 are approximately equal).

5.2 *Proper orthogonal modes in a simplified vibroimpact problem*

Let us consider the one-dimensional, discrete system shown in Figure 4. Ten identical masses m are connected to each other (and to a wall at one end) by identical linear springs of stiffness k and identical dashpots of coefficient c. The forcing on each mass is identical with

$$F_1(t) = F_2(t) = \ldots = F_{10}(t) = A \sin \omega t.$$

The fifth mass has a hard stop which intermittently contacts a nonlinear spring; the spring force is Kx^3 when the displacement x of mass 5 is negative, and zero otherwise.

The equations of motion for this system are easy to write, and are not reproduced here. The system was numerically simulated using *ode23*, a Matlab routine implementing a low-order Runge–Kutta method with adaptive step size control. The program's default error tolerances were used (10^{-3} for relative error and 10^{-6} for absolute error). The solution, though adaptively refined using internal error estimates, was finally provided by the routine at equally-spaced points in time (this is a convenient feature of this routine). The simulation was from t = 500 to t = 6000 equal intervals. For initial conditions, all ten displacements and velocities were taken to be zero. In the simulation, the on/off switchings of the cubic spring were not monitored, since the nonlinearity is three times differentiable, the routine uses a low-order method, there is adaptive error control, and overall speed of computation was not a concern in this simulation.

The parameters used in the simulation were:

$$m = 1,\ k = 1,\ c = 0.3,\ A = 0.08,\ \text{w} = 0.2,\ \text{and}\ k = 5.$$

Figure 2 shows the results of the simulation. Figure 2 *a* shows the displacement vs time of mass 4, which is seen settling to a periodic motion. In Figure 2 *b*, the displacements vs time of masses 2 and 5 are shown over a shorter time period. The 'impacts' of mass 5 are clearly seen. Figure 2 *c* depicts the singular values of the matrix of positions (velocities are not used). Note that the mean displacements of the masses were nonzero because of the one-sided spring at mass 5, and in the computations the mean displacements were subtracted from the columns of the data matrix (see section 3.5). Finally, the first three proper orthogonal modes are shown in Figure 2 *d*. The relative magnitudes of the maximum displacements of masses 2 and 5, in Figure 2 *b*, are consistent with the relative magnitudes of the corresponding elements of the first, dominant, proper orthogonal mode shown in Figure 2 *d*. Note that in Figure 2 *d* the three mode shapes have comparable magnitudes. This is as it should be, because the mode shapes themselves are all normalized to unit magnitude; it is time-varying coefficient of the third mode, say, which is much smaller than that of the first mode.

The computer code used for these calculations (to process the previously computed displacement vs time data) is given in the appendix.

From Figure 2 *c* and *d*, we observe that the first two singular values are together much larger than the rest; as such, the response is dominated by the first two modes. However, the effect of the nonsmooth impacting behaviour shows up more clearly in the third mode (Figure 2 *d*), although it has a relatively small overall amplitude. This is consistent with Figure 2 *b*, where it is seen that the displacements at a 'typical' location (mass 2) are significantly smoother than those at the impact location (mass 5). The displacement of mass 5 itself is slow for relatively long portions of time, and has some localized high-frequency, small-amplitude behaviour near the impacts. Roughly speaking, the softness of the impact contact (cubic spring) as well as the damping in the structure lead to smooth-looking mode shapes capturing most of the action most of the time. In other words, the 'fast' dynamics introduced by the impacts is localized in space (meaning it is significant only at and near the impact location), and also localized in time (meaning it is significant for only a short period following each impact); and in an overall average sense, the amplitude of the high-speed, impact-induced oscillations is small. For this reason, the modes that dominate the POD do not show the impact behaviour clearly.

The foregoing example provides a glimpse of how the POD may be used to extract qualitative and quantitative information about the spatial structure of a system's behaviour; and how *a posteriori* data analysis might lead to an improved understanding of the system.

6. Some words of caution

6.1 *Sensitivity to coordinate changes*

The POD is sensitive to coordinate changes: it processes numbers, whose physical meaning it does not know. The SVD of a matrix A is not the same as the SVD of another matrix AB, where B is an invertible $n \times n$ matrix (corresponding to talking linear changes of variables). In particular, inappropriate scaling of the variables being measured (easily possible in systems with mixed measurements such as accelerations, displacements and strain) can lead to misleading if not meaningless results.

In the experimental study of chaotic systems one sometimes uses delay coordinate embedding (see, e.g. Ott[10]); which implicitly involves a strongly *nonlinear* change of variables. Readers may like Ravindra's brief note[11] discussing one inappropriate use of the POD in this connection.

6.2 *Subspaces vs surfaces*

The analysis is based on linear combinations of basis functions. The POD cannot distinguish between a featureless cloud of points on some plane, and a circle on the same plane. While the circle obviously has a one-dimensional descriptions, on blindly using the POD one might conclude that the system had two dominant 'modes'.

6.3 *Rank vs information content*

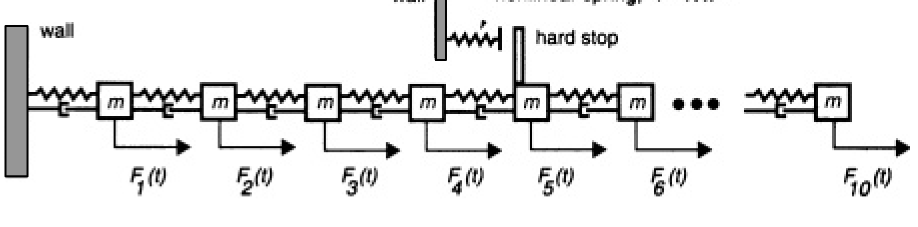

Figure 4. Simple model of a vibrioimpact system.

As the SVD can be used to compute low-rank approximations to data matrices, it is tempting to think that rank is related to information content. To see that this is not necessarily true, consider an $n \times n$ diagonal matrix (rank n) with another $n \times n$ matrix which has the same nonzero entries but in a single column (rank one). Though the ranks differ widely, the information contents of these two matrices may reasonably be called identical.

Incidentally, this diagonal matrix vs single-nonzero-column matrix example has a nice correspondence with two types of physical phenomena: the diagonal matrix corresponds to travelling disturbances (e.g. pulses), while the single-nonzero-column matrix corresponds to disturbances that stay localized (e.g. standing waves). Readers are invited to mimic the surface-approximation example of section 5.1 with the function $z(x, t) = \mathrm{e}^{-a(x-t)^2}$, which represents a travelling hump, with (say) $a = 5$, $-2 \le x \le 2$, $-2 \le t \le 2$, and with (say) a grid of 80 points along each axis. In this example (whose details are not presented here for reasons of space), it is found that the largest several singular values have comparable magnitudes; and low-rank approximations to the surface are poor. For increasing a, the approximation at any fixed less-than-full rank gets poorer. This example shows that some physical systems may be poorly suited to naïve analysis using the POD.

6.4 *Modal 'energies'*

The eigenvalues of A^TA (or the squares of the singular values of A) are sometimes referred to as 'energies' corresponding to the proper orthogonal modes. In signal processing, this energy is not a physical energy. In incompressible fluid mechanics with velocity measurements, this energy is related to the fluid's kinetic energy. However, in structural dynamics problems with, say, displacement and/or velocity measurements, there is generally no direct correspondence between this energy and either the system's kinetic energy or its potential energy or any combination thereof. Thinking of the eigenvalues of A^TA as 'energies' in a general mechanical context is incorrect in principle and may yield misleading results.

For example, consider a two-mass system. Let one mass be 10^{-4} kg and let it vibrate sinusoidally at a frequency ω with an amplitude of 1 m. Let the second mass be 10^4 kg, and let it vibrate sinusoidally at a frequency 2ω with an amplitude of 10^{-2} m. Then the first proper orthogonal mode corresponds to motion of the first mass only, while the second proper orthogonal mode corresponds to motion of the second mass only. The modal 'energy' in the first proper orthogonal mode is 10^4 times larger than that in the second mode. However, the actual average kinetic energy of the first mass (first mode) is 10^4 times *smaller* than that of the second mass (second mode). (Readers who like this example may also enjoy the discussion in Feeny and Kappagantu[4].)

6.5 *Proper orthogonal modes and reduced-order modelling*

Proper orthogonal modes are frequently used in Galerkin projections to obtain low-dimensional models of high-dimensional systems. In many cases, reasonable to excellent models are obtained. However, the optimality of the POD lies in *a posteriori* data reconstruction, and there are no guarantees (as far as I know) of optimality in modelling. Examples can be constructed, say along the lines of the system considered in section 6.4 above, where the POD provides misleading results that lead to poor models. As another example, a physical system where a localized disturbance travels back and forth is poorly suited to analysis using the POD; and while such system might in fact have a useful low-dimensional description, the POD may fail to find it because that description does not match the form of eq. (1).

The previous warnings notwithstanding, through a combination of engineering judgement and luck, the POD continues to be fruitfully applied in a variety of engineering and scientific fields. It is, in my opinion, a useful tool at least for people who regularly deal with moderate to high-dimensional data.

Appendix – Code for numerical examples

All computations presented in this paper were carried out using the commercial software Matlab. Some of the codes used are provided below.

A.1 From section 5.1

```
x=linspace(0,1,25);
t=linspace(0,2,50);
[X,T]=meshgrid(x,t);
z=exp(-abs((X-.5).*(T-1)))+sin(X.*T);         % generate the data matrix A

subplot(3,2,1)
surf(X,T,z)                                   % plot the surface
axis([0,1,0,2,0.4,2.1])
xlabel('x'), ylabel('t'), zlabel('z'), title('Actual surface')
```

```
[u,s,v]=svd(z);                                  % note: the means are not subtracted
                                                 %       from the columns

for k=1:3                                        % for rank 1,2 and 3 approximations
        zz=u(:,1:k)*s(1:k,1:k)*v(:,1:k)';        % compare with Eq. 5, and following text
        subplot(3,2,k+1)
        surf(X,T,zz),axis([0,1,0,2,0.4,2.1])
        xlabel('x'), ylabel('t'), zlabel('z')
        title(['Rank ',num2str(k),' approximation'])
end


subplot(3,2,5)                                   % plot the singular values, semilog scale
s=diag(s); semilogy(s,'o')
xlabel('number'), ylabel('singular value'), title('Singular values of z')

subplot(3,2,6)
v=v(:,1:3);                      % drop all but first 3 columns of V
c=v'*z';                         % note, this is the transpose of z*v, which
                                 %       corresponds to A*v in the text

plot(t,c(1,:),'--',t,c(2,:),':',t,c(3,:),'-.')  % plot modal coordinates
xlabel('t'), ylabel('modal coordinates')
title('The modal contributions')
legend('1','2','3')
```

A.2 From section 5.2

```
load vibdata                            % data file containing time t, and y=[x,xdot]
x=y(:,1:10);

subplot(2,2,1)
plot(t,x(:,4))                          % plot displacement of mass 4
xlabel('time'),ylabel('displacement')
title('(a) displacement of mass 4')
axis([0,500,-3,17])

subplot(2,2,2)                          % start another displacement plot

plot(t(4000:5500),x(4000:5500,2),'-.',t(4000:5500),x(4000:5500,5),'-')
xlabel('time'),ylabel('displacement')
title('(b) dash-dot: mass 2; solid line: mass 5')
axis([340,455,-3,20])

for k=1:10
        x(:,k)=x(:,k)-mean(x(:,k));     % mean subtraction
end


[u,s,v]=svd(x(2001:6000,:),0);          % the second argument (i.e., 0) is used for
                                        % far-from-square matrices to save time and memory

                                        % note: first 2000 points are dropped (transients)
s=diag(s);
```

```
subplot(2,2,3)
semilogy(s,'o')                          % plot singular values

xlabel('singular value number'),ylabel('singular value')
title('(c) singular values of mean-subtracted data')
axis([0,11,1e-2,1e4])

subplot(2,2,4)                           % plot proper orthogonal modes

plot(1:10,v(:,1),'o-',1:10,v(:,2),'*-',1:10,v(:,3),'<-')
xlabel('location (mass number)'),ylabel('first 3 proper orthogonal modes')
title('(d) modes 1 (circles), 2 (stars) and 3 (triangles)')
axis([0,11,-0.7,0.7])
```

Notes

1. We assume these functions are bounded and integrable. In experiments, measurements are bounded and discrete; integration is equivalent to summation; and the subtleties of integration can safely be ignored. Here, I stay away from integration theory. The interested reader may consult, e.g. Rudin[7].
2. Strictly speaking, one should say 'rank at most k'.
3. If $m > N$, then $r = N$; there are m eigenvalues but only N singular values; the largest N eigenvalues equal the squares of the N singular values; and the smallest $m - N$ eigenvalues are all exactly zero. If $m \leq N$, then $r = m$; and the m eigenvalues equal the squares of the m singular values. (Readers may wish to work out the details of these calculations using a numerical example of their choice.)

1. Kosambi, D. D., *J. Indian Math. Soc.*, 1943, **7**, 76–88.
2. Holmes, P., Lumley, J. L. and Berkooz, G., *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*, Cambridge Monogr. Mech., Cambridge University Press, 1996.
3. Cusumano, J. P., Sharkady, M. T. and Kimble, B. W., *Philos. Trans. R. Soc. London*, Ser. A, 1994, **347**, 421–438.
4. Feeny, B. F. and Kappagantu, R., *J. Sound Vibr.*, 1998, **211**, 607–616.
5. Koditschek, D., Schwind, W., Garcia, M. and Full, R., 1999 (manuscript under preparation).
6. Ruotolo, R. and Surace, C., *J. Sound Vibr.*, 1999, **226**, 425–439.
7. Rudin, W., *Principles of Mathematical Analysis*, International Series in Pure and Applied Mathematics, McGraw-Hill, 1976, 3rd edn.
8. Golub, G. H. and Van Loan, C. F., *Matrix Computations*, Johns Hopkins University Press, Baltimore, 1990, 2nd edn.
9. Porter, D. and Stirling, D. S. G., *Integral Equations: A Practical Treatment, From Spectral Theory to Applications*, Cambridge Texts in Applied Mathematics, Cambridge University Press, 1990.
10. Ott, E., *Chaos in Dynamical Systems*, Cambridge University Press, 1993.
11. Ravindra, B., *J. Sound Vibr.*, 1999, **219**, 189–192.