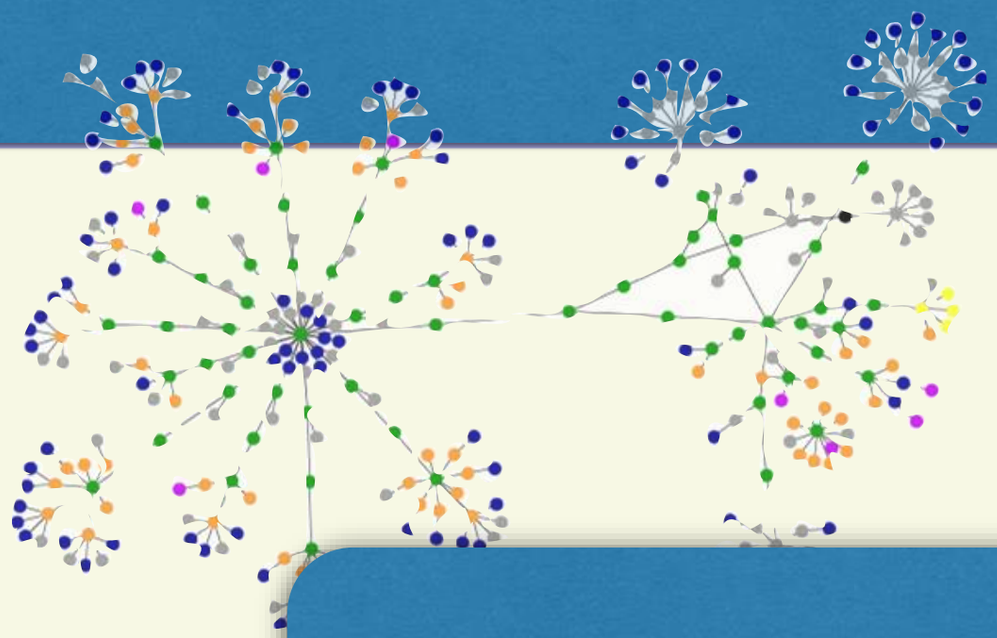
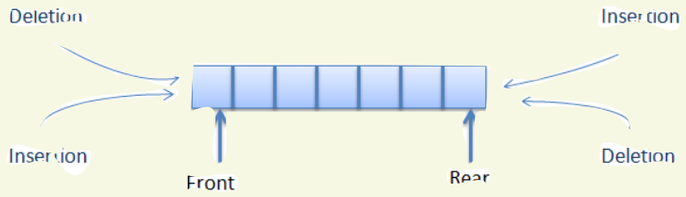
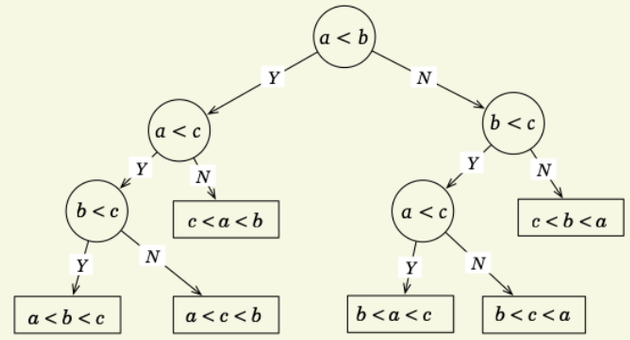






COURSE
“Técnicas Matemáticas para Big Data”
University of Aveiro

Algorithm 2.1: INSERTION-SORT(A)

```
1  for j ← 2 to A.size do
2      key ← A[j]
       // Insert A[j] into the sorted sequence A[1 . . j − 1
3      i ← j − 1
4      while i > 0 and A[i] > key do
5          A[i + 1] ← A[i]
6          i−−
7      A[i + 1] ← key
```

- Introduction to the math Dimension problem
- Standard PCA and SVD
- Variants of PCA (e.g. Kernel PCA)
- Kernel Trick
- Embedding in a Feature Space
- Practical notebook

The Curses and Blessings of Dimensionality

In traditional statistical data analysis, we think of observations of instances of a phenomena, these observations being a vector of values measured on several variables (e.g. blood pressure, weight, height, ...). **It is assumed many observations and a few, more or less well-chosen variables.** However, the trend today is towards more observations but even more so, to radically larger numbers of variables – voracious, automatic, systematic collection of hyper-informative detail about each observed instance. We are seeing examples where the **observations gathered on individual instances are curves, or spectra, or images, or even movies**, so that a single observation has dimensions in the thousands or billions, while there are only tens or hundreds of instances available for study.

The **curse of dimensionality** is a phrase used by several subfields in the mathematical sciences for:

(i) apparent intractability of systematically searching through a high-dimensional space;

(ii) apparent intractability of accurately approximating a general high-dimensional function;

(iii) the apparent intractability of integrating a high-dimensional function.

The **blessings of dimensionality** are less widely noted, but they include:

(i) concentration of measure phenomenon;

(ii) random fluctuations are very well controlled in high dimensions..

Volume of a d-dimension unitary sphere

Consider the unit sphere is d dimensions. Its volume is given by

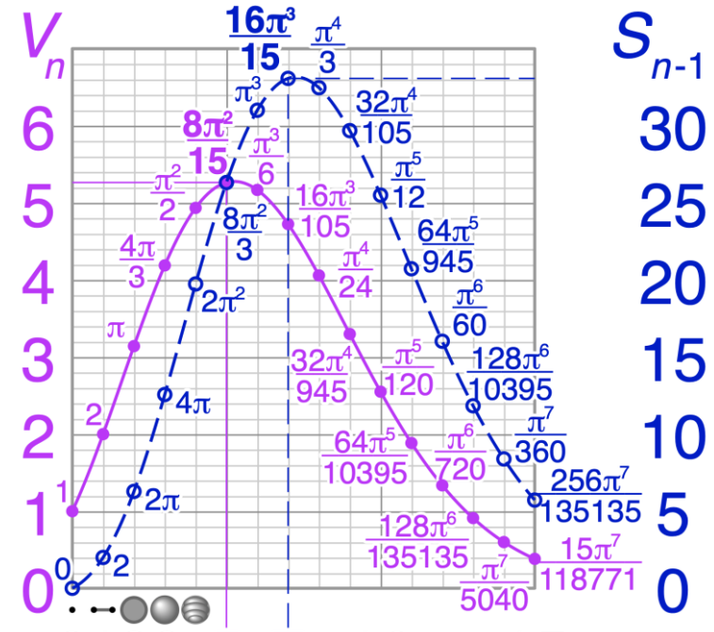
$$V(d) = \frac{\pi^{\frac{d}{2}}}{\frac{d}{2}\,\Gamma\left(\frac{d}{2}\right)}$$

where Γ is the Gamma function. Recall that for positive integers n, $\Gamma(n) = (n-1)!$. Using Stirling's Formula,

$$\Gamma(n) \sim \sqrt{\frac{2\pi}{n}}\left(\frac{n}{e}\right)^n$$

we can see $\Gamma\left(\frac{d}{2}\right)$ grows much faster than $\pi^{\frac{d}{2}}$, and hence

$$V(d) \to 0 \qquad \text{as} \qquad d \to \infty.$$



Volume of a d-dimension unitary cube

Claim: Most of the volume of the high-dimensional cube is located in its corners.

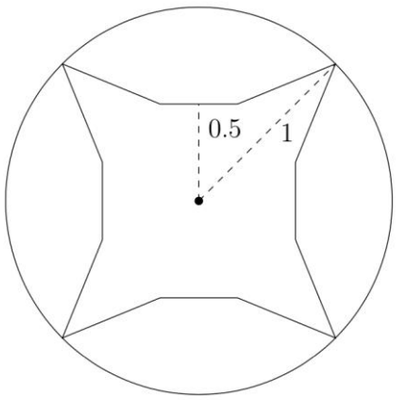


Figure 4: Projections of the 4-dimensional unit sphere and unit cube, centered at the origin (4 of the 16 vertices of the hypercube are shown).

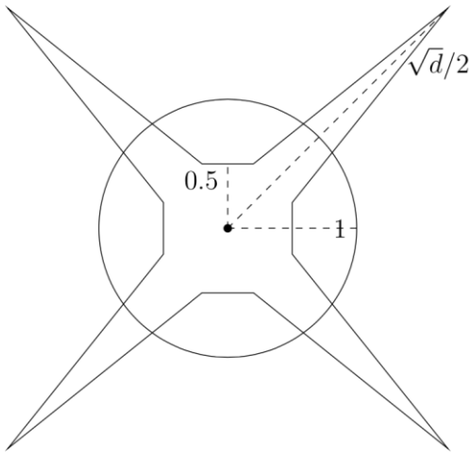


Figure 5: Projections of the d-dimensional unit sphere and unit cube, centered at the origin (4 of the 2^d vertices of the hypercube are shown).

see notes/N01-Higher_Dimensions.pdf

1 Surprises in high dimensions

Our intuition about space is based on two and three dimensions and can often be misleading in high dimensions. It is instructive to analyze the shape and properties of some basic geometric forms, which we understand very well in dimensions two and three, in high dimensions. To that end, we will look at the sphere and the cube as their dimension increases.

1.1 Geometry of the d-dimensional Sphere

Consider the unit sphere is d dimensions. Its volume is given by

$$V(d) = \frac{\pi^{\frac{d}{2}}}{\frac{d}{2}\,\Gamma\left(\frac{d}{2}\right)}$$

where Γ is the Gamma function. Recall that for positive integers n, $\Gamma(n) = (n-1)!$. Using Stirling's Formula,

$$\Gamma(n) \sim \sqrt{\frac{2\pi}{n}}\left(\frac{n}{e}\right)^n$$

we can see $\Gamma\left(\frac{d}{2}\right)$ grows much faster than $\pi^{\frac{d}{2}}$, and hence

$$V(d) \to 0 \qquad \text{as} \qquad d \to \infty.$$

Brief Catalog of Dimension Reduction Algorithms

1. Methods based on Statistics and Information Theory

. . .

2. Methods based on Dictionaries

. . .

3. Methods based on projections

. . .

Brief Catalog of Dimension Reduction Algorithms

1. Methods based on Statistics and Information Theory

1.1. Vector quantisation and mixture models

1.2. Principal component analysis (PCA)

1.3. Singular value decomposition (SVD)

. . .

2. Methods based on Dictionaries

. . .

3. Methods based on projections

. . .

Brief Catalog of Dimension Reduction Algorithms

1. Methods based on Statistics and Information Theory

- **1.1. Vector quantisation and mixture models**
- **1.2. Principal component analysis (PCA)**
- **1.3. Singular value decomposition (SVD)**

Advantages and Problems ?

Q: Is there another linear combination of the current basis that better expresses a dataset?

[Pearson1901] Pearson, K. On Lines and Planes of Closest Fit to Systems of Points in Space. Philosophical Magazine, 1901, 2, 559-572

[Golub1970] Golub, G. H. & Reinsch, C. Singular value decomposition and least squares solutions Numerische Mathematik, 1970, 14, 403-420

Brief Catalog of Dimension Reduction Algorithms

1. Methods based on Statistics and Information Theory

1.1. Vector quantisation and mixture models
1.2. Principal component analysis (PCA)
1.3. Singular value decomposition (SVD)

Advantages and Problems ?

PCA vs Least Squares ?

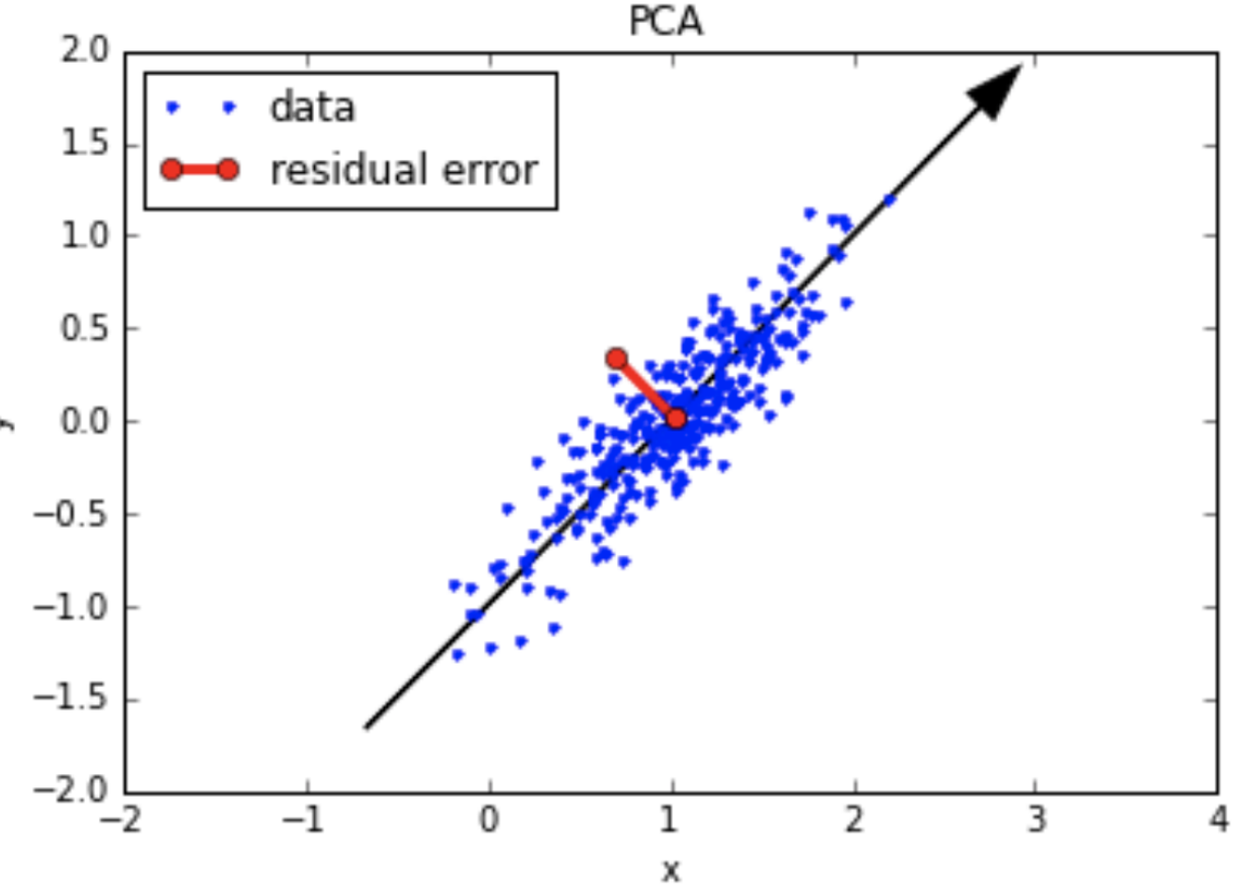
Brief Catalog of Dimension Reduction Algorithms

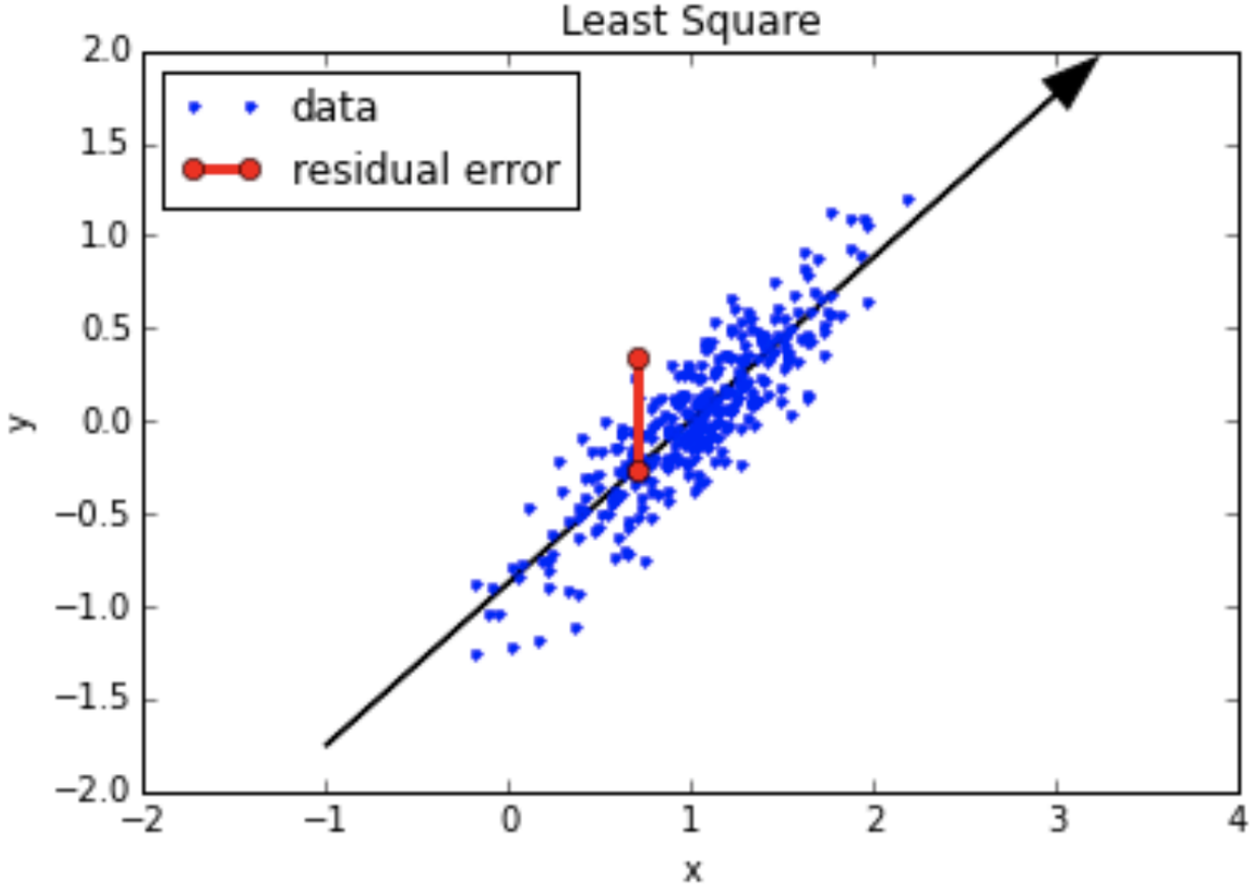
1. Methods based on Statistics and Information Theory

1.1. Vector quantisation and mixture models
1.2. Principal component analysis (PCA)
1.3. Singular value decomposition (SVD)

Advantages and Problems ?

PCA vs Least Squares





Brief Catalog of Dimension Reduction Algorithms

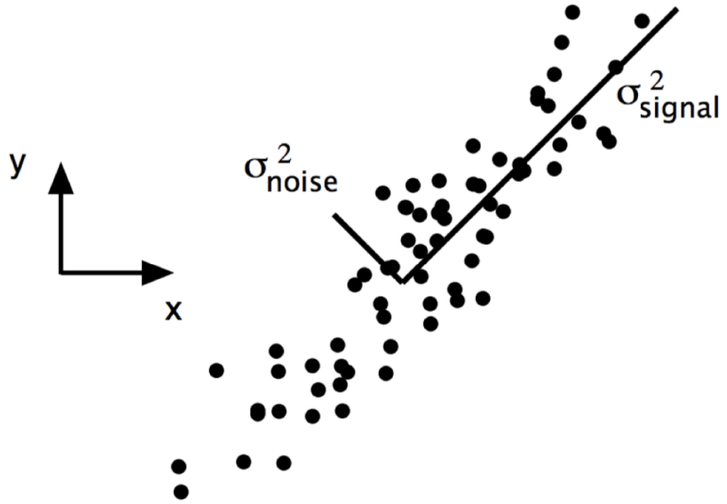
1. Methods based on Statistics and Information Theory

1.1. Vector quantisation and mixture models
1.2. Principal component analysis (PCA)
1.3. Singular value decomposition (SVD)

Advantages and Problems ?

Sign-to-noise ratio vs PCA

$$SNR = \frac{\sigma^2_{signal}}{\sigma^2_{noise}}$$



High SNR: precise measurement

Low SNR: noisy data

We assume interesting dynamics occur on the direction of larger variance (high SNR)

As the motion of the ball is in a straight line, we assume the perpendicular direction to the line of best fit corresponds to noise

Outliers Importance?

see notes/N02-Brief_SVD_PCA.pdf

1 Singular Value Decomposition and Principal Component Analysis

In these lectures we discuss the SVD and the PCA, two of the most widely used tools in machine learning. Principal Component Analysis (PCA) is a linear dimensionality reduction method dating back to Pearson (1901) and it is one of the most useful techniques in exploratory data analysis. It is also known under different names such as the Karhunen-Love Transform, the Hotelling transform, and Proper Orthogonal Decomposition (POD). PCA can be applied to a data set comprising of n vectors $x_1, \ldots, x_n \in \mathbb{R}^d$ and in turn returns a new basis for $\mathbb{R}^d$ whose elements are terms the principal components. It is important that the method is completely data-dependent, that is, the new basis is only a function of the data. The PCA builds on the SVD (or the spectral theorem), we therefore start with the SVD.

1.1 Singular Value Decomposition (SVD)

Consider a matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ or $\mathbb{C}^{m\times n}$ and let us assume that $m \geq n$. Then the *singular value decomposition* (SVD) of $\boldsymbol{A}$ is given by [1]

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{W}^*,$$

where $\boldsymbol{U}$ is $m \times m$, $\boldsymbol{D}$ is $m \times n$, $\boldsymbol{W}$ is $n \times n$, $\boldsymbol{U}$ and $\boldsymbol{W}$ are unitary (i.e., $\boldsymbol{U}^*\boldsymbol{U} = \boldsymbol{U}\boldsymbol{U}^* = \boldsymbol{I}_m$, $\boldsymbol{W}\boldsymbol{W}^* = \boldsymbol{W}^*\boldsymbol{W} = \boldsymbol{I}_n$), and $\boldsymbol{D}$ is a diagonal (rectangular) matrix

which generates the matrix equation $X = U'S'V'^{\mathrm{T}}$, similar to Equation 5.1. Unlike the SVD, however, even though the $\{\mathbf{v}'_k\}$ are an orthonormal basis, the $\{\mathbf{u}'_k\}$ are not in general orthogonal. Nevertheless this demonstrates how the SVD is similar to a Fourier transform, where the vectors $\{\mathbf{v}_k\}$ are determined in a very specific way from the data using Equation 5.1, rather than being given at the outset as for the Fourier transform. Similar to low-pass filtering in Fourier analysis, later we will describe how SVD analysis permits filtering by concentrating on those singular vectors that have the highest singular values.

1.2 Illustrative applications

SVD and PCA have found wide-ranging applications.

Image processing and compression. The property of SVD to provide the closest rank-l approximation for a matrix X (Equation 5.2) can be used in image processing for compression and noise reduction, a very common application of SVD. By setting the small singular values to zero, we can obtain matrix approximations whose rank equals the number of remaining singular values (see Equation 5.2). Each term $\mathbf{u}_k s_k \mathbf{v}_k^{\mathrm{T}}$ is called a *principal image*. Very good approximations can often be obtained using only a small number of terms (Richards, 1993). SVD is applied in similar ways to signal processing problems (Deprettere, 1988).

Immunology. One way to capture global prototypical immune response patterns is to use PCA on data obtained from measuring antigen-specific IgM (dominant antibody in primary immune responses) and IgC (dominant antibody in secondary immune responses) immunoglobulins using ELISA assays. Fesel and Coutinho (Fesel and Coutinho, 1998) measured IgM and IgC responses in Lewis and Fischer rats before and at three time points after immunization with myelin basic protein (MBP) in complete Freud's adjuvant (CFA), which is known to provoke experimental allergic encephalomeyelitis (EAE). They discovered distinct and mutually independent components of IgM reaction repertoires, and identified a small number of strain-specific prototypical regulatory responses.

Molecular dynamics. PCA and SVD analysis methods have been developed for characterizing protein molecular dynamics trajectories (Garcia, 1992; Romo *et al.*, 1995). In a study of myoglobin, Romo *et al*. used molecular dynamics methods to obtain atomic positions of all atoms sampled during the course of a simulation. The higher principal components of the dynamics were found to correspond to large-scale motions of the protein. Visualization of the first three principal components revealed an interesting type of trajectory that was described as resembling beads on a string, and revealed a visibly sparse sampling of the configuration space.

Fesel C., Coutinho A. Dynamics of serum IgM autoreactive repertoires following immunization: strain specificity, inheritance and association with autoimmune disease susceptibility. Eur J Immunol 1998; 28:3616-29.

Garcia A.E. Large-Amplitude Nonlinear Motions in Proteins. Phys Rev Lett 1992; 68:2696-99.

Romo T.D., Clarage J.B., Sorensen D.C., Phillips G.N., Jr. Automatic identification of discrete substates in proteins: singular value decomposition analysis of time-averaged crystallographic refinements. Proteins 1995; 22:311-21.

Small-angle scattering. SVD has been used to detect and characterize structural intermediates in biomolecular small-angle scattering experiments (Chen *et al.*, 1996). This study provides a good illustration of how SVD can be used to extract biologically meaningful signals from the data. Small-angle scattering data were obtained from partially unfolded solutions of lysozyme, each consisting of a different mix of folded, collapsed and unfolded states. The data for each sample was in the form of intensity values sampled at on the order of 100 different scattering angles. UV spectroscopy was used to determine the relative amounts of folded, collapsed and unfolded lysozyme in each sample. SVD was used in combination with the spectroscopic data to extract a scattering curve for the collapsed state of the lysozyme, a structural intermediate that was not observed in isolation.

Information Retrieval. SVD became very useful in Information Retrieval (IR) to deal with linguistic ambiguity issues. IR works by producing the documents most associated with a set of keywords in a query. Keywords, however, necessarily contain much synonymy (several keywords refer to the same concept) and polysemy (the same keyword can refer to several concepts). For instance, if the query keyword is "feline", traditional IR methods will not retrieve documents using the word "cat" – a problem of synonymy. Likewise, if the query keyword is "java", documents on the topic of Java as a computer language, Java as an Island in Indonesia, and Java

Chen L., Hodgson K.O., Doniach S. A lysozyme folding intermediate revealed by solution X-ray scattering. J Mol Biol 1996; 261:658-71.

documents on the topic of Java as a computer language, Java as an Island in Indonesia, and Java as a coffee bean will all be retrieved – a problem of polysemy. A technique known *Latent Semantic Indexing* (LSI) (Berry *et al.*, 1995) addresses these problems by calculating the best rank-*l* approximation of the keyword-document matrix using its SVD. This produces a lower dimensional space of singular vectors that are called *eigen-keywords* and *eigen-documents*. Each eigen-keyword can be associated with several keywords as well as particular senses of keywords. In the synonymy example above, "cat" and "feline" would therefore be strongly correlated with the same eigen-keyterm. Similarly, documents using "Java" as a computer language tend to use many of the same keywords, but not many of the keywords used by documents describing "Java" as coffee or Indonesia. Thus, in the space of singular vectors, each of these senses of "java" is associated with distinct eigen-keywords.

Berry M.W., Dumais S.T., Obrien G.W. Using linear algebra for intelligent information-retrieval. Siam Review 1995; 37:573-95.

Brief Catalog of Dimension Reduction Algorithms

1. Methods based on Statistics and Information Theory

- **1.1. Vector quantisation and mixture models**
- **1.2. Principal component analysis (PCA)**
- **1.3. Singular value decomposition (SVD)**

- **1.2a Incremental, stream or online PCA**
- **1.2b Nonlinear PCA**
- **1.2c PCA rotations and Sparse PCA**
- **1.2d Localised PCA and subspace segmentation**
- **1.2e Kernel PCA and multidimensional scaling**
- **1.2f Robust PCA**
- . . .

Brief Idea (Incremental PCA)

Let us assume that we have observed a number of input vectors **x** and we have already performed their dimensionality reduction with PCA. Let us assume that we are given new observations and we want to refine our directions $\mathbf{w}_i$ to accommodate the new vectors. In the standard PCA approach we would have to re-estimate the covariance matrix of **x** (now using all the vectors, old and new), and to **recompute its eigenvalue decomposition**. Incremental methods (such as Incremental PCA [Artac2002]) provide clever ways of updating our estimates of the best directions $\mathbf{w}_i$ based on the old estimates of the best directions and the new data available. In this way, we can efficiently process new data as they arrive. For this reason, incremental methods are also known as stream methods or online methods. This idea can be applied to many of the methods discussed in this review and will not be further commented.

---

1. [Artac2002] Artac, M.; Jogan, M. & Leonardis, A. Incremental PCA or On-Line Visual Learning and Recognition Proc.16th International Conference on Pattern Recognition (ICPR), 2002, 3, 30781

Brief Catalog of Dimension Reduction Algorithms

1. Methods based on Statistics and Information Theory

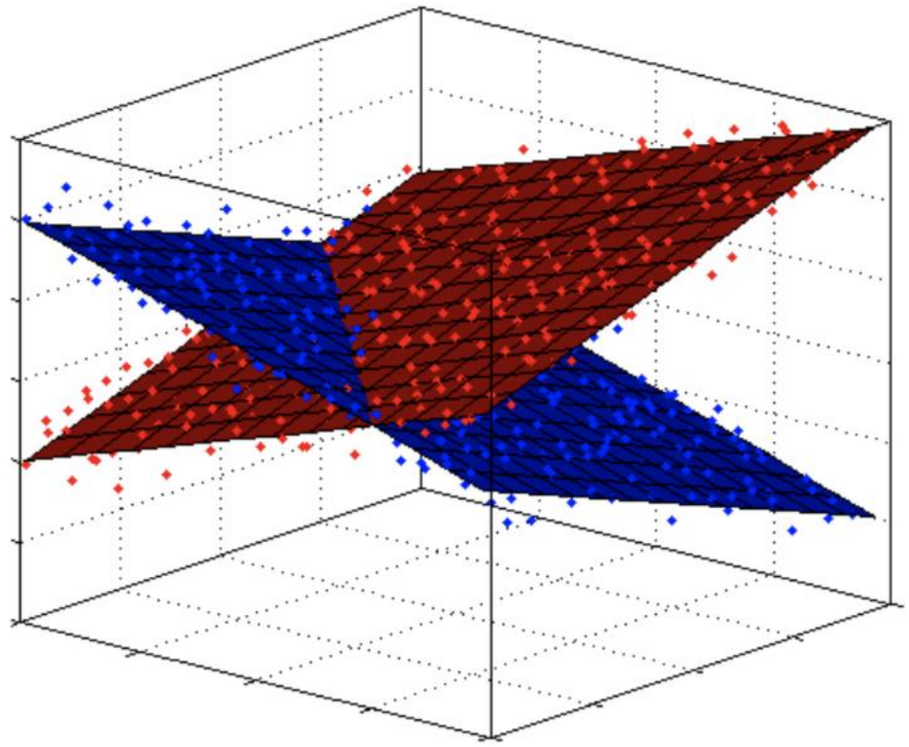
- 1.1. Vector quantisation and mixture models
- 1.2. Principal component analysis (PCA)
- 1.3. Singular value decomposition (SVD)

→

- 1.2a Incremental, stream or online PCA
- 1.2b Nonlinear PCA
- 1.2c PCA rotations and Sparse PCA
- 1.2d **Localised PCA** and subspace segmentation
- 1.2e Kernel PCA and multidimensional scaling
- 1.2f Robust PCA
- . . .

Brief Idea (Localised PCA)

A single linear model has severe limitations for modelling real data. Because the data can be heterogeneous and contain multiple subsets each of which is best fitted with a different linear model. Suppose that we have a high dimensional dataset, that can be conveniently decomposed into different small datasets or clusters, which can be approximated well by different linear subspaces of small dimension by means of a principal component analysis.



Brief Catalog of Dimension Reduction Algorithms

1. Methods based on Statistics and Information Theory

- 1.1. Vector quantisation and mixture models
- 1.2. Principal component analysis (PCA)
- 1.3. Singular value decomposition (SVD)

→

- 1.2a Incremental, stream or online PCA
- 1.2b Nonlinear PCA
- 1.2c PCA rotations and Sparse PCA
- 1.2d Localised PCA and subspace segmentation
- 1.2e **Kernel PCA** and multidimensional scaling
- 1.2f Robust PCA
- . . .

Brief Idea (Kernel PCA)

Some authors [Girolami2002] have proposed to use a non-linear embedding of the input vectors $\mathbf{\Phi(x)}$ into a higher dimensional space (dimensionality expansion, instead of reduction), and then perform the vector quantization in this higher dimensional space (this kind of algorithms are called Kernel algorithms and are further explained below with Kernel PCA). The reason for performing this non-linear mapping is that the topological spherical balls induced by the distance $\|\mathbf{\Phi(x)} - \mathbf{\Phi(x}_k\mathbf{)}^2\|$ in the higher-dimensional space, correspond to non-spherical neighborhoods in the original input space, thus allowing for a richer family of distance functions.

1. [Girolami2002] Girolami, M. Mercer kernel-based clustering in feature space. IEEE Trans. Neural Networks, 2002, 13, 780-784

The Kernel Trick

Given **any algorithm** that only requires dot products, we can construct different nonlinear versions of it, by setting different kernel functions

$$k(x, x') = \langle \Phi(x), \Phi(x') \rangle$$

PCA case (i.e. Kernel PCA)

Notation: $a_i \in \mathbb{R}^m \quad A \in \mathbb{R}^{n \times m}$

Centered data: $A^T 1 = 0 \quad \Leftrightarrow \sum_i a_i = 0$

1. Find principal components, by computing eigenvectors sorted in the order of decreasing eigenvalue and establish a cutoff;
2. Project test points onto principal components;
3. Use coefficients for visualization, classification, regression, etc.

Find principal components by diagonalizing $C = \frac{1}{n} \sum_i a_i^T a_i$

The Kernel PCA

If we send data to a higher dimensional space:

$$x \mapsto \Phi(x)$$

and have it centered: $\sum_i \Phi(a_i) = 0$

Covariance is: $C = \frac{1}{n} \sum_i \Phi(a_i)^T \Phi(a_i)$

$C = \frac{1}{n} \sum_i \Phi(a_i)^T \Phi(a_i)$ diagonalizable by $Cv = \lambda v$

Finding the principal directions amounts to finding coefficients of

$$v = \sum_i \alpha_i \Phi(a_i)^T$$

We do that by solving an eigenvector problem

$$K\alpha = \tilde{\lambda}\alpha$$

Last step: find the projection of a new point a onto the principal components

$$\langle \Phi(a), v_j \rangle = \sum_i \alpha_{ji} K(x, x_i)$$

Algorithm:

1. Choose your kernel
2. Construct centered kernel matrix $\tilde{K} = JKJ$
3. Solve the eigenvalue problem $K\alpha = \tilde{\lambda}\alpha$
4. For any data point, represent it $y_j = \sum_i \alpha_{ji} K(x, x_i)$

Kernel Function:

Real-valued function of two arguments

When symmetric and non-negative can be interpreted as a similarity measure

It is a way to introduce prior knowledge about the data

Kernel Functions

- Polynomial Kernel: $K(\mathbf{x}, \mathbf{x}') = (1 + \langle \mathbf{x}, \mathbf{x}' \rangle)^k$

- Gaussian Kernel: $k(x, x') = \exp(-1/2(x - x')^T \Sigma^{-1}(x - x'))$

- Cosine Similarity: $k(x, x') = \frac{x^T x'}{\|x\| \|x'\|}$

- other e.g. Mercer Kernels

Kernel PCA Properties

Can give a good reencoding of the data when it lies along a non-linear manifold

Can have n positive eigenvalues (not necessarily dimensionality reduction)

Unlike PCA, some directions in feature space might not have preimages in input space.

Embeddings into Feature Spaces

The expressive power of halfspaces is rather restricted – for example, the following training set is not separable by a halfspace.

Let the domain be the real line; consider the domain points $\{-10, -9, -8, \ldots, 0, 1, \ldots, 9, 10\}$ where the labels are $+1$ for all x such that $|x| > 2$ and -1 otherwise.

To make the class of halfspaces more expressive, we can first map the original instance space into another space (possibly of a higher dimension) and then learn a halfspace in that space. For example, consider the example mentioned previously. Instead of learning a halfspace in the original representation let us first define a mapping $\psi : \mathbb{R} \to \mathbb{R}^2$ as follows:

$$\psi(x) = (x, x^2).$$

We use the term *feature space* to denote the range of ψ. After applying ψ the data can be easily explained using the halfspace $h(x) = \text{sign}(\langle \mathbf{w}, \psi(x) \rangle - b)$, where $\mathbf{w} = (0, 1)$ and $b = 5$.

Embeddings into Feature Spaces

The basic paradigm is as follows:

1. Given some domain set $\mathcal{X}$ and a learning task, choose a mapping $\psi : \mathcal{X} \to \mathcal{F}$, for some *feature space* $\mathcal{F}$, that will usually be $\mathbb{R}^n$ for some n (however, the range of such a mapping can be any *Hilbert space*, including such spaces of infinite dimension, as we will show later).
2. Given a sequence of labeled examples, $S = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$, create the image sequence $\hat{S} = (\psi(\mathbf{x}_1), y_1), \ldots, (\psi(\mathbf{x}_m), y_m)$.
3. Train a linear predictor h over $\hat{S}$.
4. Predict the label of a test point, $\mathbf{x}$, to be $h(\psi(\mathbf{x}))$.

Note that, for every probability distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, we can readily define its image probability distribution $\mathcal{D}^\psi$ over $\mathcal{F} \times \mathcal{Y}$ by setting, for every subset $A \subseteq \mathcal{F} \times \mathcal{Y}$, $\mathcal{D}^\psi(A) = \mathcal{D}(\psi^{-1}(A))$.[1] It follows that for every predictor h over the feature space, $L_{\mathcal{D}^\psi}(h) = L_{\mathcal{D}}(h \circ \psi)$, where $h \circ \psi$ is the composition of h onto ψ.

[1] This is defined for every A such that $\psi^{-1}(A)$ is measurable with respect to $\mathcal{D}$.

~ In groups of three:

- Go to **UCI Machine Learning Repository** and choose a dataset of your will.

- Open *C03_PCAtemplate.ipynb*

- Understand the content of the notebook and try to apply PCA, without scaling, to your dataset. (See scikit learn information on PCA).

- Try to apply PCA, with scaling, to your dataset. (See different scaling methodologies in scikit learn).