Dimensionality reduction
-
Concepts, and applications in scRNA-seq and DNA microscopy

Gergo Bohner

March 2, 2019

Contents

1 The goals of dimensionality reduction

Dimensionality reduction is a widely applied set of methods in various data analysis pipelines. It has three main aims:

- *Describe the data manifold*
  Parametrise a space that we believe the observed data lives in. This space also serves as generalisation of where we expect future data to show up. Furthermore certain parameterisations may facilitate interpretation of features.

- *Reduce the observation noise*
  We may separate the data variance into ”within-manifold” (signal) and ”out-of-manifold” (noise), this is called variance partitioning. Often only the within-manifold signal is used for further processing.

- *Visualise the concepts discovered in the data*
  Most collected data nowadays is very high (100+) dimensional, whereas most humans can only conceptualise a few dimensions at once. We have the responsibility to choose the most effective, yet accurate visualisations of the data to communicate features of the data - which may be even more informative together with information derived otherwise (such as derived clusters or external information).

1.1 Examples of specific goals

Finding pure examples of specific dimensionality reduction applications is difficult, as dimensionality reduction methods are often used in conjunction with other techniques to communicate ideas. In the next few pages I show examples of well-defined, distinct applications of various dimensionality reduction methods, to help clarify the breadth of goals we may think of.

1.1.1 Manifold learning

(e.g. Isomap, LLE)
Estimate within-manifold distances, predict what data is likely in the future, learn about structure embedded within the data

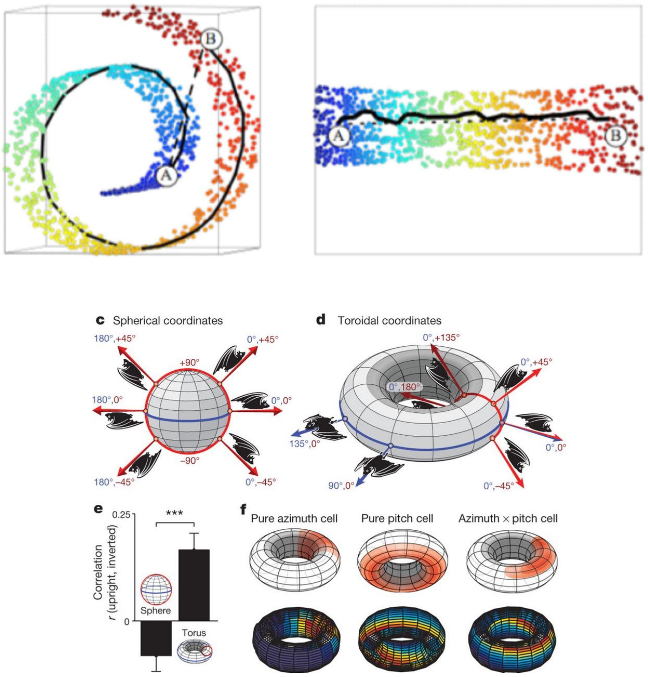


Figure 1: Examples for manifold estimation. (top) Swiss roll artificial dataset shows the concepts of a 2D manifold embedded in 3D (color purely serves as visual aid). (bottom) Comparing different manifold hypotheses (spherical vs toroidal) in behaving bats to explain neural variability *(Finkelstein et al, Nature 2015)*

1.1.2 Feature discovery

(e.g. PCA, ICA, NMF)
Find a meaningful ”basis” for the manifold - a set of features whose combination explains the data well, and leads to investigable hypotheses about the building blocks of the phenomenon that resulted in the collected data.

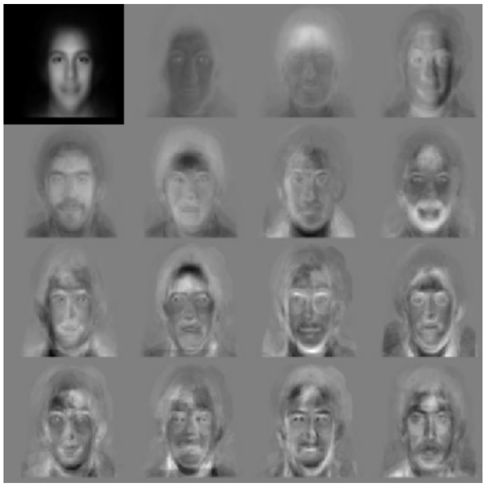


Figure 2: Example feature discovery - Given a corpus of grayscale face portraits, find the ”average” face, and the main directions of variation (in pixel space). Note that even though the principal components are shown as 2D images, in reality the spatial structure is purely due to actual structure in the data, the algorithm was not looking for it (like convolutional neural nets explicitly do).

1.1.3 Reduce observation noise

(e.g. PCA, pPCA, FA, projection on manifold)
Experimenters can assume various sources of noise that is corrupting their input data, and select the appropriate method that is capable of separating signal from noise.

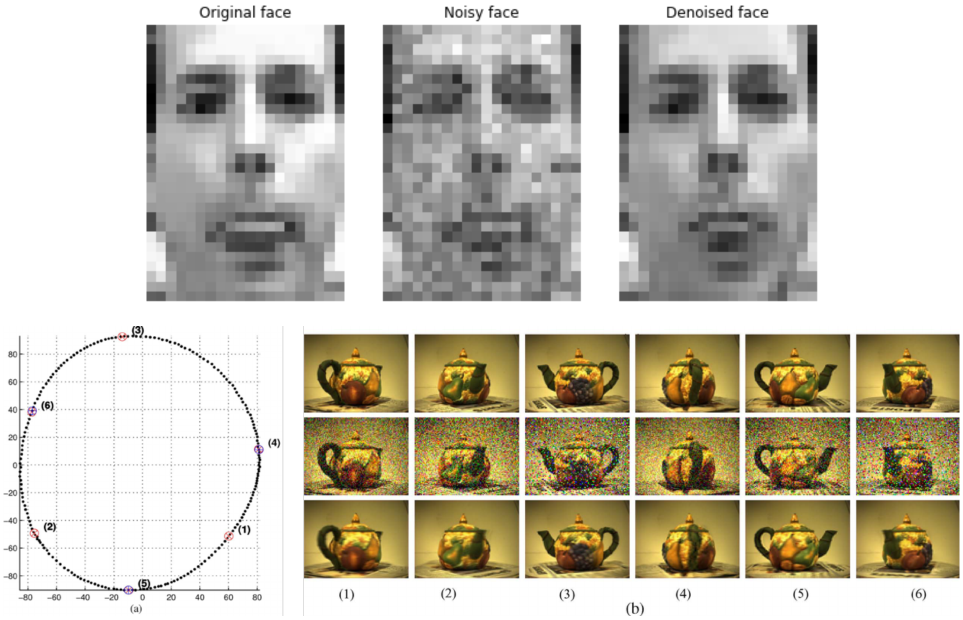


Figure 3: Denoising examples - (top) Using PCA to reduce the effects of additive, uniform and identically distributed Gaussian noise (PCA is optimal for this kind of noise). (bottom) Nonlinear manifold-based associative image denoising. Rotating teapot images embedded in low-D space (a), projection on learned manifold results in noise free images (b). *Huang et al. Manifold-Based Learning and Synthesis* `doi10.1109/TSMCB.2008.2007499`

1.1.4 Visualise high-D data via non-linear embedding

(e.g. LLE, Isomap, t-SNE)
Many dimensionality reduction methods have no easy-to-understand concepts in either a) what features of the high dimensional data they wish to keep unchanged in the low dimensional embedding or b) how they map from the high dimensional space to the low dimensional.

Yet many of them become popular due to their ability to create embeddings in which humans may discover structure in, despite the lack of mathematical guarantees. These methods should be exclusively used for data visualisation, and to only communicate features or concepts of the data that have already been confirmed by other means (such as external information or clustering).

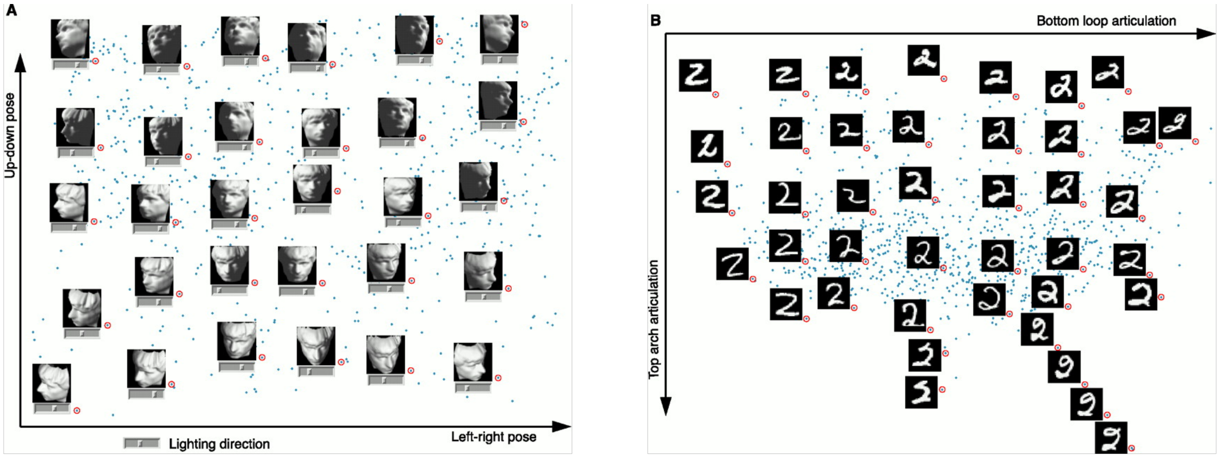


Figure 4: Visualising high-D data via isomap - Datasets can easily be projected into 3D (left) or 2D (right) spaces. Humans can then look at how samples change in the embedding space, and come up with interpretations for the axes. However, these interpretations then need to be treated as hypothesis, translated back into provable mathematical definitions, and shown to be supported by the data. *J. B. Tenenbaum et al. ISOMAP - A Global Geometric Framework for Nonlinear Dimensionality Reduction* `http://web.mit.edu/cocosci/isomap/isomap.html`

1.2 Example of interspersed goals

Most often dimensionality reduction methods will serve multiple of the above goals at once, usually combining visualisation of high-D data, feature discovery and communicating of additional meta-data or concepts discovered via other methods (such as coloring by cluster identity).

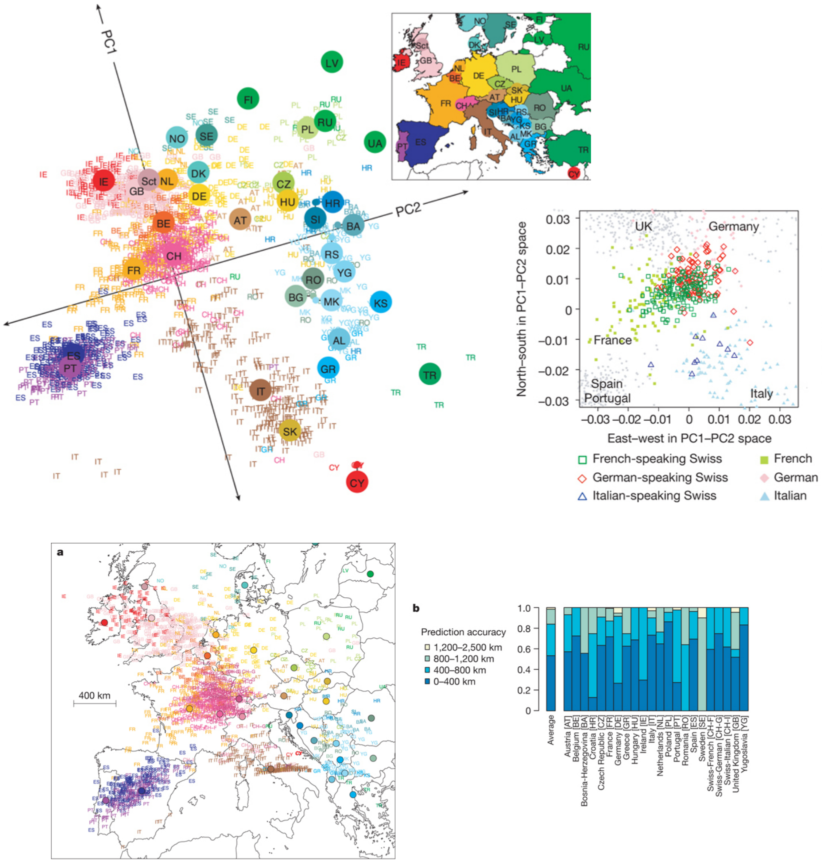
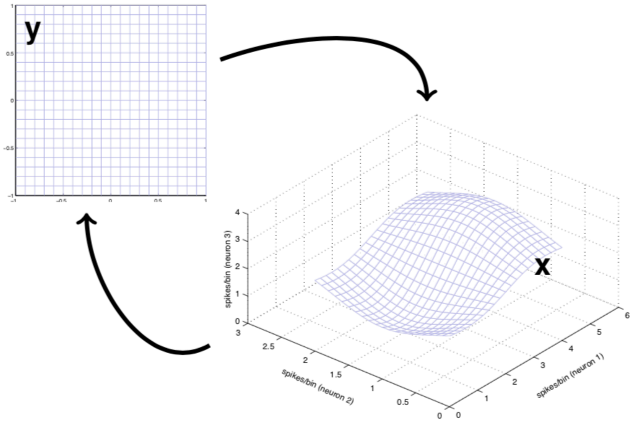


Figure 5: Interspersed goal - Communicating PCA embedding to 2D, cluster identity (from metadata) and geographical matching in the same figure. (top) PCA embedding of genetic sequencing data, in 2D, colored by nationality, cluster center of masses indicated. (top, inset) zoom in on Swiss population, colored by mother-tongue. (bottom) geographical prediction accuracy of sample location, split by country.

2 Mathematical concepts

Goal: Find the manifold.

More precisely, find $\mathbf{y}_i \in \mathbb{R}^{D_y},\ (D_y < D_x)$ so that $\mathbf{y}_i$ parameterises the location of $\mathbf{x}_i$ on the manifold.



Core ideas:

- preserve "local" structure
- preserve "information"

2.1 Linear methods

(PCA, MDS)

Let $\mathbf{y}_i = P^\top \mathbf{y}_i$ for a projection matrix $P^\top$.

$P \in \mathbb{R}^{D_x \times D_y}$ defines a linear mapping from data to manifold, and vice versa.

Linearity

- preserves local structure
- preserves global structure

2.1.1 PCA - Principal component analysis

Idea: look for projection that keeps data as spread out as possible ⇒ most variance, means it preserves most "information" *(Knowing the location of a point along a direction where points are more spread out helps us identify the point more!)*

PCA algorithm

PCA directly implements this idea:

- Find the direction with the greatest variance.
- Project the data onto it (residuals now live in $D_x - 1$ dimensions)
- Repeat until a set maximum embedding dimensions, or when residual variance drops below a given threshold

A collection of data points

$$\{ \mathbf{x}_i \in \mathbb{R}^{D_x \times N} \}_{i=1}^{N}$$

can be represented as a matrix, where each column is a data point

$$[\ \mathbf{x}_1, \ \mathbf{x}_2, \ \cdots , \ \mathbf{x}_N \ ] = \mathbf{X} \in \mathbb{R}^{D_x \times N}$$

PCA then acts as a matrix factorisation on the "scatter matrix", that is the average of outer products, or the scaled empirical covariance matrix

$$\mathbf{S} = \frac{1}{N} \sum_{i=1}^{N} (\mathbf{x}_i - \mathbf{x}_{\text{mean}})(\mathbf{x}_i - \mathbf{x}_{\text{mean}})^\top = \frac{1}{N} \mathbf{X}_c \mathbf{X}_c^\top$$

In general, the principal components are exactly the eigenvectors of the empirical covariance matrix $\mathbf{S}$, ordered by decreasing eigenvalue.

The PCs are the columns of the projection matrix P, and they define a D_y dimensional linear manifold. Each $\mathbf{y}_i$ represent a set of coordinates on the low-dimensional manifold.

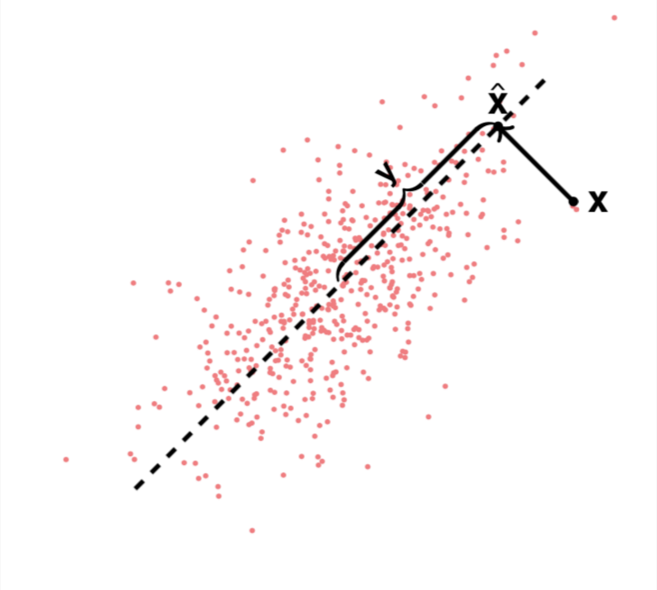
PCA uses #1

We can compute the $\mathbf{y}_i$, the **”coordinates on the embedded manifold”** via the projection matrix:

$$\mathbf{y}_i = P^\top \mathbf{x}_i$$

This emphasizes the use of PCA for the following goals:

- Manifold discovery (find and parametrise a low dimensional manifold to project into)
- Feature discovery (The discovered PCs - the columns of the P projection matrix) are often meaningful, and may be interpreted.
- Visualisation - The high dimensional data may be shown as embedded in low-D, with keeping as much ”information” as possible (PCA is the optimal linear method).



PCA uses #2

But also we can find a **”reconstruction”** of the original point via projecting back from the embedding

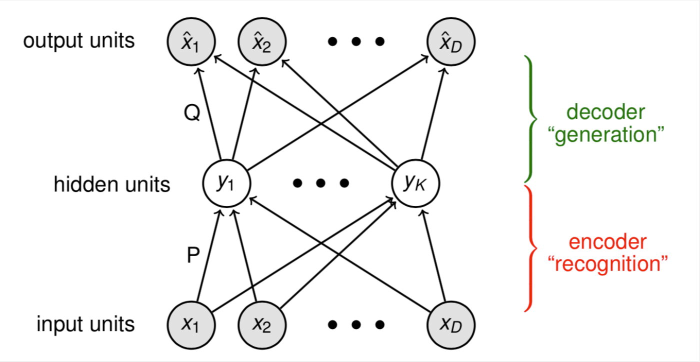
$$\hat{\mathbf{x}}_i = P\mathbf{y}_i,$$

and combining both operations, we can estimate a **”lossy reconstruction”** of a data point $\mathbf{x}_i$:

$$\hat{\mathbf{x}}_i = PP^\top \mathbf{x}_i$$

This gives rise a number of interesting views of PCA (and dimensionality reduction in general), as it can be used for

- Denoising ($\hat{\mathbf{x}}$ is a less noisy version of $\mathbf{x}$)
- Compression (even though we accept $\hat{\mathbf{x}}$ as imperfect, representing only $\mathbf{y}$ and P may take significantly less space, for eg. jpeg format)
- Autoencoding is basically the same as denoising, but we assume the embedding and the reconstrucing projections may be different (even though in PCA they are transposes of one-another). *A linear autoencoder neural network trained to minimise squared error learns to perform PCA (Baldi & Hornik, 1989).*



There are a number of extensions to PCA, that target different issues:

- PCA assumes that noise only affects ”out-of-manifold” dimensions, and all dimensions are weighted equally.
  - pPCA (probabilistic PCA) assumes that noise is isotropic (affects all dimensions equally), and takes it into account when estimates the signal
  - FA (factor analysis) allows for independent noise along each dimension
- The autoencoder view is only optimal if the projections are linear. Non-linear autoencoders (such as multilayer neural networks) are implementing the same idea - to learn compressed, yet effective representations - much more powerfully.

2.1.2 MDS - Multidimensional scaling

Suppose all we were given were distances or symmetric “dissimilarities” Δ_{ij}.

$$\Delta = \begin{bmatrix} 0 & \Delta_{12} & \Delta_{13} & \Delta_{14} \\ \Delta_{12} & 0 & \Delta_{23} & \Delta_{24} \\ \Delta_{13} & \Delta_{23} & 0 & \Delta_{34} \\ \Delta_{14} & \Delta_{24} & \Delta_{34} & 0 \end{bmatrix}$$

Goal: Find vectors $\mathbf{y}_i$ such that $\|\mathbf{y}_i - \mathbf{y}_j\| \approx \Delta_{ij}$.

This is called **Multidimensional Scaling (MDS)**.

Given the some numeric data $\mathbf{X}$, we can compute the Euclidean distances $\Delta_{ij} = \|\mathbf{x}_i - \mathbf{x}_j\|$. The resulting optimal MDS embedding (that now attempts to keep the distances intact, rather than the maximising the variance retained like PCA) is exactly **equivalent to the PCA embedding!** This is called a **”dual representation”**.

MDS advantages

Why would then we ever use MDS then?

Representing data as purely the pairwise dissimilarities between data points is a very powerful idea:

- (nonmetric MDS) We can work on arbitrary, non-numeric datasets as well, as long as we can somehow compute pairwise dissimilarities (see kernel PCA below).
- (metric graph-based MDS) We can replace the Euclidean distances with other distance calculations, that lead to other algorithms:
  - → Isomap uses approximate geodesic distances (distances along an estimated non-linear manifold)
  - → Maximum Variance Unfolding (MVU) preserves only distances amongst k nearest neighbours
  - → Locally linear embedding (LLE) defines non-symmetric distances that represent local linear reconstructability
- (kernel PCA) The distance preservation problem can actually be rewritten in a different form, that attempts to minimise the difference in inner products (Gramian matrices) in the original and embedding spaces:

  $$\arg\min_{\{\mathbf{y}_i\}_{i=1}^{N}} \sum_{i=1}^{N} \sum_{j=1}^{N} (\mathbf{x}_i^\top \mathbf{x}_j - \mathbf{y}_i^\top \mathbf{y}_j)$$

  A popular non-linear extension is to replace the inner products $\mathbf{x}_i^\top \mathbf{x}_j$ with a positive definite kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$, and solve for $\mathbf{y}$ vectors that preserve these modified set of inner products.

  The popular metric kernel choices (Gaussian or Polynomial) are often not well suited for dimensionality reduction. The graph-based algorithms can be seen as a special case of kernel PCA with approximate data-dependent graph kernels (that are defined by neighborhood relationships).

  We can also define kernels over non-metric input spaces (such as bag of words for text mining), which lead

2.2 Non-linear methods

2.2.1 Isomap

2.2.2 LLE

2.2.3 t-SNE

3 Application to "Frey Face" dataset

Interactive python code to gain intuition

4 Application to scRNA-seq dataset

Analysis via various dimensionality reduction method and group discussion on results

5 Dimensionality reduction for DNA microscopy outputs

TODO

6 Overview