Unsupervised Dimension Reduction

(Chapter 15)[1]

David Barber

(adapted by Miguel Palomino)

[1] These slides accompany the book *Bayesian Reasoning and Machine Learning*. The book and demos can be downloaded from `www.cs.ucl.ac.uk/staff/D.Barber/brml`. Feedback and corrections are also available on the site. Feel free to adapt these slides for your own purposes, but please include a link the above website.

Principal Components Analysis

Data is often high dimensional. Provided there is some 'structure', data will typically lie close to a much lower dimensional 'manifold'. Here we concentrate on computationally efficient linear dimension reduction techniques.

$$\mathbf{y} = \mathbf{E}\mathbf{x} + \mathbf{c} \quad \text{or} \quad \mathbf{y} = \mathbf{E}\mathbf{x} \ \text{(for centered data)}$$

We express the approximation for (centered) datapoint $\mathbf{x}^n$ as

$$\mathbf{x}^n \approx \sum_{j=1}^{M} y_j^n \mathbf{b}^j = \mathbf{B}\mathbf{y} \equiv \tilde{\mathbf{x}}^n$$

Basis

The $\mathbf{b}^j$ are 'basis' vectors that span the subspace. Collectively we can write $\mathbf{B} = \left[\mathbf{b}^1, \ldots, \mathbf{b}^M\right]$.

Low dimensional coordinates

The y_i^n are the low dimensional co-ordinates of the data.

Minimal square loss approximation

To determine the best lower dimensional representation it is convenient to use the square distance error between $\mathbf{x}$ and its reconstruction $\tilde{\mathbf{x}}$:

$$E(\mathbf{B}, \mathbf{Y}, \mathbf{c}) = \sum_{n=1}^{N} \sum_{i=1}^{D} [x_i^n - \tilde{x}_i^n]^2$$

The optimal basis $\mathbf{B}$ is then obtained from the eigenvectors of the covariance matrix of the centred data:

$$\mathbf{S} = \mathbf{X}\mathbf{X}^\mathsf{T}$$

The computation of the eigendecomposition of a $D \times D$ matrix is $O(D^3)$. When the number N of datapoints is $N << D$, an alternative is to compute the eigendecomposition of $\mathbf{X}^\mathsf{T}\mathbf{X}$ in $O(N^3)$ and use them to obtain those of $\mathbf{S}$.

PCA via Singular value decomposition

Consider the SVD decomposition of the data matrix:

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^{\mathsf{T}}$$

where $\mathbf{U}^{\mathsf{T}}\mathbf{U} = \mathbf{I}_D$ and $\mathbf{V}^{\mathsf{T}}\mathbf{V} = \mathbf{I}_N$ and $\mathbf{D}$ is a diagonal matrix of the (positive) singular values. We assume that the decomposition has ordered the singular values so that the upper left diagonal element of $\mathbf{D}$ contains the largest singular value. The matrix $\mathbf{X}\mathbf{X}^{\mathsf{T}}$ can then be written as

$$\mathbf{X}\mathbf{X}^{\mathsf{T}} = \mathbf{U}\mathbf{D}\mathbf{V}^{\mathsf{T}}\mathbf{V}\mathbf{D}\mathbf{U}^{\mathsf{T}} = \mathbf{U}\mathbf{D}^2\mathbf{U}^{\mathsf{T}}$$

Since $\mathbf{U}\mathbf{D}^2\mathbf{U}^{\mathsf{T}}$ is in the form of an eigen-decomposition, the PCA solution is equivalently given by performing the SVD decomposition of $\mathbf{X}$, for which the eigenvectors are then given by $\mathbf{U}$, and corresponding eigenvalues by the square of the singular values.

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}^{\mathsf{T}} \approx \underbrace{\mathbf{U}_M}_{D\times M}\ \underbrace{\mathbf{D}_M}_{M\times M}\ \underbrace{\mathbf{V}_M^{\mathsf{T}}}_{M\times N} = \underbrace{\mathbf{B}}_{D\times M}\ \underbrace{\mathbf{Y}}_{M\times N}$$

where $\mathbf{U}_M$, $\mathbf{D}_M$, $\mathbf{V}_M$ correspond to taking only the first M singular values of the full matrices.

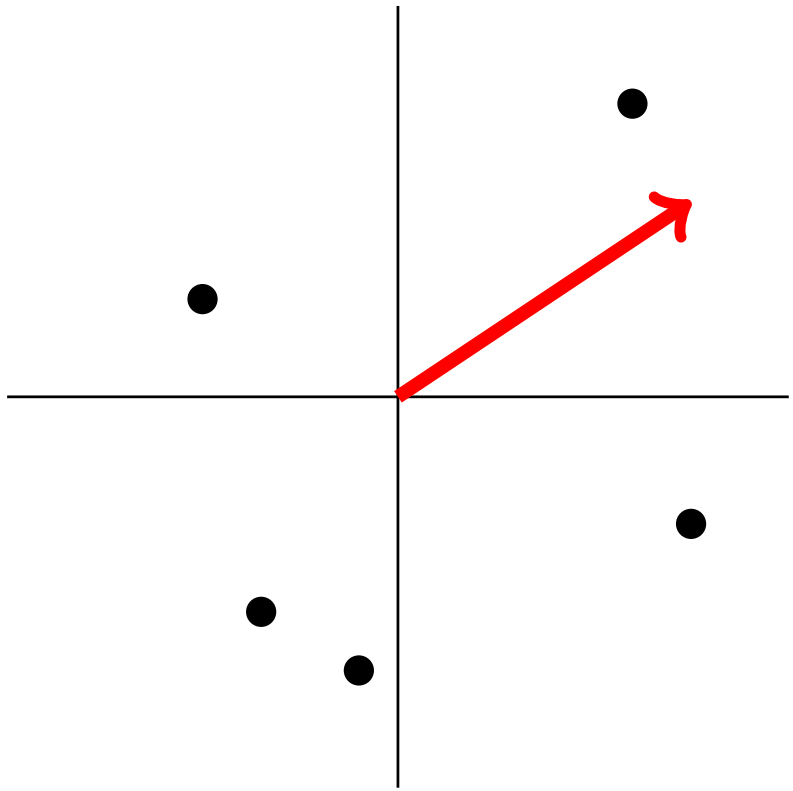
Matrix Decompositions

Given a data matrix $\mathbf{X}$ for which each column represents a datapoint, an approximate matrix decomposition is of the form $\mathbf{X} \approx \mathbf{BY}$ into a basis matrix $\mathbf{B}$ and weight (or coordinate) matrix $\mathbf{Y}$. Symbolically, matrix decompositions are of the form

$$\underbrace{\left(\quad X : \text{Data} \quad\right)}_{D \times N} \approx \underbrace{\left(\quad B : \text{Basis} \quad\right)}_{D \times M} \underbrace{\left(\quad Y : \text{Weights/Components} \quad\right)}_{M \times N}$$

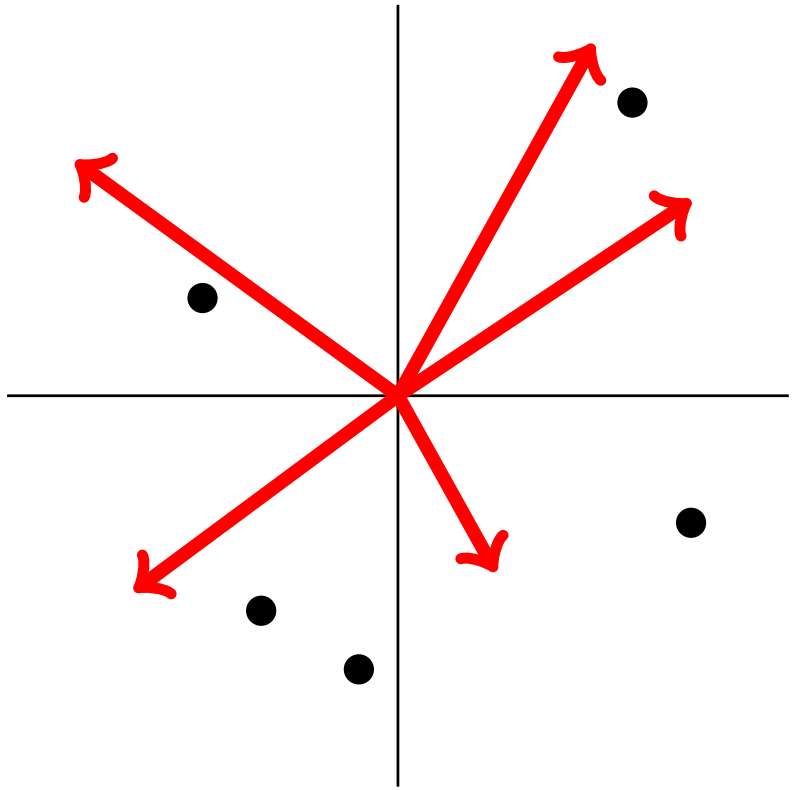
Based on the SVD of the data matrix, we see that PCA is in this class. Many methods can be considered as matrix decompositions under specific constraints.

Under-complete decompositions



When $M < D$, there are fewer basis vectors than dimensions. The matrix $\mathbf{B}$ is then called 'tall' or 'thin'. In this case the matrix $\mathbf{Y}$ forms a lower dimensional approximate representation of the data $\mathbf{X}$, PCA being a classic example.

Over-complete decompositions



For $M > D$ the basis is over-complete, there being more basis vectors than dimensions. In such cases additional constraints are placed on either the basis or components. For example, one might require that only a small number of the large number of available basis vectors is used to form the representation for any given $\mathbf{x}$. Such sparse-representations are common in theoretical neurobiology where issues of energy efficiency, rapidity of processing and robustness are of interest.

Probabilistic latent semantic analysis

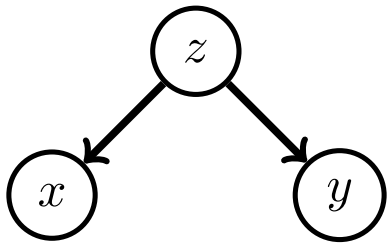
Consider two objects, x and y, where $\mathrm{dom}(x) = \{1, \ldots, I\}$ and $\mathrm{dom}(y) = \{1, \ldots, J\}$ and a dataset $(x^n, y^n, n = 1, \ldots, N)$.

We have a count matrix with elements C_{ij} which describes the number of times the joint state $x = i, y = j$ was observed in the dataset. We can transform this count matrix into a frequency matrix p with elements

$$p(x = i, y = j) = \frac{C_{ij}}{\sum_{ij} C_{ij}}$$

Probabilistic latent semantic analysis (II)

Our interest is to find a decomposition of this frequency matrix of the form



That is,

$$\underbrace{p(x=i,y=j)}_{X_{ij}} \approx \tilde{p}(x=i,y=j)$$

$$= \sum_k \tilde{p}(x=i,y=j,z=k)$$

$$= \sum_k \underbrace{\tilde{p}(x=i|z=k)}_{B_{ik}} \underbrace{\tilde{p}(y=j|z=k)\tilde{p}(z=k)}_{Y_{kj}}$$

where all quantities $\tilde{p}$ are distributions. This is then a form of matrix decomposition into positive basis $\mathbf{B}$ and positive coordinates $\mathbf{Y}$. This has the interpretation of discovering latent topics z that describe the joint behaviour of x and y.

An EM style training algorithm

For probabilities, a useful measure of discrepancy is the Kullback-Leibler divergence

$$\mathrm{KL}(p|\tilde{p}) = \langle \log p \rangle_p - \langle \log \tilde{p} \rangle_p$$

Since p is fixed, minimising the Kullback-Leibler divergence with respect to the approximation $\tilde{p}$ is equivalent to maximising the 'likelihood' term $\langle \log \tilde{p} \rangle_p$. This is

$$L \equiv \sum_{x,y} p(x,y) \log \tilde{p}(x,y)$$

It's convenient to derive an EM style algorithm to learn the parameters; in this case, $\tilde{p}(x|z)$, $\tilde{p}(y|z)$ and $\tilde{p}(z)$.

Consider

$$\mathrm{KL}(q(z|x,y)|\tilde{p}(z|x,y)) = \sum_z q(z|x,y)\log q(z|x,y) - \sum_z q(z|x,y)\log \tilde{p}(z|x,y) \geq 0$$

where $\sum_z$ implies summation over all states of the variable z. Using

$$\tilde{p}(z|x,y) = \frac{\tilde{p}(x,y,z)}{\tilde{p}(x,y)}$$

and rearranging, this gives the bound,

$$\log \tilde{p}(x,y) \geq -\sum_z q(z|x,y)\log q(z|x,y) + \sum_z q(z|x,y)\log \tilde{p}(z,x,y)$$

Plugging this into the 'likelihood' term above, we have the bound

$$L \geq -\sum_{x,y} p(x,y)\sum_z q(z|x,y)\log q(z|x,y) + \sum_{x,y} p(x,y)\sum_z q(z|x,y)\left[\log \tilde{p}(x|z) + \log \tilde{p}(y|z) + \log \tilde{p}(z)\right]$$

M-step

For fixed $\tilde{p}(x|z), \tilde{p}(y|z)$, the contribution to the bound from $\tilde{p}(z)$ is

$$\sum_{x,y} p(x,y) \sum_{z} q(z|x,y) \log \tilde{p}(z)$$

Up to a constant, this is $-\mathrm{KL}\left(\sum_{x,y} q(z|x,y)p(x,y)|\tilde{p}(z)\right)$ so that, optimally,

$$\tilde{p}(z) = \sum_{x,y} q(z|x,y)p(x,y)$$

Similarly, optimally

$$\tilde{p}(x|z) \propto \sum_{y} p(x,y)q(z|x,y)$$

and

$$\tilde{p}(y|z) \propto \sum_{x} p(x,y)q(z|x,y)$$

E-step

The optimal setting for the q distribution at each iteration is

$$q(z|x,y) = \tilde{p}(z|x,y)$$

which is fixed throughout the M-step.

Convergence

L is guaranteed to increase (and the Kullback-Leibler divergence decrease) under iterating between the E and M-steps, since the method is analogous to an EM procedure.

Conditional PLSA

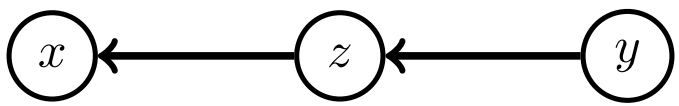
In some cases it is more natural to consider a conditional frequency matrix

$$p(x = i|y = j)$$

and seek an approximate decomposition

$$\underbrace{p(x = i|y = j)}_{X_{ij}} \approx \sum_k \underbrace{\tilde{p}(x = i|z = k)}_{B_{ik}} \underbrace{\tilde{p}(z = k|y = j)}_{Y_{kj}}$$

The model in this case is:

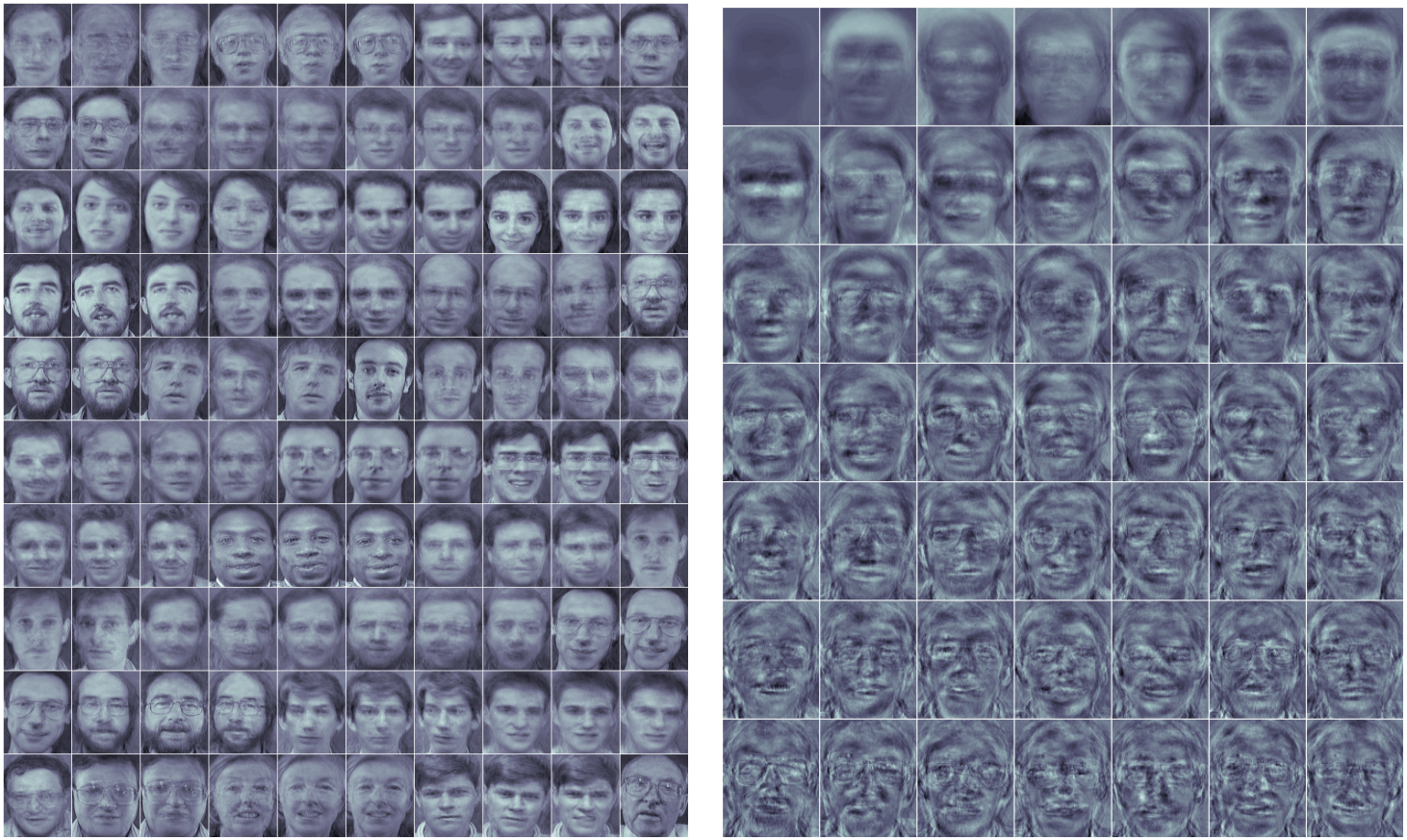


Deriving an EM style algorithm for this is also straightforward.

Eigenfaces

Figure: 100 of the 120 training images (40 people, with 3 images of each person). Each image consists of $92 \times 112 = 10304$ non-negative greyscale pixels. The data is scaled so that, represented as an image, the components of each image sum to 1. The average value of each pixel across all images is 9.70×10^{-5}. This is a subset of the 400 images in the full Olivetti Research Face Database

Eigenfaces



(a) (b)

Figure: **(a)**: SVD reconstruction of the images using a combination of the 49 eigen-images. **(b)**: The eigen-images are found using SVD of the above data and taking the 49 eigenvectors with largest eigenvalue. The images corresponding to the largest eigenvalues are contained in the first row, and the next 7 in the row below, *etc.*

Learning a positive basis

(a) (b) (c) (d)

Figure: **(a)**: Training data, consisting of a positive (convex) combination of the base images. **(b)**: The chosen base images from which the training data is derived. **(c)**: Basis learned using conditional PLSA on the training data. This is virtually indistinguishable from the true basis. **(d)**: Eigenbasis (sometimes called 'eigenfaces').

Positive reconstruction

(a) (b)

Figure: **(a)**: Conditional PLSA reconstruction of the images using a positive convex combination of the 49 positive base images in **(b)**. The root mean square reconstruction error is 1.391×10^{-5}. The base images tend to be more 'localised' than the corresponding eigen-images. Here one sees local structure such as foreheads, chins, *etc.*

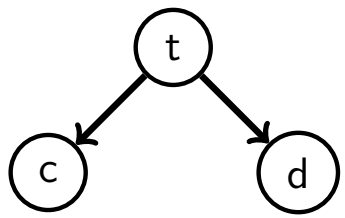
Modelling citations

We have a collection of research documents which cite other documents. For example, document 1 might cite documents $3, 2, 10$, *etc.* Given only the list of citations for each document, can we identify key research papers and the communities that cite them?

A probabilistic formulation

We use the variable $d \in \{1, \ldots, D\}$ to index documents and $c \in \{1, \ldots, D\}$ to index citations (both d and c have the same domain, namely the index of a research article). If document $d = i$ cites article $c = j$ then we set the entry of the matrix $C_{ij} = 1$. If there is no citation, C_{ij} is set to zero. We can form a 'distribution' over documents and citations using

$$p(d = i, c = j) = \frac{C_{ij}}{\sum_{ij} C_{ij}}$$



c: citation
t: topic
d: document

and use PLSA to decompose this matrix into citation-topics.

Modelling citations

The Cora corpus contains an archive of around 30,000 computer science research papers. From this archive the papers in the machine learning category are extracted, consisting of 4220 documents and 38,372 citations.

Using PLSA

The joint PLSA method is fitted to the data using $z = 7$ topics. From the trained model the expression $p(c = j|z = k)$ defines how authoritative paper j is according to community $z = k$.

Modelling citations

factor 1	(Reinforcement Learning)
0.0108	Learning to predict by the methods of temporal differences. Sutton.
0.0066	Neuronlike adaptive elements that can solve difficult learning control problems. Barto et al.
0.0065	Practical Issues in Temporal Difference Learning. Tesauro.
factor 2	(Rule Learning)
0.0038	Explanation-based generalization: a unifying view. Mitchell et al.
0.0037	Learning internal representations by error propagation. Rumelhart et al.
0.0036	Explanation-Based Learning: An Alternative View. DeJong et al.
factor 3	(Neural Networks)
0.0120	Learning internal representations by error propagation. Rumelhart et al.
0.0061	Neural networks and the bias-variance dilemma. Geman et al.
0.0049	The Cascade-Correlation learning architecture. Fahlman et al.
factor 4	(Theory)
0.0093	Classification and Regression Trees. Breiman et al.
0.0066	Learnability and the Vapnik-Chervonenkis dimension. Blumer et al.
0.0055	Learning Quickly when Irrelevant Attributes Abound. Littlestone.
factor 5	(Probabilistic Reasoning)
0.0118	Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Pearl.
0.0094	Maximum likelihood from incomplete data via the em algorithm. Dempster et al.
0.0056	Local computations with probabilities on graphical structures. Lauritzen et al.
factor 6	(Genetic Algorithms)
0.0157	Genetic Algorithms in Search, Optimization, and Machine Learning. Goldberg.
0.0132	Adaptation in Natural and Artificial Systems. Holland.
0.0096	Genetic Programming: On the Programming of Computers by Means of Natural Selection. Koza.
factor 7	(Logic)
0.0063	Efficient induction of logic programs. Muggleton et al.
0.0054	Learning logical definitions from relations. Quinlan.
0.0033	Inductive Logic Programming Techniques and Applications. Lavrac et al.

Table: Highest ranked documents according to $p(c|z)$. The factor topic labels are manual assignments based on similarity to the Cora topics.