Unsupervised Learning

Technische Universität München

Unsupervised vs Supervised

Supervised learning aims to map inputs to targets with $y = f(x)$, or in a probabilistic framework it models $p(y|x)$.

Unsupervised learning can be seen as modelling $p(x)$.

Simple structure in data x:

1. Reduce to latent state z
2. Unsupervised learning is all about compression
3. Learning aims to find generative processing mapping simple distribuition $p(z)$ to $p(x)$

Generative Models

In the following we will consider the following simple graphical model as our generative model:

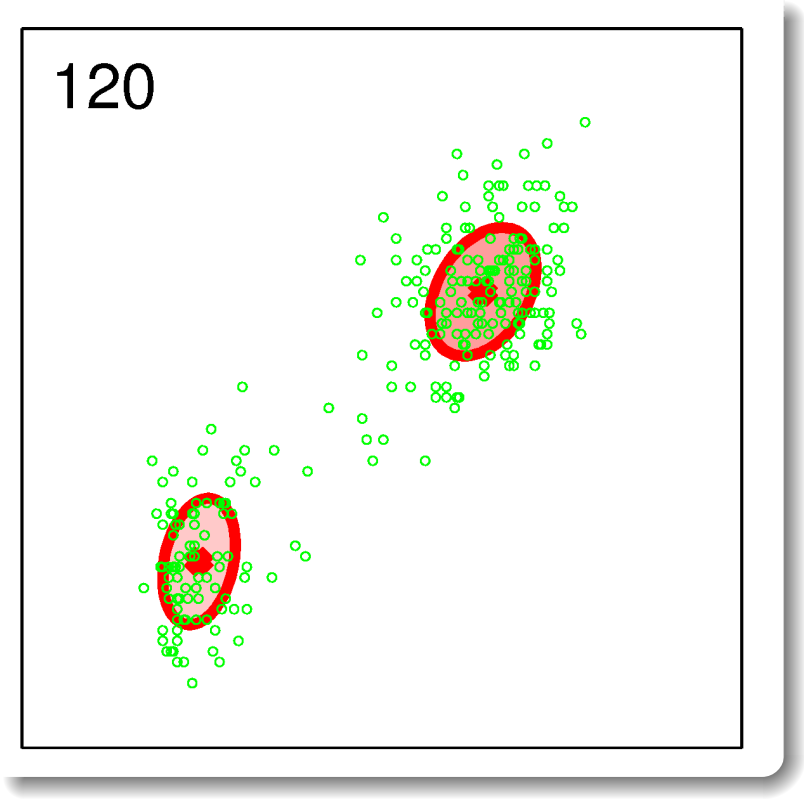
Latent distribution: $p(\boldsymbol{z})$

Generative transformation: $p(\boldsymbol{x}|\boldsymbol{z})$

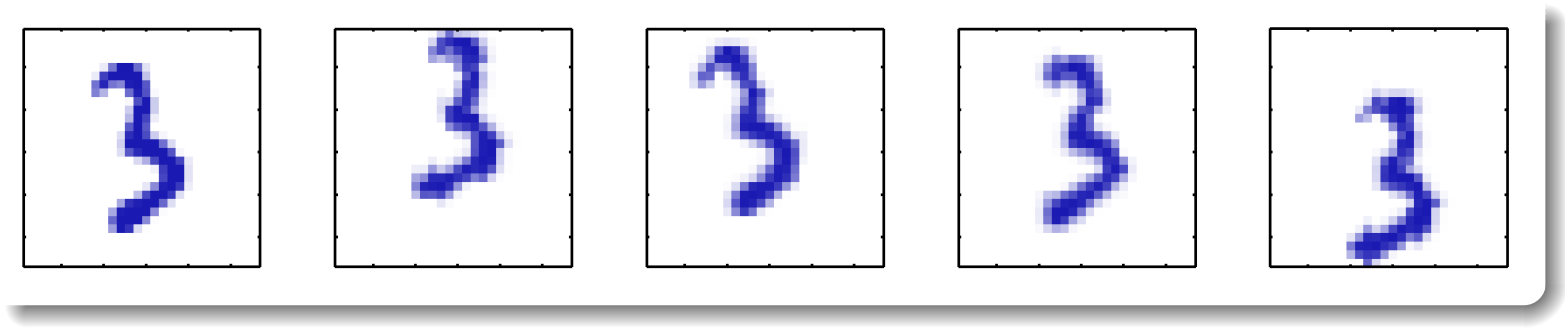
Transforming a simple latent distribution into the observable space. First sampling from $p(\boldsymbol{z})$ and transforming it with $p(\boldsymbol{x}|\boldsymbol{z})$.

Problem: $\boldsymbol{z}$ corresponding to specific $\boldsymbol{x}$ usually unknown

Example: Clustering



1. Class label as latent state
2. Compression of datapoint to single class



Synthetic data obtained by random translations and rotations of one digit image. Resulting Images are 100×100 pixels.

Figure from Bishop's PRML

Intrinsic dimensionality

$10,000$ dimensional data space, but only 3 *degrees of freedom*.

This low dimensionality is inherent to most data-sets.

Data points live on a 3 dimensional (non-linear) subspace.

Translation and rotation parameters are *latent* variables.

Distributed representation (vs. local representation with MoG): all componentes of latent variable are used at the same time to represent the data.

Uses of Models

We are not only interested in generating sample from the observations space.

- Inference of latent $\boldsymbol{z}$ from datapoint $\boldsymbol{x}$
- Density $p(\boldsymbol{x})$
- Correction/impainting with information about $p(\boldsymbol{x})$
- Likelihood for comparing models (only possible for probabilistic models)

Problems:

- Parameters of $p(x)$ unknown
- $p(x)$ multi-modal and complicated
- Inversion of $p(\boldsymbol{x}|\boldsymbol{z})$ unknown

Simple Models

Why simple linear models?

We need to be able to compute $p(x)$ and $p(z|x)$. In some cases this can be done analytically, in other cases iterative learning becomes harder with increasing complexity of the generative model.

Factor Analysis (FA)

$$\boldsymbol{x} \in \mathbb{R}^d, \qquad \boldsymbol{z} \in \mathbb{R}^l, \qquad l \ll d$$

$$p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{0}, \boldsymbol{I}) \tag{1}$$

(*simple* latent distribution without loss of generality!)

$$p(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\boldsymbol{x}|\, \boldsymbol{W}\boldsymbol{z} + \boldsymbol{\mu}, \boldsymbol{\Psi}) \tag{2}$$

$\boldsymbol{W}$: *factor loading* matrix, $\mathbb{R}^{d \times l}$

$\boldsymbol{\Psi}$: diagonal matrix (*uniquenesses*), $\mathbb{R}^{d \times d}$

Matrix Factorization

Likelihood for FA

One of the properties we would like to use:

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})d\boldsymbol{z} = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{C}) \tag{3}$$

with

$$\boldsymbol{C} = \boldsymbol{W}\boldsymbol{W}^T + \boldsymbol{\Psi} \tag{4}$$

why? (see *Multivariate Gaussians*)

or ($\boldsymbol{\psi} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Psi})$):

$$E(\boldsymbol{X}) = E(\boldsymbol{W}\boldsymbol{z} + \boldsymbol{\mu} + \boldsymbol{\psi}) = \boldsymbol{\mu} \tag{5}$$

$$Cov(\boldsymbol{X}) = E((\boldsymbol{W}\boldsymbol{z}+\boldsymbol{\psi})(\boldsymbol{W}\boldsymbol{z}+\boldsymbol{\psi})^T) = E(\boldsymbol{W}\boldsymbol{z}\boldsymbol{z}^T\boldsymbol{W}^T+\boldsymbol{\psi}\boldsymbol{\psi}^T) = \boldsymbol{W}\boldsymbol{W}^T+\boldsymbol{\Psi} \tag{6}$$

Statistically more efficient than a full covariance Gaussian: $O(ld)$ vs. $O(d^2)$.

Likelihood for FA

FA defines a density on $\boldsymbol{X}$. Computing the likelihood for a data point $\boldsymbol{x}$ requires $\boldsymbol{C}^{-1}$ and $|\boldsymbol{C}|$.

$$\boldsymbol{C}^{-1} = \boldsymbol{\Psi}^{-1} - \boldsymbol{\Psi}^{-1}\boldsymbol{W}(\boldsymbol{I}_l + \boldsymbol{W}^T\boldsymbol{\Psi}^{-1}\boldsymbol{W})^{-1}\boldsymbol{W}^T\boldsymbol{\Psi}^{-1} \tag{7}$$

(via Shermann-Morrison-Woodbury formula. $O(l^3)$.)

$$|\boldsymbol{C}| = |\boldsymbol{I}_l + \boldsymbol{W}^T\boldsymbol{\Psi}^{-1}\boldsymbol{W}||\boldsymbol{\Psi}^{-1}| \tag{8}$$

(via matrix determinant lemma. $O(l^3)$.)

Computationally more efficient than a full covariance Gaussian.

Probabilistic PCA (PPCA)

Constrain FA model: $\boldsymbol{\Psi} = \sigma^2 \boldsymbol{I}$ and $\boldsymbol{W}$ is orthonormal.

Likelihood:

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}|\boldsymbol{z}) p(\boldsymbol{z}) d\boldsymbol{z} = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{C}) \tag{9}$$

with

$$\boldsymbol{C} = \boldsymbol{W}\boldsymbol{W}^T + \sigma^2 \boldsymbol{I} \tag{10}$$

Closed form solution for estimating $\boldsymbol{W}$:

$$\boldsymbol{W} = \boldsymbol{L}_l (\boldsymbol{\Lambda}_l - \sigma^2 \boldsymbol{I})^{\frac{1}{2}} \boldsymbol{R} \tag{11}$$

with $\boldsymbol{L}_l$ the $d \times l$ matrix whose columns are the first l eigenvectors of the empirical covariance matrix $\boldsymbol{S}$ and $\boldsymbol{\Lambda}_l$ the diagonal matrix of corresponding eigenvalues ($\boldsymbol{R}$ is an arbitrary orthogonal matrix).

$$\sigma^2 = \frac{1}{d-l} \sum_{j=l+1}^{d} \lambda_j \tag{12}$$

PPCA captures (or models) variance along principal directions:
Standardize data (unit of measurement)!

FA vs. PPCA

Compare generated samples.

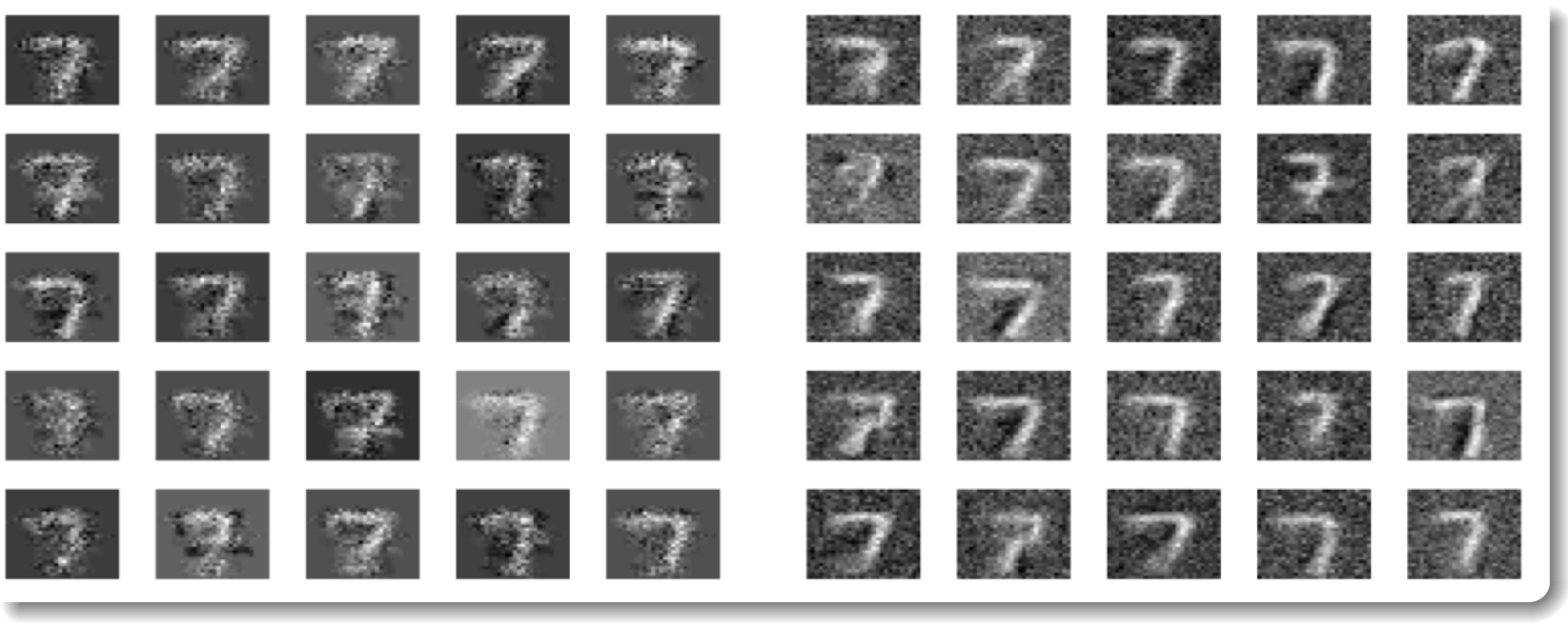


Figure from *Bayesian Reasoning and Machine Learning*, David Barber.

PCA

Closely related to PPCA with variance going towards zero.

Find an orthogonal set of l linear basis functions $\boldsymbol{w}_j \in \mathbb{R}^d$ and corresponding low-dimensional projections $\boldsymbol{z}_j \in \mathbb{R}^l$ such that the average *reconstruction error* is minimized:

$$J = \frac{1}{N} \sum_i ||\boldsymbol{x} - \hat{\boldsymbol{x}}||^2 \tag{13}$$

with $\hat{\boldsymbol{x}}_i = \boldsymbol{W}\boldsymbol{z}_i + \boldsymbol{\mu}$.

Equivalent (non obvious!) formulation: Find directions with *maximal projected variance*.

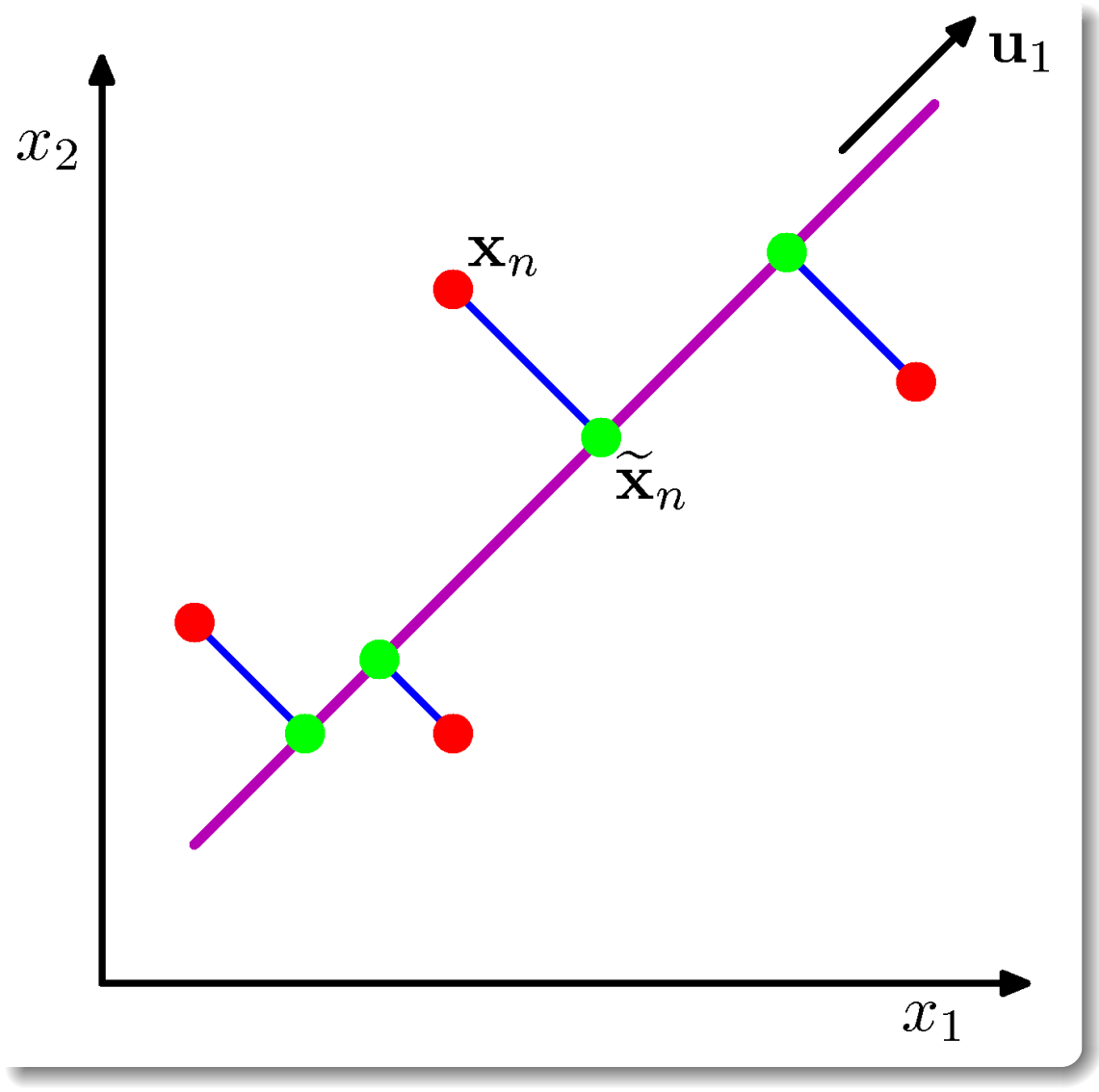


Figure from Bishop's PRML

Biggest variance along purple line is achieved when points show lowest distance to principal component (blue lines).

Solution similar to PPCA:

$$\boldsymbol{W} = \boldsymbol{L}_l \tag{14}$$

$\boldsymbol{L}_l$ is the $d \times l$ matrix whose columns are the first l eigenvectors of $\boldsymbol{S}$.

Note: $\boldsymbol{L}_l^T \boldsymbol{L}_l = \boldsymbol{I}_l$ but not $\boldsymbol{L}_l \boldsymbol{L}_l^T = \boldsymbol{I}$, unless $d = l$.

$$J = \sum_{i=l+1}^{d} \lambda_i \tag{15}$$

What should be the dimensionality of the latent embedding?

Datapoint with first four Eigenvalues.

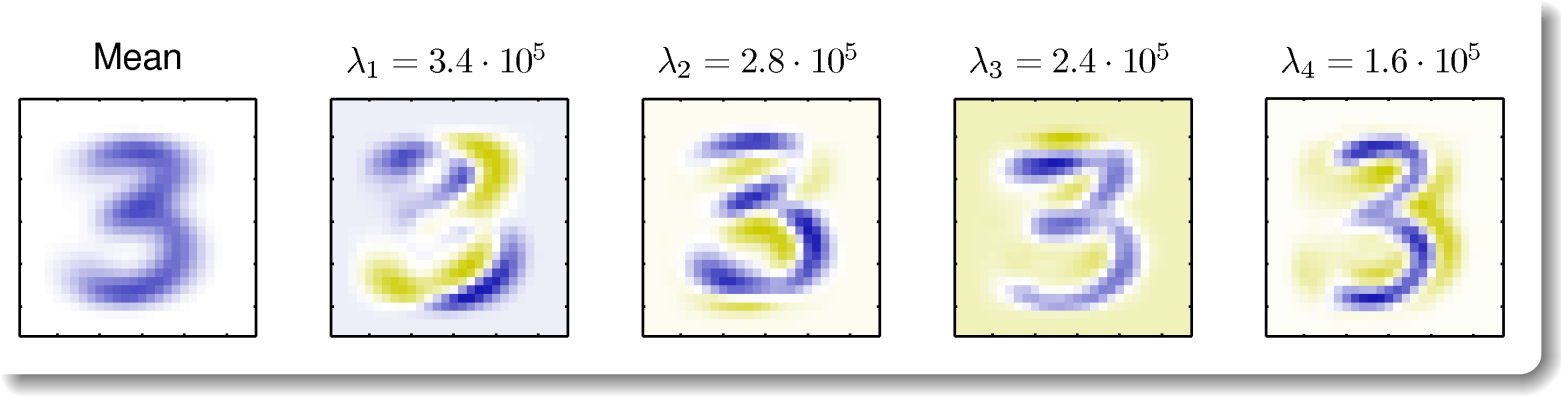


Figure from Bishop's PRML

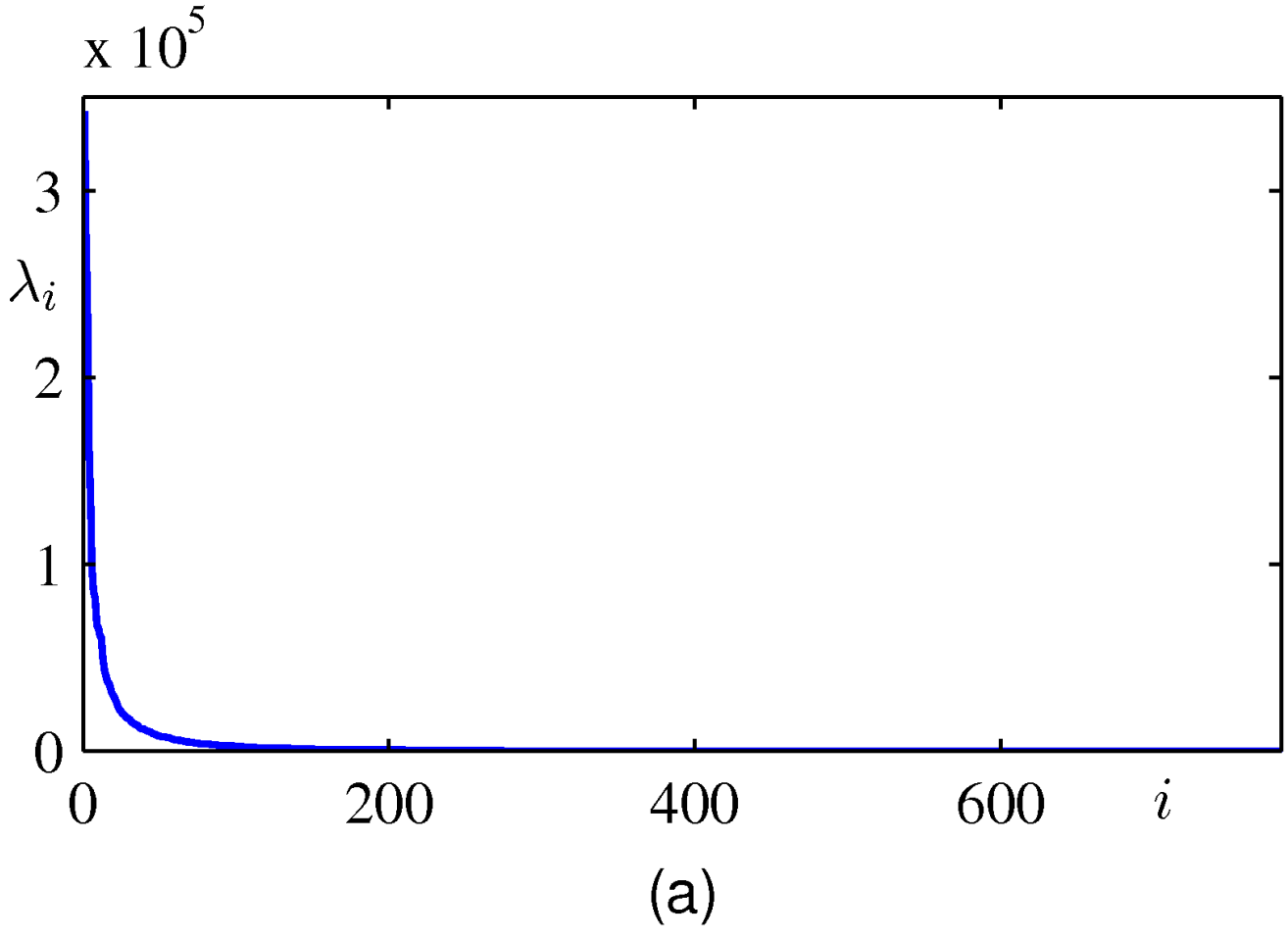


Figure from Bishop's PRML

Eigenfaces

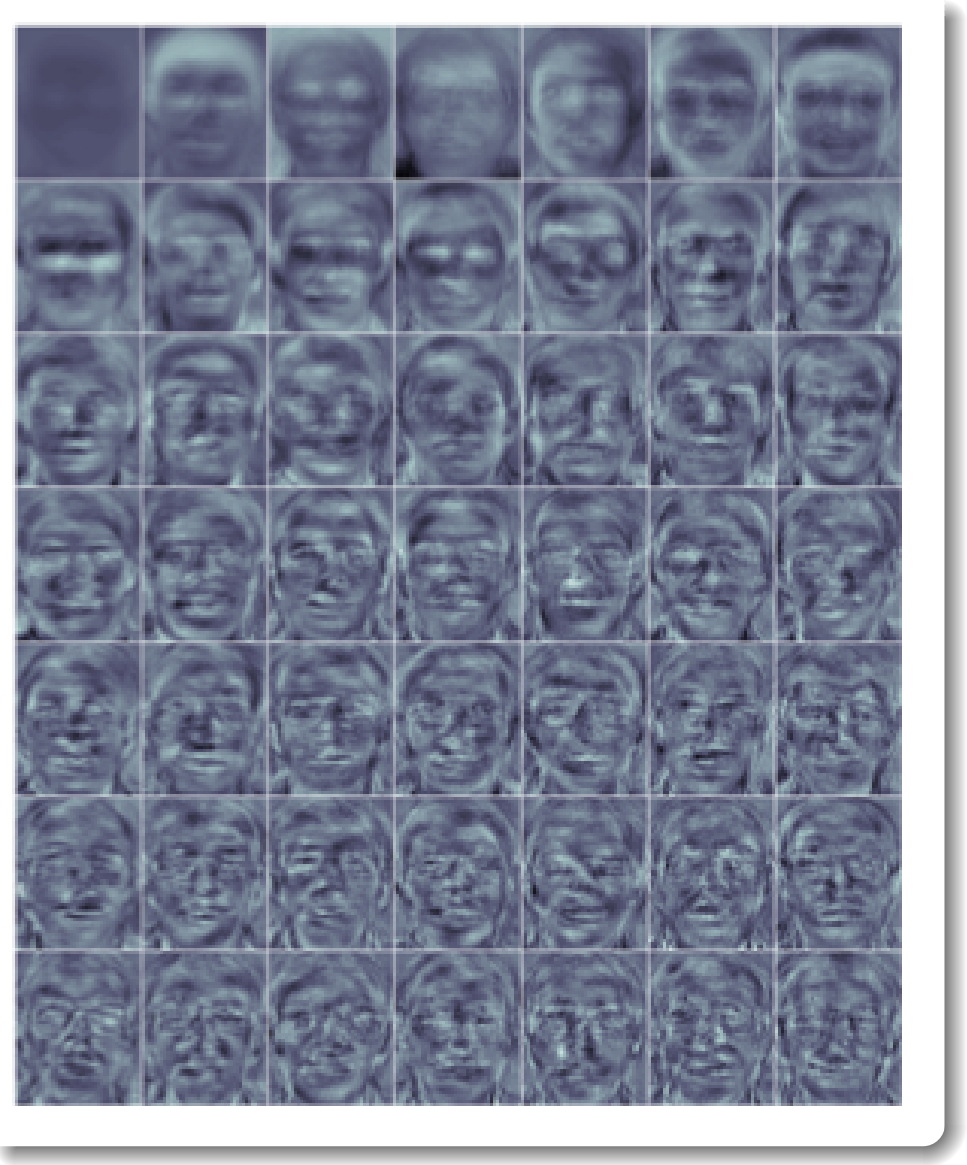


Figure from *Bayesian Reasoning and Machine Learning*, David Barber.

Whitening/ZCA

PCA is often used as a preprocessing step.

$$\boldsymbol{z} = \boldsymbol{\Lambda}^{-\frac{1}{2}} \boldsymbol{L}_l^T (\boldsymbol{x} - \boldsymbol{\mu}) \tag{16}$$

$\boldsymbol{z}$ is *whitened*, that is, the emprical covariance matrix is $\boldsymbol{I}$.

Zero Component Analysis (ZCA) stays in the original domain:

$$\hat{\boldsymbol{x}} = \boldsymbol{L}_d (\boldsymbol{\Lambda} + \varepsilon \boldsymbol{I})^{-\frac{1}{2}} \boldsymbol{L}_d^T (\boldsymbol{x} - \boldsymbol{\mu}) \tag{17}$$

ICA – Independent Component Analysis

Generative model:

- some (unknown) data $\boldsymbol{s} \in R^n$ (n independent *sources*).
- unknown *mixing* matrix $\boldsymbol{A} \in R^{n \times n}$.
- observed $\boldsymbol{x} \in R^n$: $\boldsymbol{x} = \boldsymbol{A}\boldsymbol{s}$

Given m observations $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$, can we recover the $\boldsymbol{s}$'s? Yes, if we would know $\boldsymbol{W} = \boldsymbol{A}^{-1}$.

ICA – Ambiguities

- Permutations
- Scaling
- Sources $\boldsymbol{s}_j$ must be *non-gaussian*.

Distribution of $\boldsymbol{S}$ must be non-gaussian. What does this mean for $\boldsymbol{X}$?

$$p_X(\boldsymbol{x}) = p_{\boldsymbol{S}}(\boldsymbol{W}\boldsymbol{x})|\boldsymbol{W}| = \prod_i p_{\boldsymbol{S}_i}(\boldsymbol{w}_i^T\boldsymbol{x})|\boldsymbol{W}|$$

where $\boldsymbol{w}_i^T$ is the i-th *row* of $\boldsymbol{W}$.

ICA – MLE

We can do MLE on $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ if we specify densities for the individual sources (standard examples would be $p_{\boldsymbol{S}_i}(s) \approx \sigma'(s)$ or $p_{\boldsymbol{S}_i}(s) \approx -\tanh'(y)$).

The loglikelihood then is

$$\ell(\boldsymbol{W}) = \sum_{i=1}^{m} \sum_{j=1}^{n} \log p_{\boldsymbol{S}_i}(\boldsymbol{w}_i^T \boldsymbol{x}) + \log |\boldsymbol{W}|$$

There is no closed form solution because of the involved non-linearities! We could find $\boldsymbol{W}$ by e.g. gradient descent.

$$\nabla_{\boldsymbol{W}} \ell(\boldsymbol{W}) = \sum_{i=1}^{m} \left(\begin{pmatrix} 1 - 2g_1(\boldsymbol{w}_1^T \boldsymbol{x}) \\ 1 - 2g_2(\boldsymbol{w}_2^T \boldsymbol{x}_i) \\ \cdots \\ 1 - 2g_n(\boldsymbol{w}_n^T \boldsymbol{x}_i) \end{pmatrix} \boldsymbol{x}_i^T + \boldsymbol{W}^{-T} \right)$$

g_i is the antiderivative (*Stammfunktion* in german) of $p_{\boldsymbol{S}_i}$.

Non-linear Independent Components Estimation

Inference becomes intractable with complicated generative models.

Just imagine the log determinant of the Jacobian for a deep neural network.

Solution for special network structure:

Non-linear Independent Components Estimation
Laurent Dinh, David Krueger, Yoshua Bengio