Continuous Latent Variables

Mikaela Klami

Adaptive Informatics Research Centre
Laboratory of Computer and Information Science

16.4.2007
T-61.6020 Machine Learning: Basic Principles

Outline

1. Principal Component Analysis
   - Traditional PCA
   - Probabilistic PCA
   - Kernel PCA
2. Nonlinear or non-Gaussian latent variable models
   - Overview
   - Some methods

Continuous latent variables?

- latent variable: unknown variable, one for each data point (in contrast to model parameters of which we only have one set)
- previously (Chapter 9): models with discrete latent variables
- e.g. mixture of Gaussians; the latent variable tells to which cluster a data point belongs to
- here: models where some, or all, latent variables are continuous
- a continuous latent variable may e.g. represent a location on a subspace or a manifold

PCA - Principal Component Analysis

- the PCA finds a linear subspace that passes close to the data
- orthogonal projection of the data onto a lower-dimensional linear space
- widely used for e.g. dimensionality reduction, feature extraction and data visualization
- dates back to long before probabilistic latent variables models, and has later been re-interpreted as one
- two equivalent formulations:
  1. maximum variance formulation
  2. minimum-error formulation

Maximum variance formulation

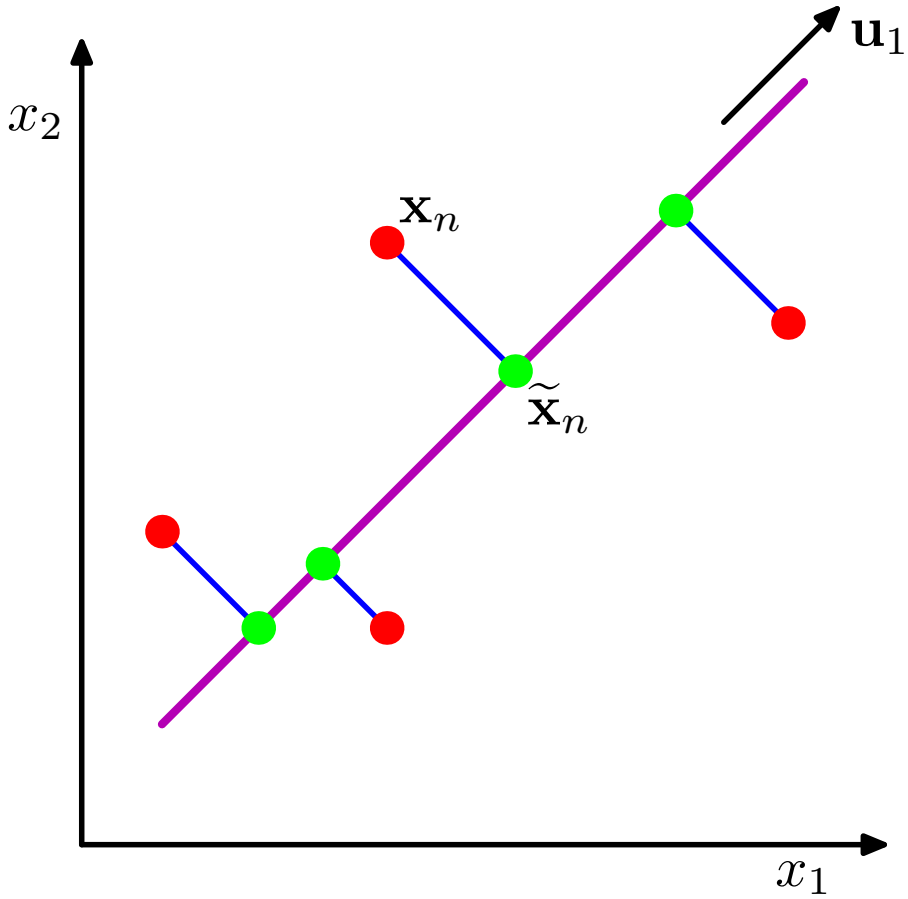
- idea: try to keep as much of the variation in the data as possible
- maximize the variance of the projected data points $\mathbf{u}_1^T \mathbf{x}_n$
- variance expressed as $\mathbf{u}_1^T \mathbf{S} \mathbf{u}_1$, where $\mathbf{S}$ is the covariance matrix of the data
- maximize with restriction $\mathbf{u}_1^T \mathbf{u}_1 = 1$ using Lagrange multipliers
  $\Rightarrow \mathbf{S}\mathbf{u}_1 = \lambda_1 \mathbf{u}_1$ (an eigenvalue problem)
- other components than first restricted to be orthogonal to all previous components
- solving the eigenvalue problem gives all projection dimensions $\mathbf{u}$ at once as eigenvectors of $\mathbf{S}$; the first is the one that corresponds to the largest eigenvalue λ, etc.

Minimum error formulation

- idea: if we replace observations with their projections, the projections should be as close as possible to the observations
- in any orthonormal basis with basis vectors $\mathbf{u}_i$, we can express the data as $\mathbf{x}_n = \sum_{i=1}^{D} \alpha_{ni} \mathbf{u}_i$
- here we approximate the data by a lower-dimensional representation $\tilde{\mathbf{x}}_n$ by keeping only the first M of the coefficients α_{ni} for each data point
- find the basis that minimizes the squared error in making the approximation: $J = \frac{1}{N} \sum_{n=1}^{N} \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2$
- ...a long derivation in the book, again leads to: $\mathbf{S}\mathbf{u}_i = \lambda_i \mathbf{u}_i$ $\Rightarrow$ the formulations lead to the same method

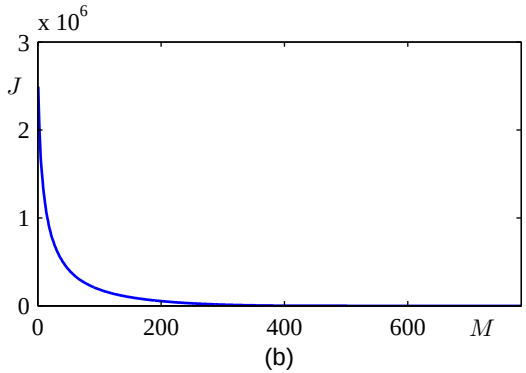
Illustration of the two formulations

- Maximum variance: maximize spread of green dots on the line
- Minimum error: minimize average squared length of blue lines



Applications

- dimensionality reduction: in many applications, a large number of dimensions can be dropped without introducing noticeable projection error

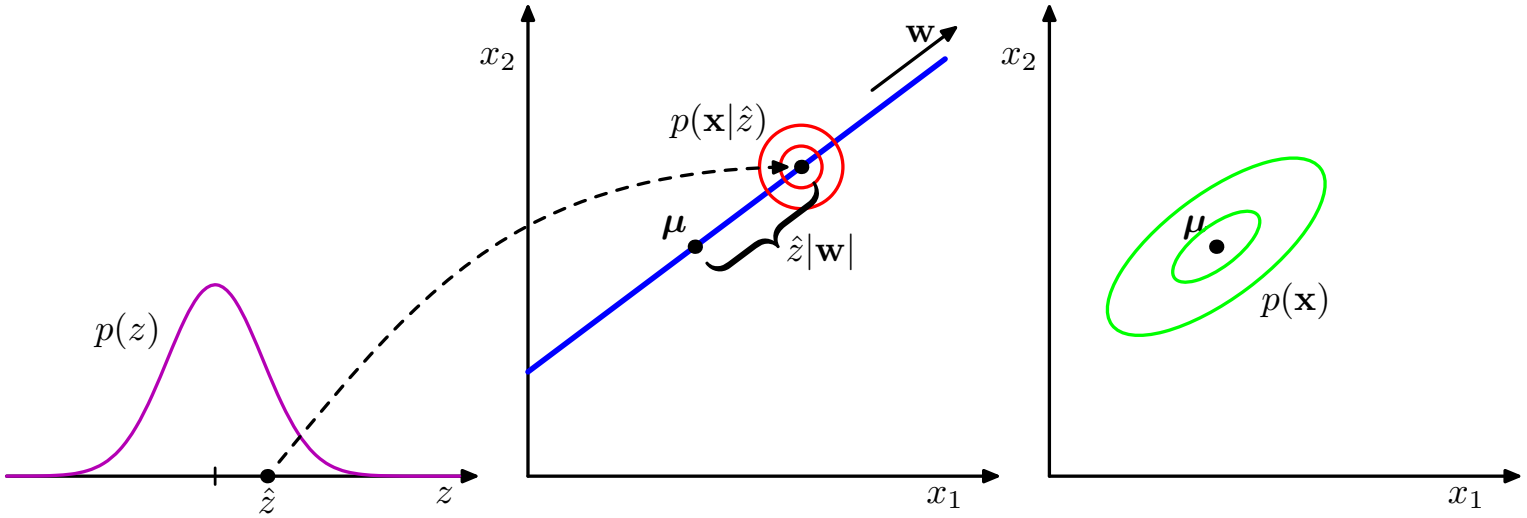


- preprocessing ("whitening" of data): we want to have comparable scales and no correlation between measurements; PCA can be used for this as $\mathbf{y}_n = \mathbf{\Lambda}^{-1/2}\mathbf{U}^T(\mathbf{x}_n - \bar{\mathbf{x}})$
- visualization: plot data points into two-dimensional plane as $\mathbf{u}_1^T\mathbf{x}_n$ and $\mathbf{u}_2^T\mathbf{x}_n$

Why probabilistic PCA?

- allows dealing with missing values in data
- allows fitting local PCAs into data with several clusters, using a mixture formulation
- leads to EM algorithm which is actually faster to compute under certain circumstances
- basis for Bayesian treatment of PCA
- can be used to model class-conditional densities and hence can be applied to classification problems: new samples are assigned to the class whose PCA they fit better
- etc.

Generative process



- $\mathbf{z} \sim N(0, \mathbf{I})$
- $\mathbf{x} = \mathbf{W}\mathbf{z} + \boldsymbol{\mu} + \boldsymbol{\epsilon}$
- $\mathbf{W}$ specifies a set of directions, $\boldsymbol{\mu}$ is the mean of the data, and the latent variable vector $\mathbf{z}$ tells how much we should move from the mean along each direction
- $\mathbf{x}$ generated by adding spherical noise $\boldsymbol{\epsilon}$ to that location

Maximum likelihood formulation

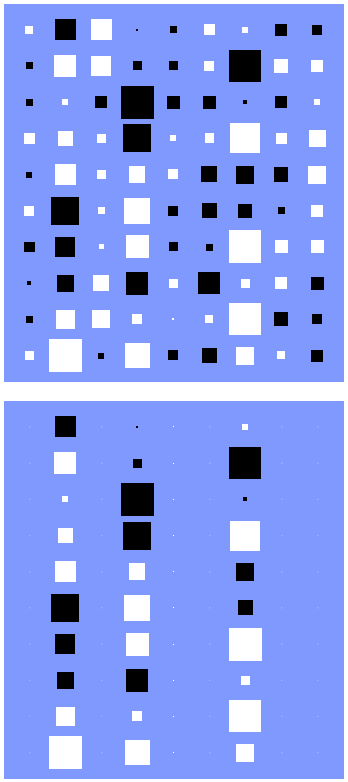
- the model parameters can be solved by maximizing the likelihood of the model given the data: $\log p(\mathbf{X}|\mathbf{W}, \mu, \sigma^2) = -\frac{N}{2}[D \log(2\pi) + \log|\mathbf{C}| + Tr(\mathbf{C}^{-1}\mathbf{S})]$, where $\mathbf{C} = \mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I}$
- the solution can be found analytically as $\mathbf{W} = \mathbf{U}(\mathbf{\Lambda} - \sigma^2\mathbf{I})^{1/2}\mathbf{R}$
- that is, $\mathbf{W}$ is the PCA projections $\mathbf{U}$ scaled according to the variances of the dimensions, multiplied by an arbitrary rotation matrix $\mathbf{R}$
- $\Rightarrow$ probabilistic PCA finds only the same subspace as traditional PCA, but not necessarily the actual projection dimensions

EM algorithm for PCA

- in practice, we can maximize the likelihood using an EM algorithm
- E-step: estimate the expected values for the latent variables **z**
- M-step: find model parameters that maximize the likelihood given the values of **z**
- computational advantage: does not require eigendecomposition of the covariance matrix, which is slow in high-dimensional spaces
  $\Rightarrow$ may be faster to compute a small-dimensional projection using the EM instead of traditional methods
- enables extensions and handling missing data

Bayesian PCA

- Bayesian PCA obtained by specifying so-called Automatic Relevance Determination (ARD) prior for **W**:
  - $\mathbf{w}_i \sim N(0, \alpha_i^{-1}\mathbf{I})$
  - when α_i is large, values of $\mathbf{w}_i$ are forced to be close to zero
- $\Rightarrow$ Bayesian PCA is able to discover how many projection dimensions are needed
- can be solved with either variational approximation or sampling



Kernel PCA

- a non-linear version of PCA can be obtained by making a non-linear transformation of the data, and applying standard PCA for the transformed values
- normal PCA: $\mathbf{S}\mathbf{u}_i = \lambda_i \mathbf{u}_i$, where $\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}_n \mathbf{x}_n^T$
- nonlinear transformation $\phi(\mathbf{x})$ leads to $\mathbf{C}\mathbf{v}_i = \lambda_i \mathbf{v}_i$, where $\mathbf{C} = \frac{1}{N}\sum_{n=1}^{N} \phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^T$
- **kernel trick**: instead of solving the above equation explicitly, we can use the kernel trick (Chapter 6)
- express $\mathbf{v}$ as a linear combination of the $\phi(\mathbf{x}_n)$: $\mathbf{v}_i = \sum_{n=1}^{N} a_{in}\phi(\mathbf{x}_n)$
- this gives: $\mathbf{K}\mathbf{a}_i = \lambda_i N \mathbf{a}_i$, where $\mathbf{K}$ is the kernel matrix with elemens $k(\mathbf{x}_n, \mathbf{x}_m) = \phi(\mathbf{x}_n)^T \phi(\mathbf{x}_m)$

Other models with continuous latent variables

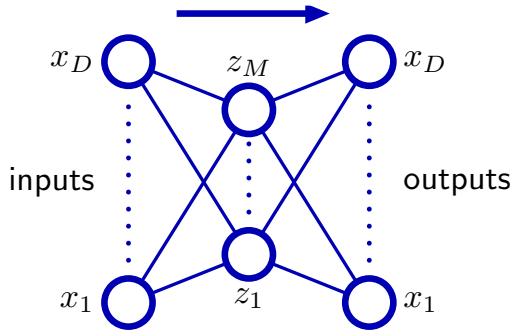
- PCA is a linear method that assumes normal distributions (for both $\mathbf{z}$ and ϵ)
- here, we consider models that have continuous latent variables, but do not make either the linearity or the Gaussianity assumption (or both)
- relaxing either assumption makes computation often considerably more complex
- purpose: to show how familiar methods can be interpreted as latent variable models

ICA - Independent Component Analysis

- latent variables represent unknown signals
- in ICA, the latent variables are independent: $p(\mathbf{z}) = \prod_{j=1}^{M} p(z_j)$
- observed data as a linear mixture of latent variables: $\mathbf{x} = \mathbf{A}\mathbf{z}$
- both $\mathbf{A}$ and $\mathbf{z}$ are unknown (blind source separation task)
- $\mathbf{z}$ can be detected if $\mathbf{z}$ have non-Gaussian distributions

Autoassociative Neural Networks

- multilayer perceptron trained to replicate inputs
- because the hidden layer is smaller, there is necessarily error in the replication



- minimization of that error gives autoassociative mapping, and the latent variables (hidden layer nodes) can be used e.g. as compressed versions of inputs
- equivalent to PCA if network is linear
- with nonlinear units and multiple layers, provides a nonlinear dimensionality reduction method

Nonlinear manifolds

- instead of a linear subspace, we may want to look for a nonlinear manifold, i.e. a curved subspace
- a mixture of linear models can approximate a non-linear surface: e.g. mixture of probabilistic PCAs
- a single nonlinear model for the surface: e.g. principal curves
- other methods: Multidimensional Scaling (MDS), Locally Linear Embedding (LLE), Isometric Feature Mapping (isomap)

Nonlinear manifolds - continued

- Generative Topographic Mapping (GTM): two-dimensional latent grid, allows using summation instead of integration in marginalization
  - note: does not actually have continuous latent variables!
- aims to preserve the topology of the data space
- similar to Self-Organizing Maps (SOM), but has a probabilistic formulation (BMU index corresponds to the latent variable)

Conclusion

- a continuous latent variable describes a location in a subspace or manifold (instead of a cluster index like with the discrete latent variables)
- PCA was viewed from three different angles: maximizes variance; minimizes reconstruction error; or probabilistic formulation
- kernel PCA provides a nonlinear extension via the kernel trick
- some other continuous latent variable models: ICA, autoassociative neural networks, and methods for modeling nonlinear manifolds

Questions?