Probabilistic generative models and unsupervised learning II

Zoubin Ghahramani

Department of Engineering
University of Cambridge, UK

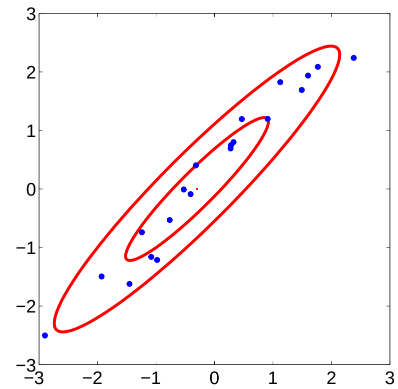
Machine Learning Department
Carnegie Mellon University, USA

zoubin@eng.cam.ac.uk
http://learning.eng.cam.ac.uk/zoubin/

IPAM Probabilistic Models of Cognition
Lectures July 2007

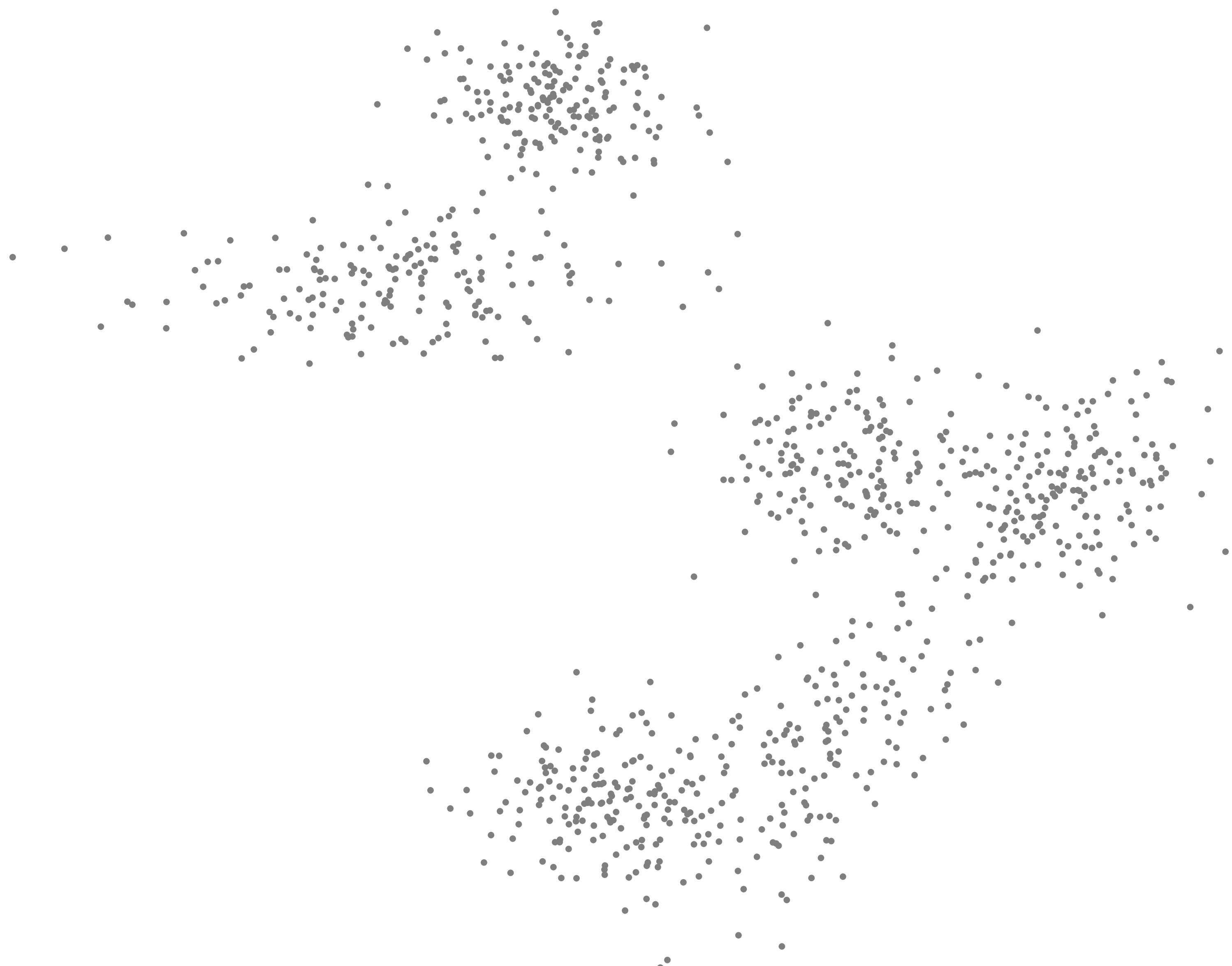
Limitations of the Multivariate Gaussian

Gaussians are fundamental and widespread, but not every distribution of interest is Gaussian.

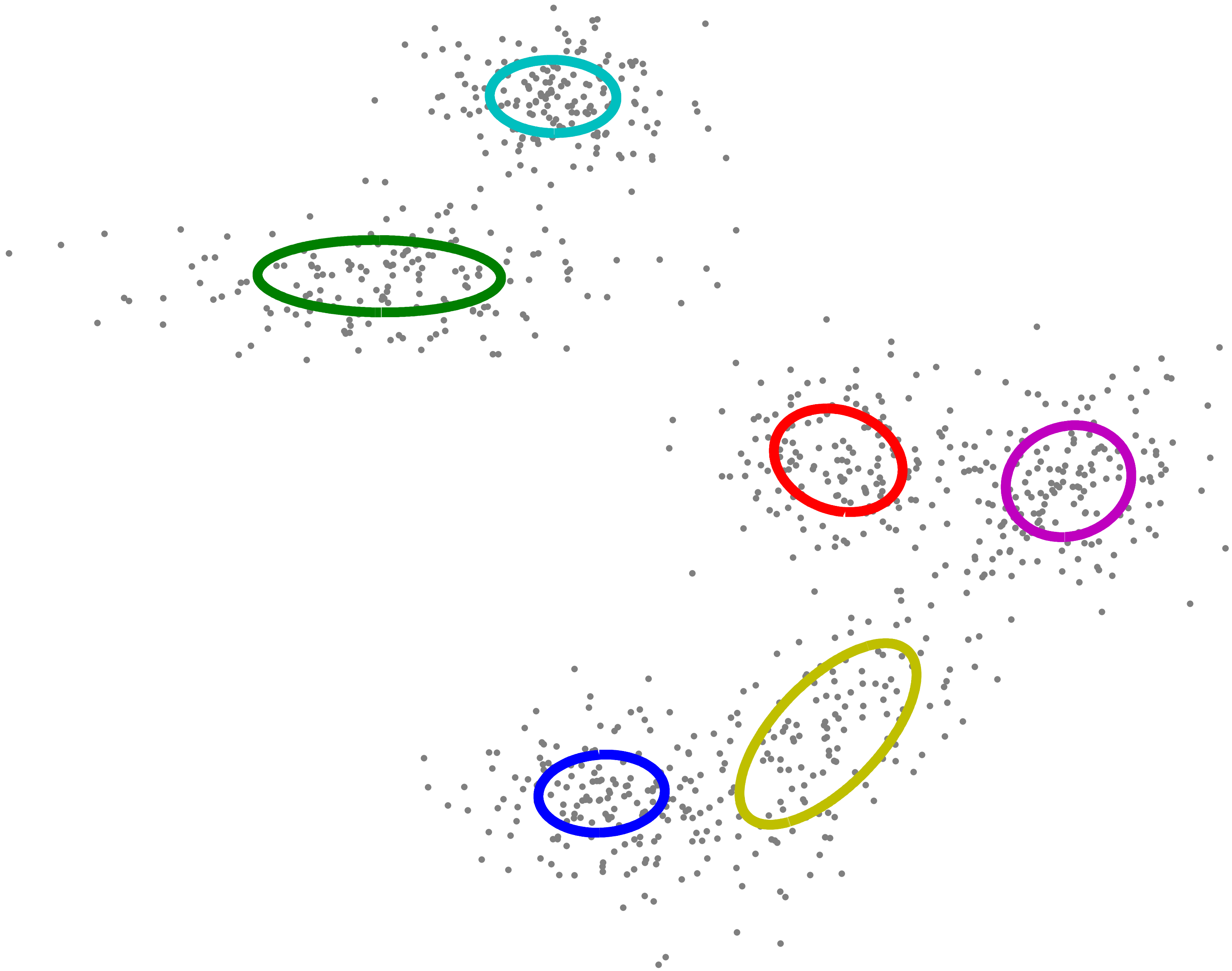


- Some processes produce outliers.
- Some data has higher-order or non-linear structure.
- Not all random processes fit the central limit theorem.
- Even if data are Gaussian, if D is large the full multivariate Gaussian model may be difficult to handle. There are $D(D+1)/2$ parameters in the covariance matrix.

What about this data?



What about this data?



What about this data?

`embed_demo`

Outline

- Clustering, K-means, and Mixture models
- Dimensionality reduction, Factor analysis
- Latent Variable Models
- The EM algorithm

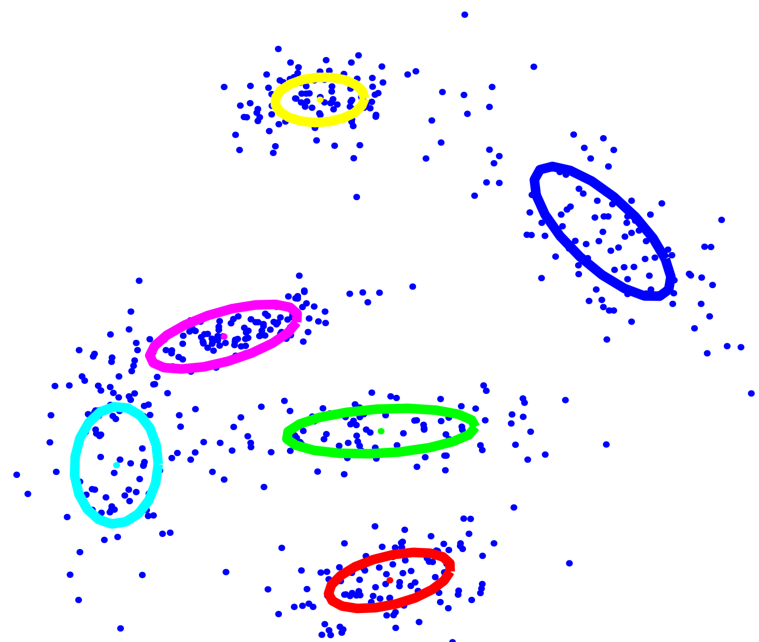
Clustering

Given some data, the goal is to discover "clusters" of points.

Roughly speaking, two points belonging to the same cluster are generally more similar to each other or closer to each other than two points belonging to different clusters.

Examples:

- cluster news stories into topics
- cluster genes by similar function
- cluster movies into categories
- cluster astronomical objects



The K-Means Algorithm

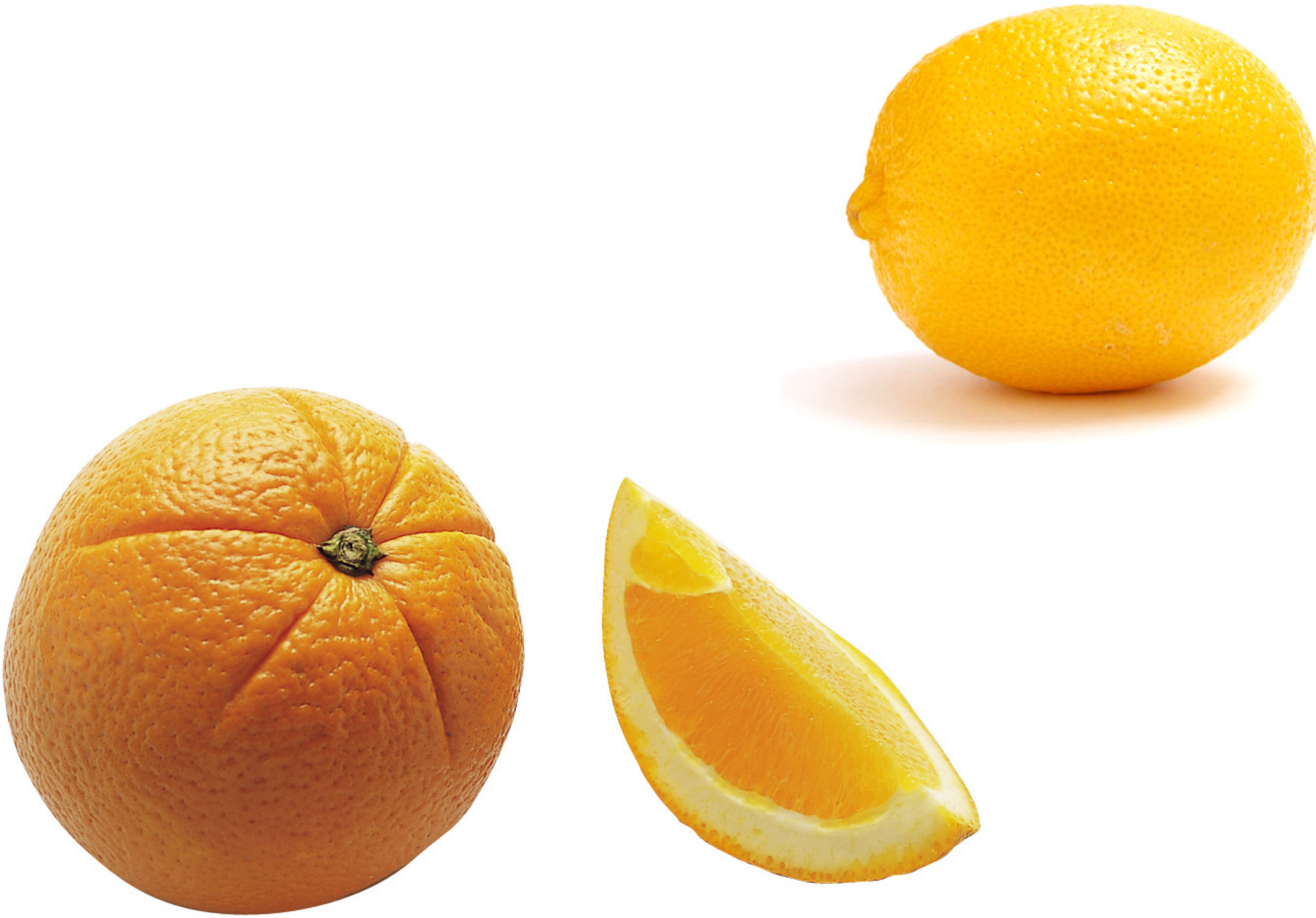
Input: `Data Set` $\mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ `where` $\mathbf{x}_n \in \Re^D$

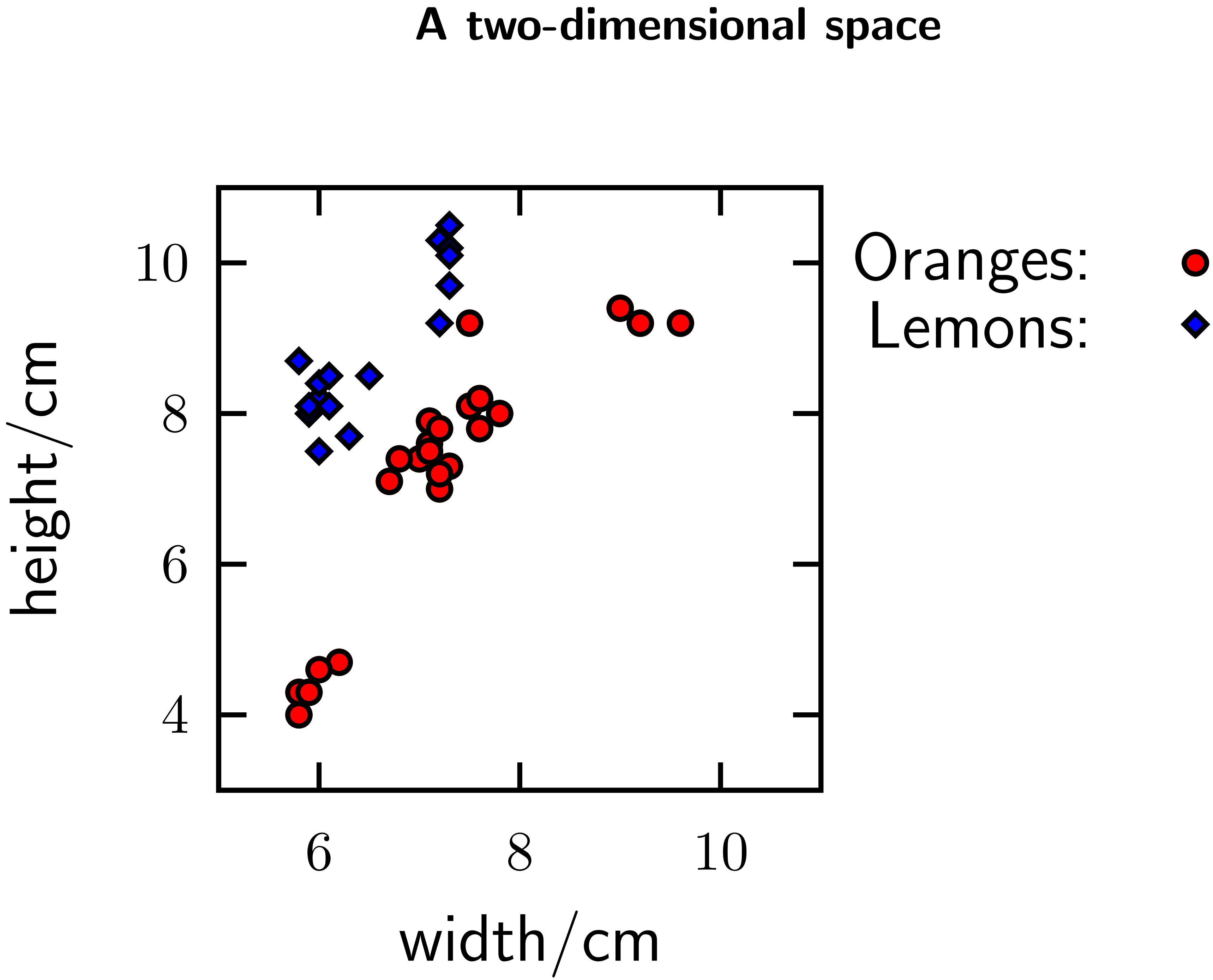
Initialize Centers: $\mathbf{m}_k \in \Re^D$ `for` $k = 1 \ldots K$.

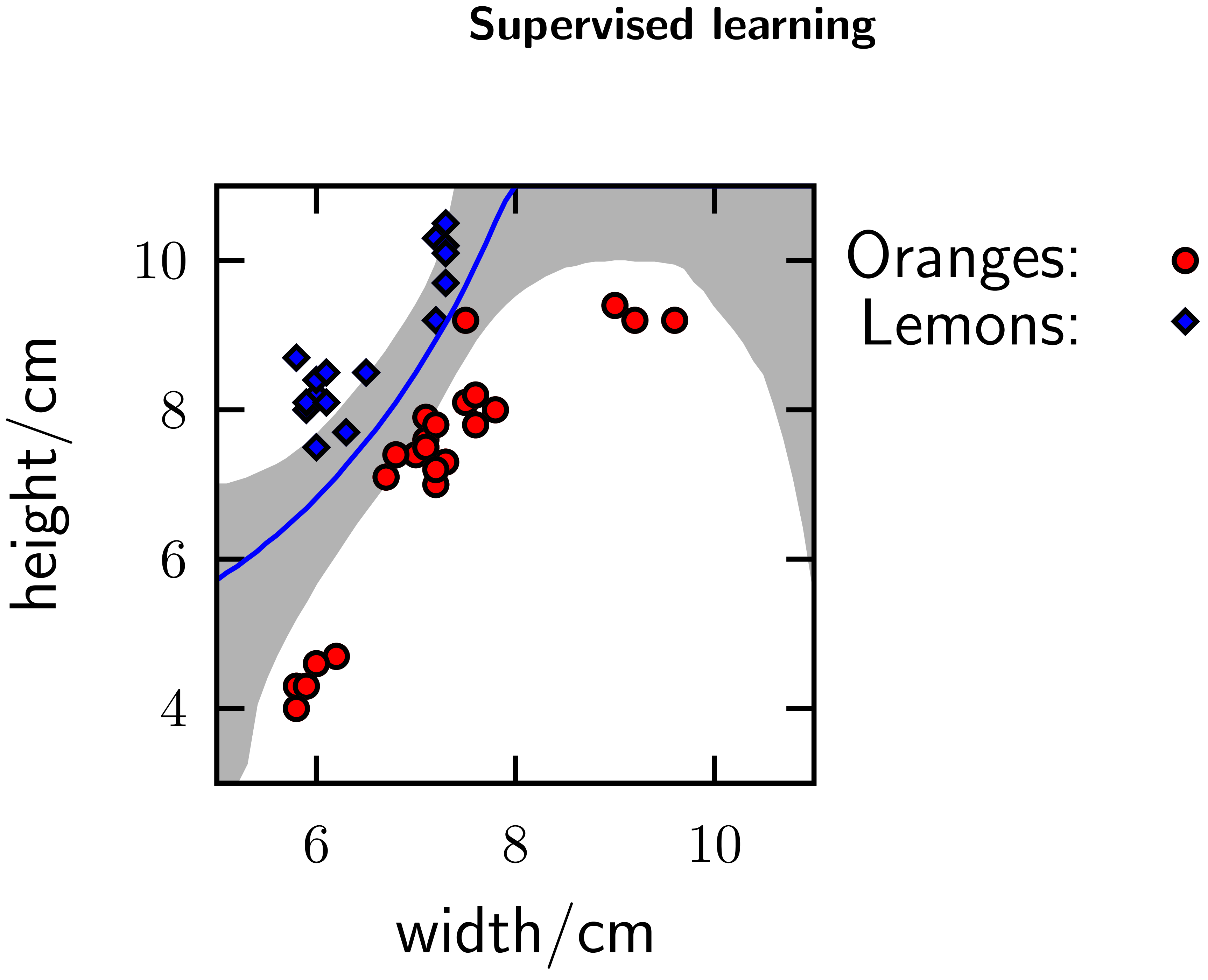
```
repeat:

  for n = 1...N:

    let s_n = argmin_k ||x_n - m_k|| % assign data points to nearest
  center

  end for


  for k = 1...K:

    let m_k = mean{x_n : s_n = k} % re-compute means

  end for

until convergence (s has not changed)
```

kmeansdemo

Oranges and Lemons
Thanks to Iain Murray

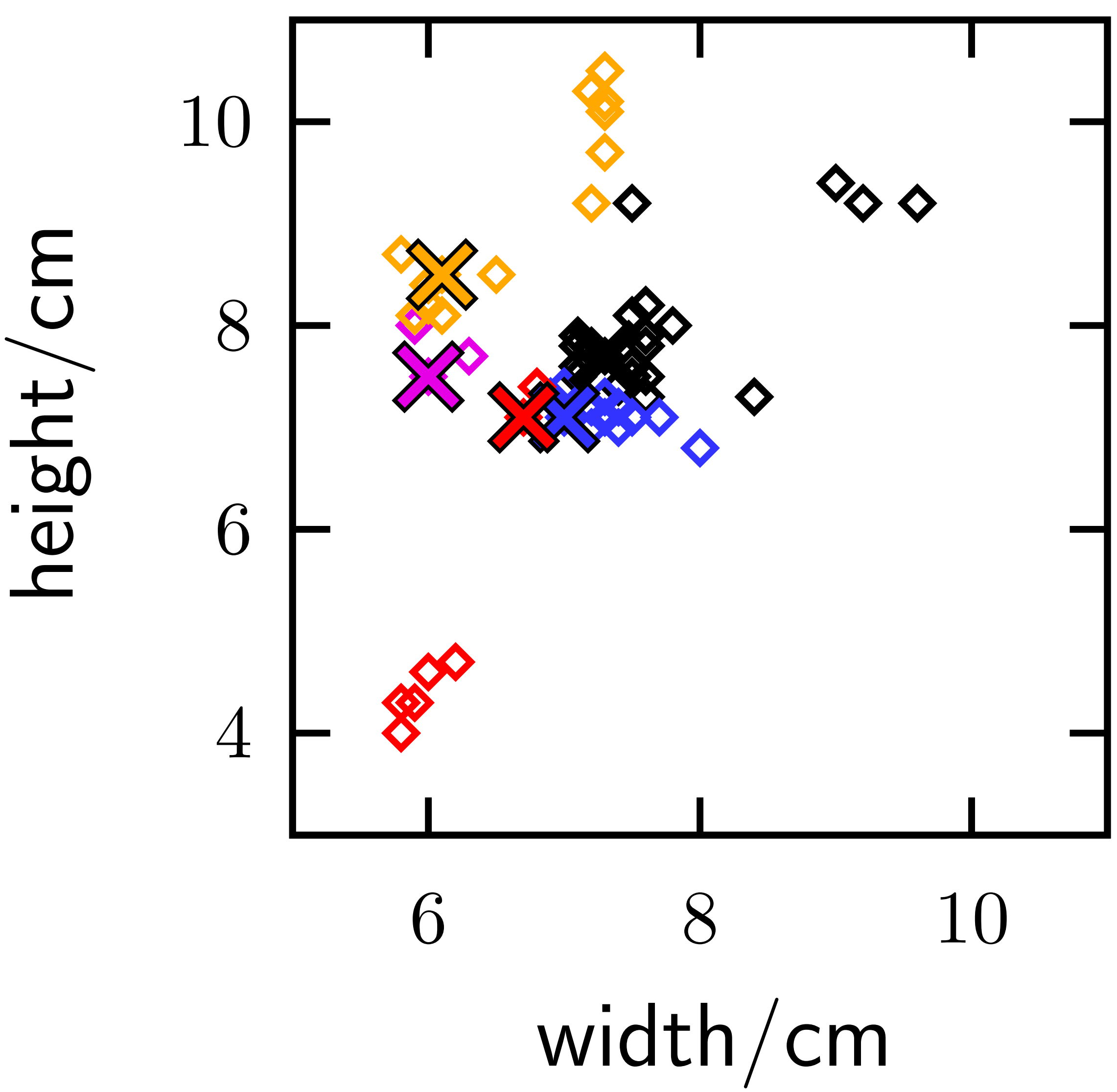



A two-dimensional space
10
8
6
4
height/cm
6
8
10
width/cm
Oranges:
Lemons:



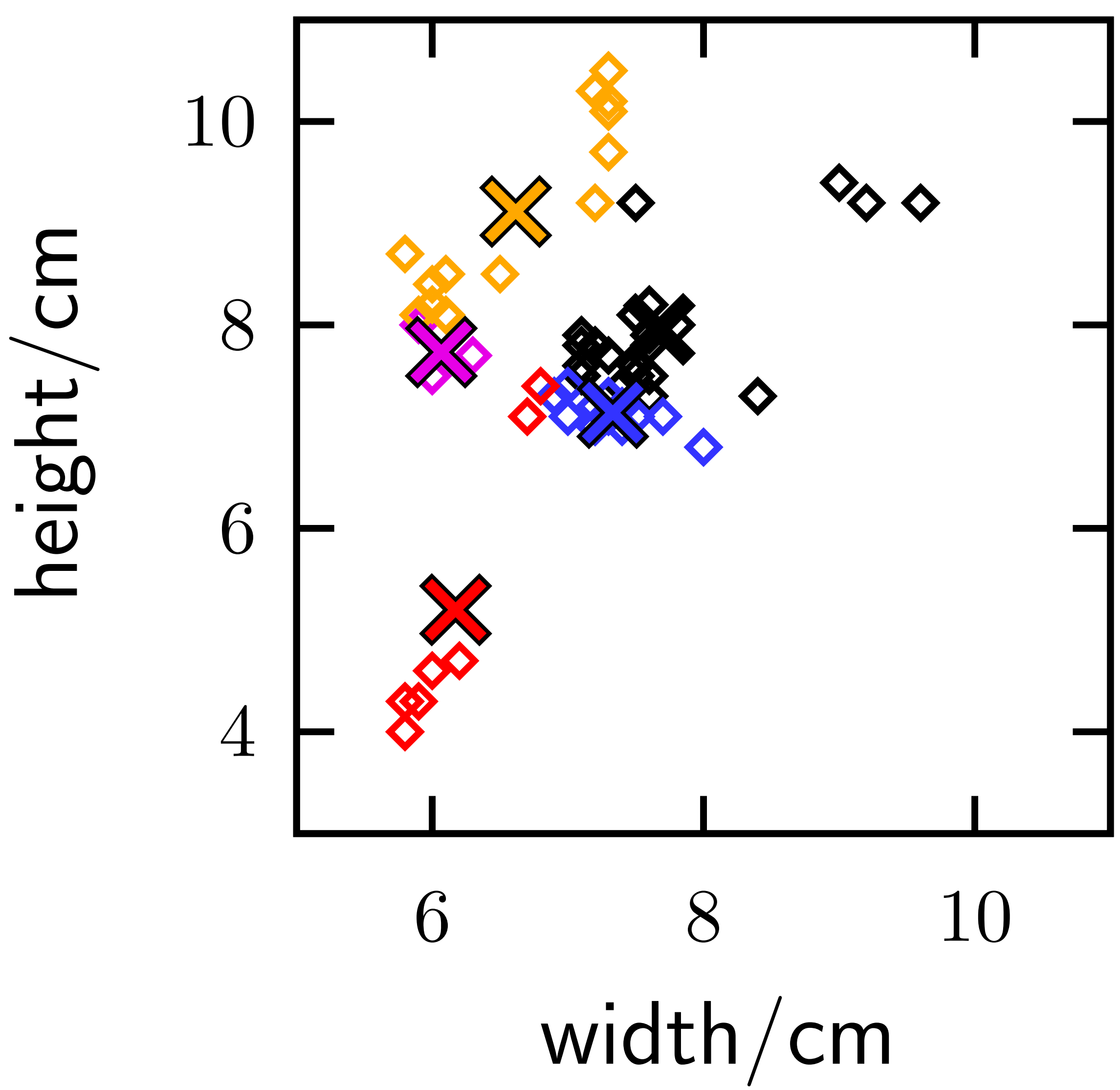
Supervised learning
10
8
6
4
6
8
10
height/cm
width/cm
Oranges:
Lemons:

Clustering

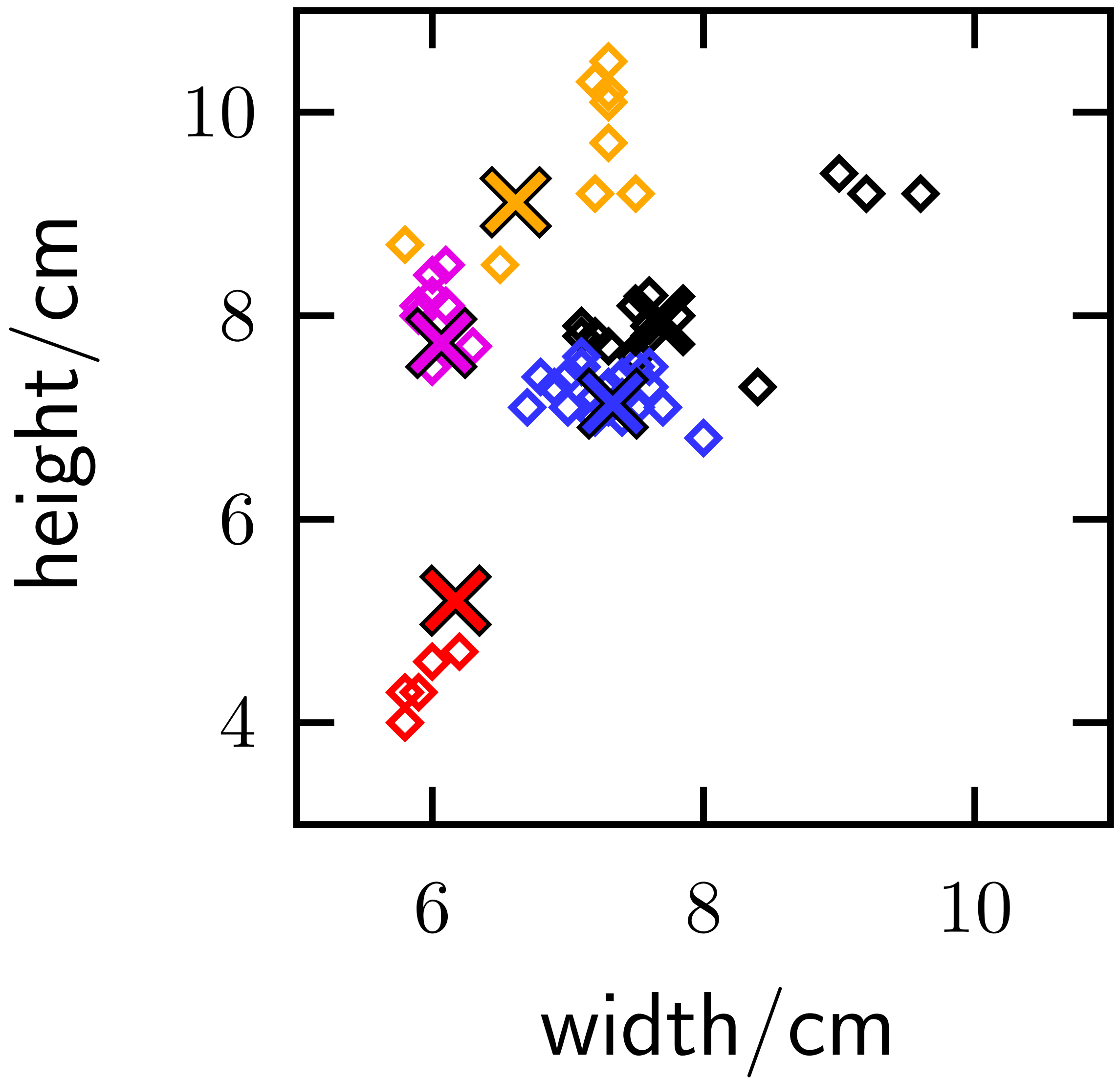


start end

Clustering

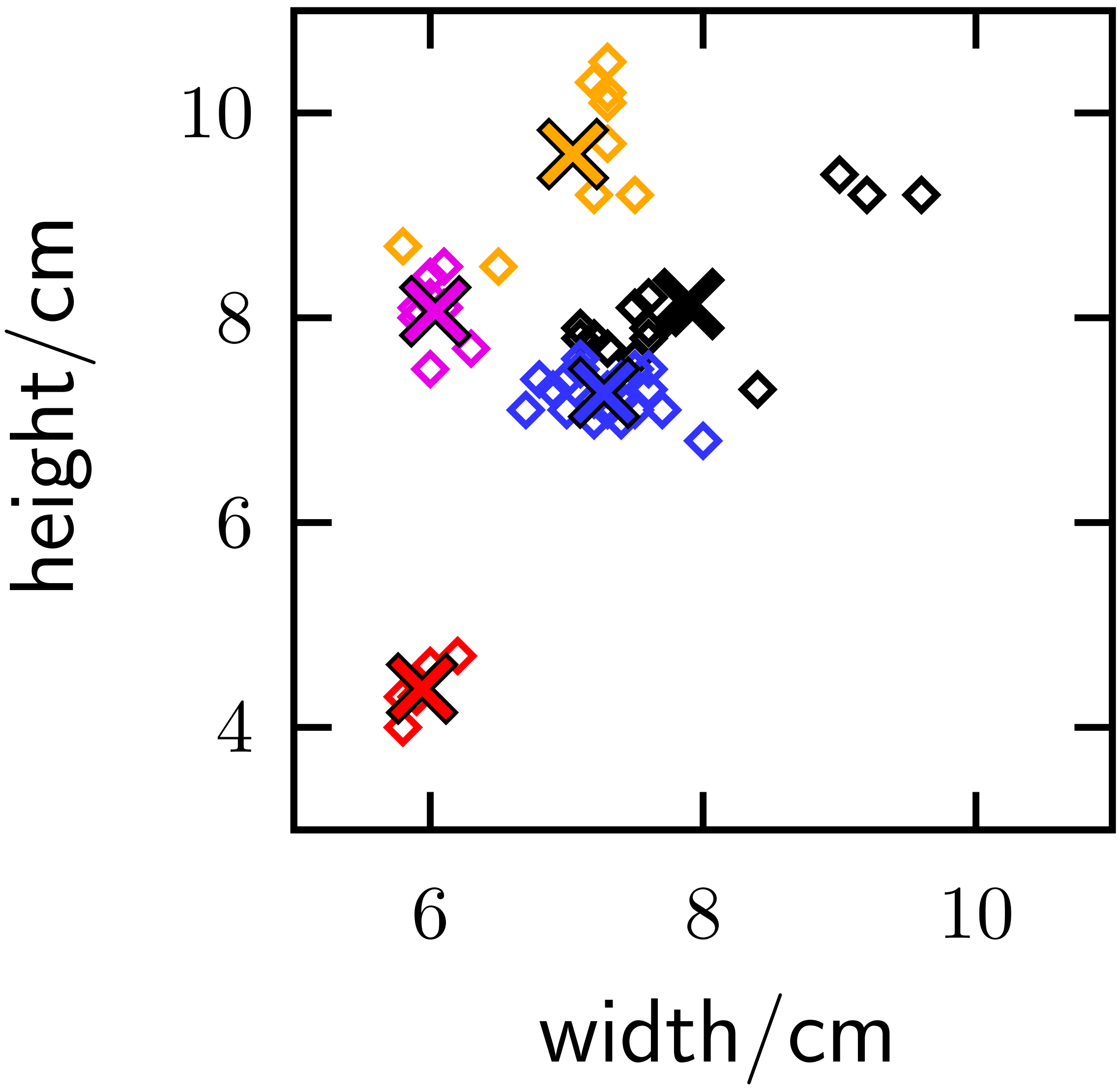


Clustering



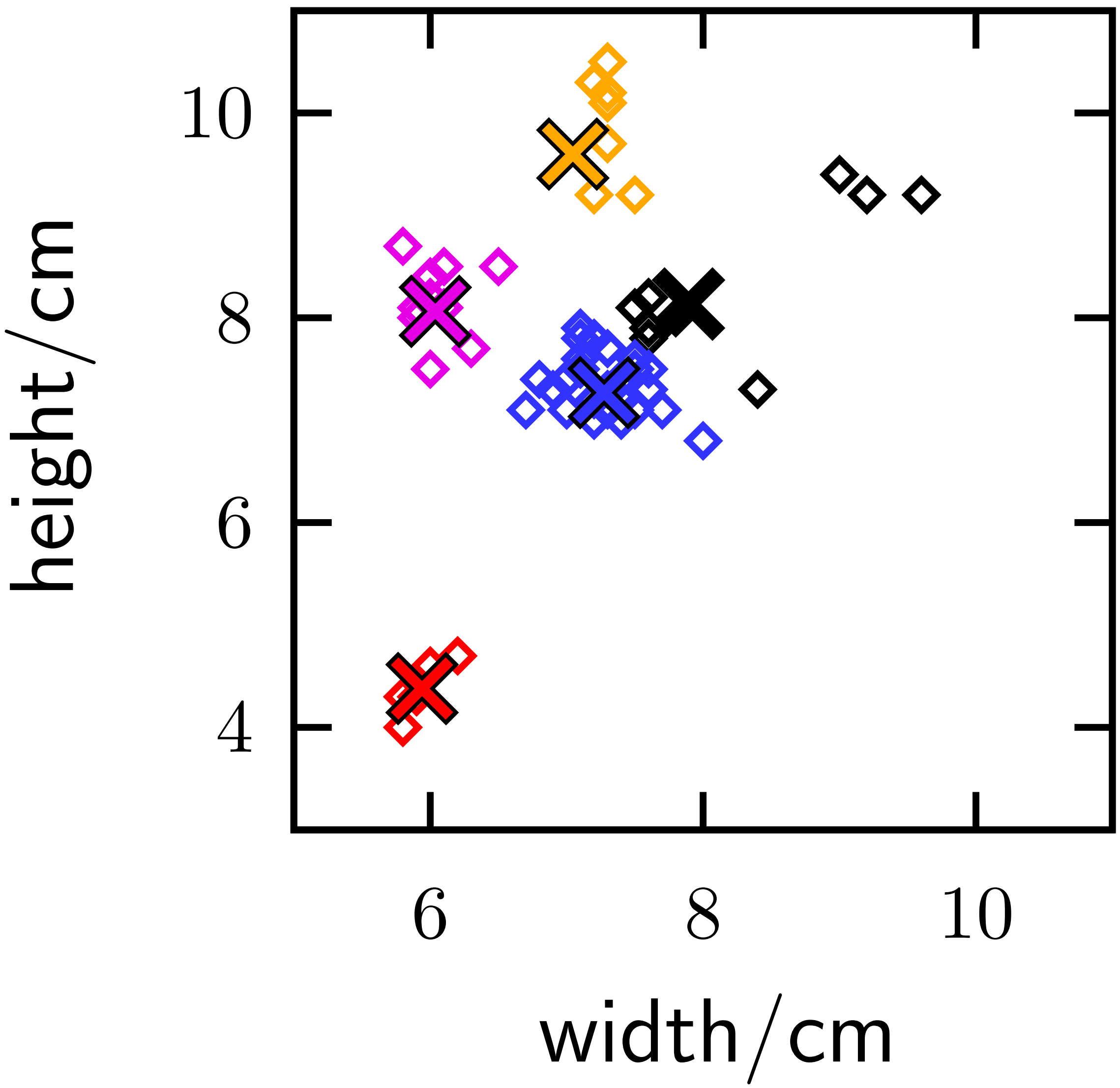
start end

Clustering



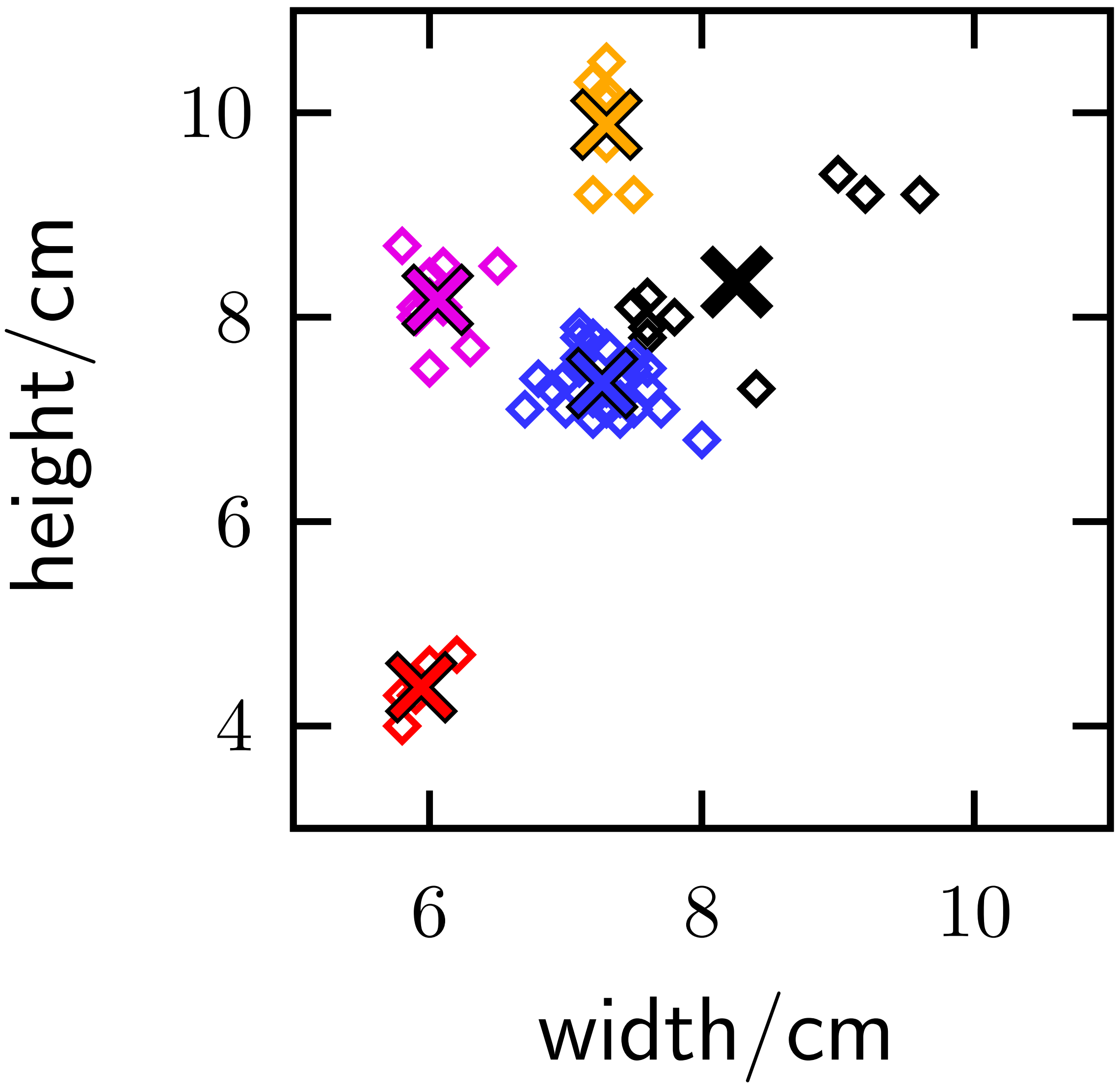
start end

Clustering



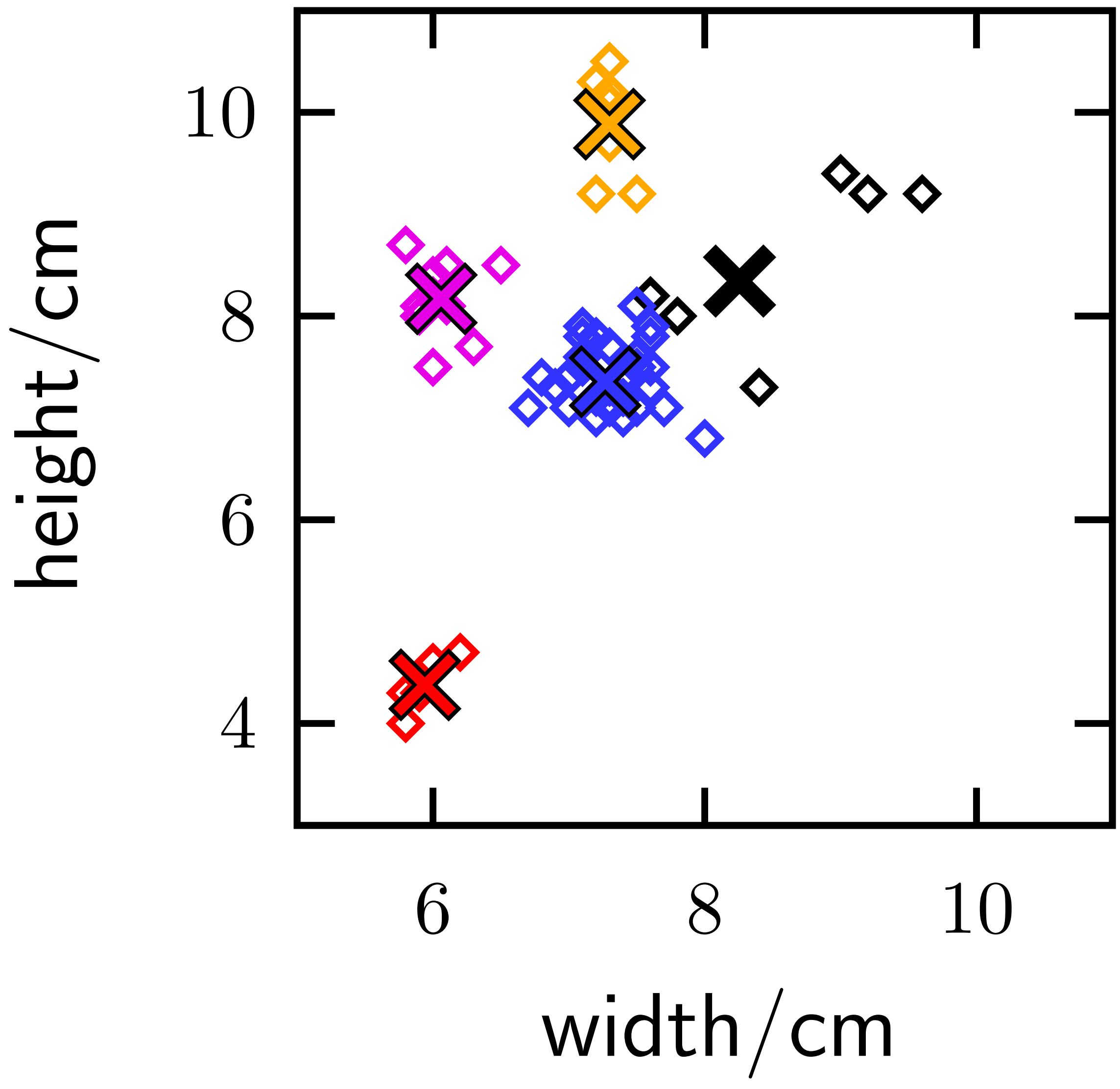
start end

Clustering



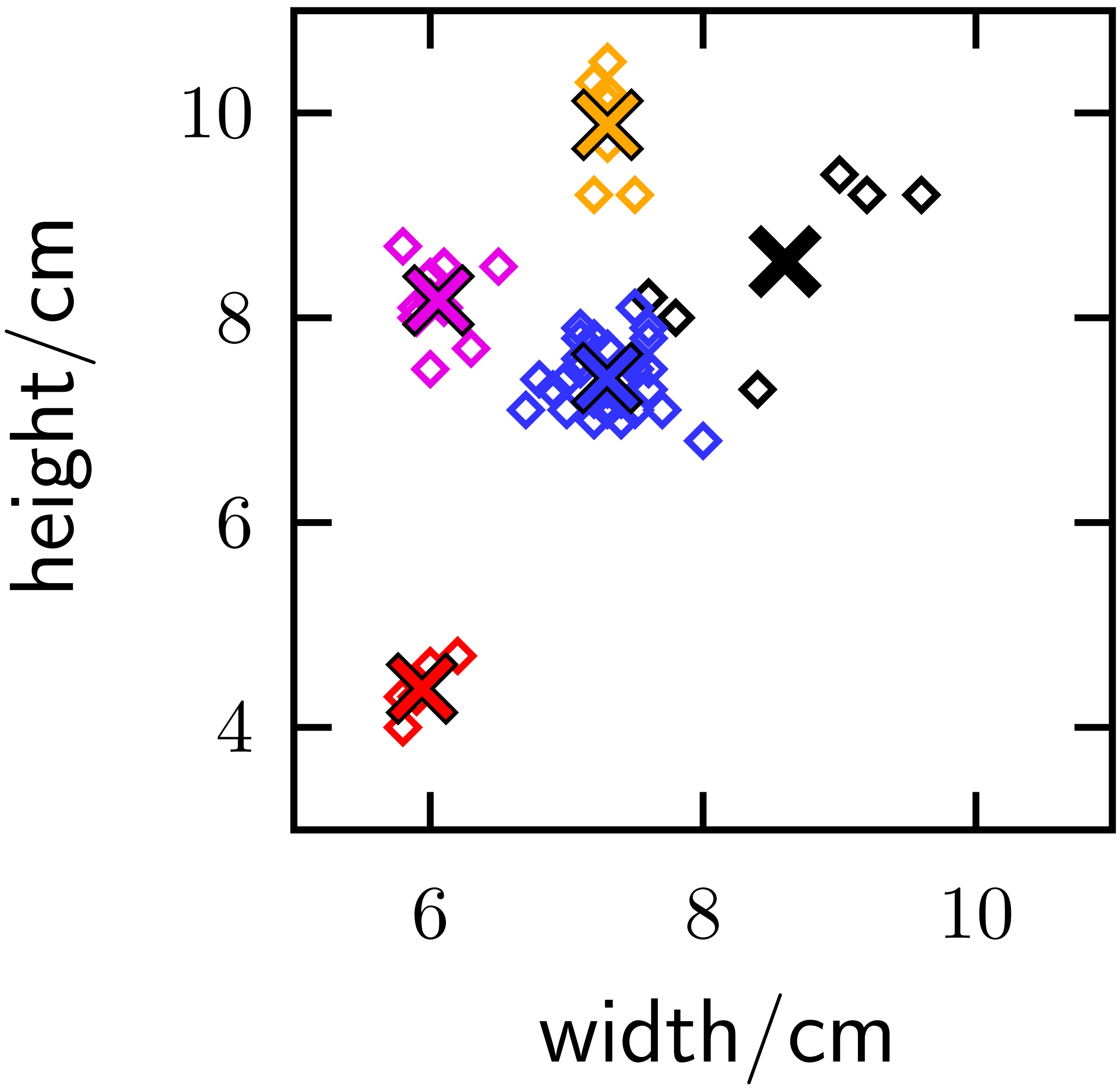
start end

Clustering

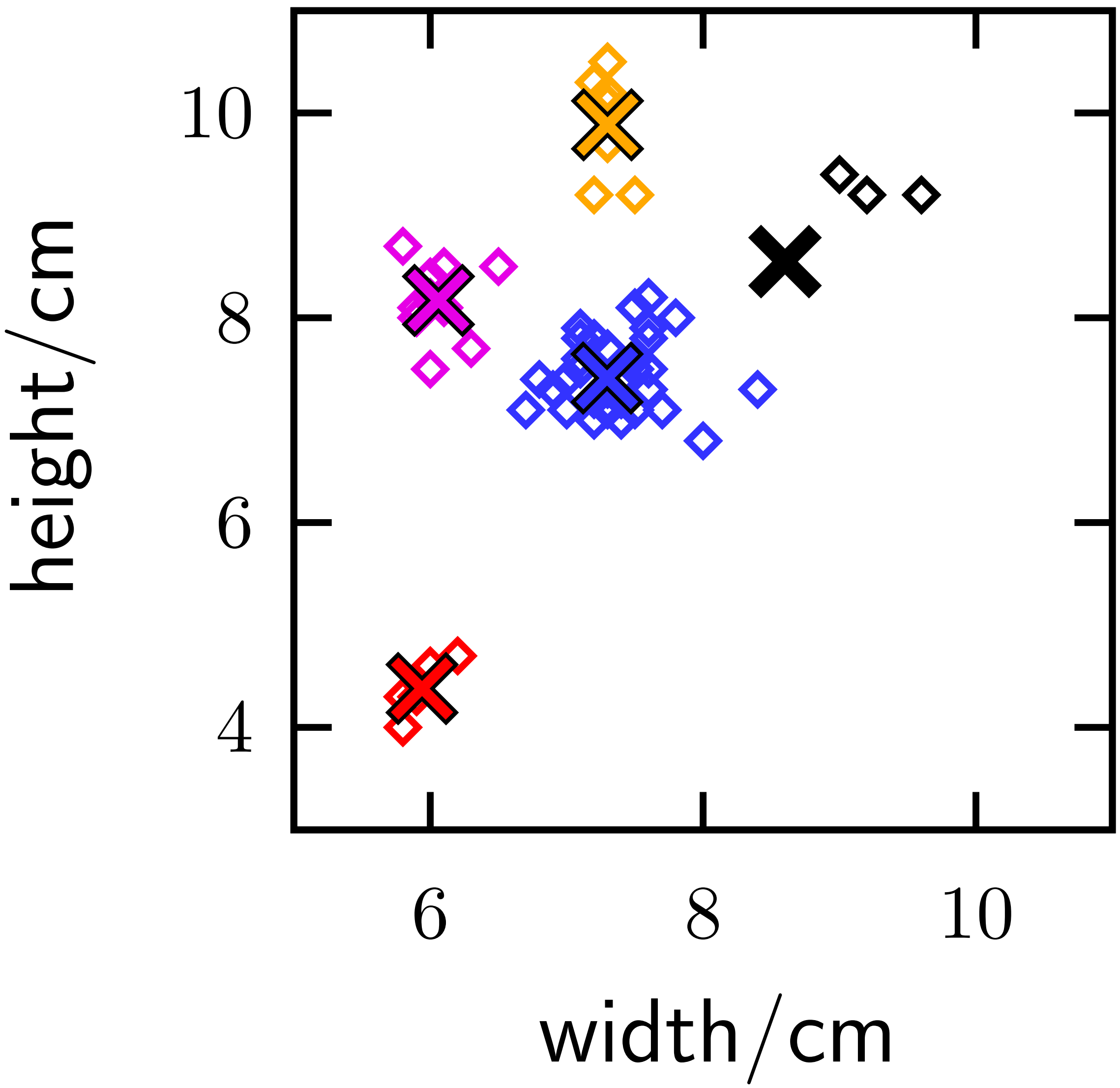


start end

Clustering

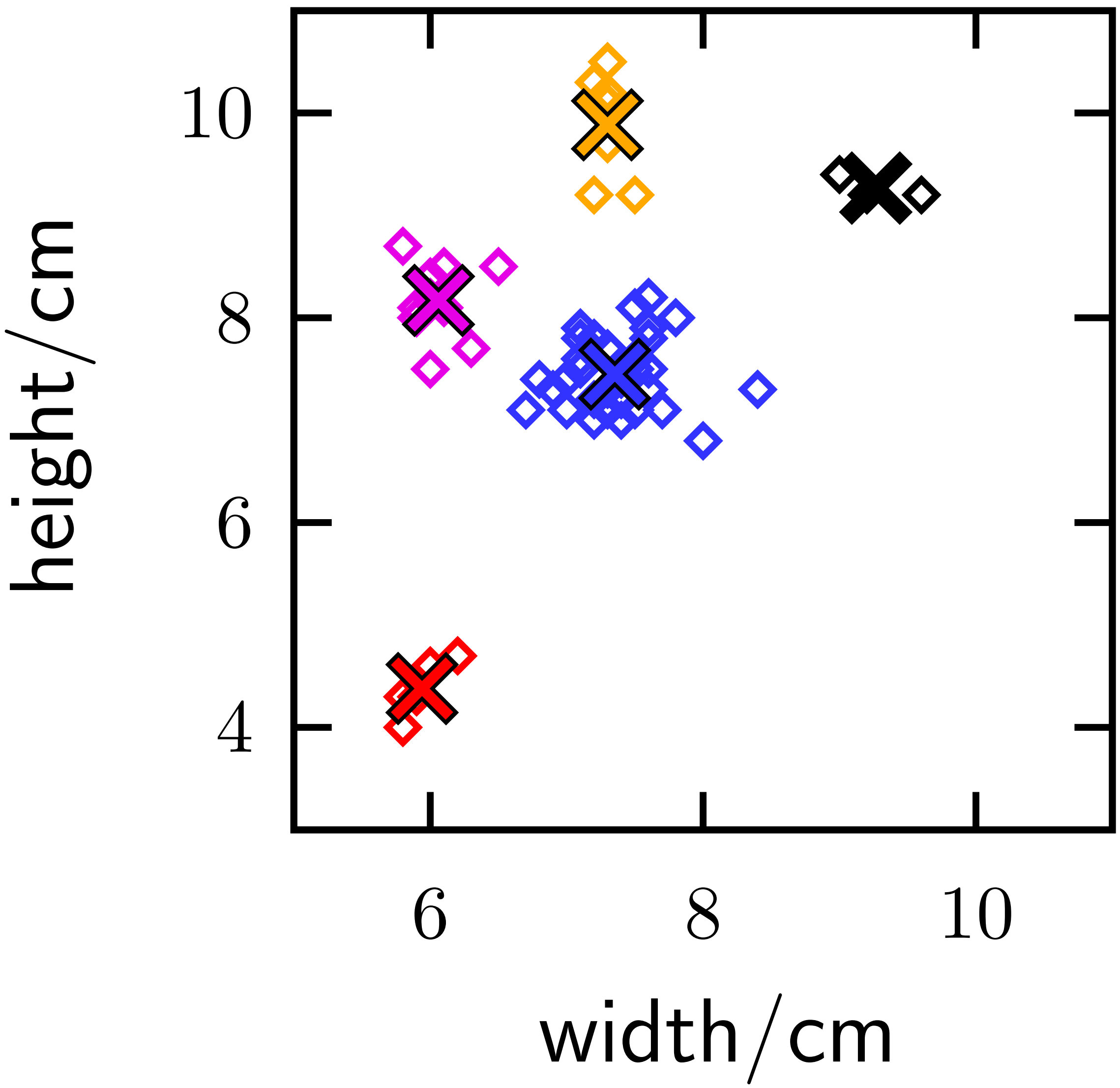


Clustering



start end

Clustering



start end

A Cost Function for K-means

Let $s_{nk} = 1$ if data point n is assigned to cluster k and zero otherwise.

Note: $\sum_k s_{nk} = 1$.

Cost

$$C = \sum_{nk} s_{nk} \|\mathbf{x}_n - \mathbf{m}_k\|^2$$

The K-means algorithm tries to minimize the cost function C with respect to $\{s_{nk}\}$ and $\{\mathbf{m}_k\}$, subject to $\sum_k s_{nk} = 1$ and $s_{nk} \in \{0, 1\}$.

K-means:

- minimize C with respect to $\{s_{nk}\}$, holding $\{\mathbf{m}_k\}$ fixed.
- minimize C with respect to $\{\mathbf{m}_k\}$, holding $\{s_{nk}\}$ fixed.

Finding the global optimum of C is a *hard* problem.

A probabilistic interpretation of K-means

Multivariate Gaussian density ($\mathbf{x} \in \Re^D$):

$$p(\mathbf{x}|\mu, \Sigma) = |2\pi\Sigma|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu)^\top \Sigma^{-1}(\mathbf{x}-\mu)\right\}$$

Multivariate Gaussian density with mean $\mathbf{m}_k$ and identity covariance matrix I.

$$\begin{aligned} p(\mathbf{x}|\mathbf{m}_k) &= |2\pi I|^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mathbf{m}_k)^\top(\mathbf{x}-\mathbf{m}_k)\right\} \\ p(\mathbf{x}|\mathbf{m}_k) &= \frac{1}{(2\pi)^{D/2}} \exp\left\{-\frac{1}{2}\|\mathbf{x}-\mathbf{m}_k\|^2\right\} \end{aligned}$$

A mixture model:

$$p(\mathbf{x}_n|\{\mathbf{m}_k\}) = \sum_k w_k \, p(\mathbf{x}_n|\mathbf{m}_k)$$

where w_k is the mixing proportion (e.g. set $w_k = 1/K$).

A probabilistic interpretation of K-means

Multivariate Gaussian density with mean $\mathbf{m}_k$ and identity covariance matrix I.

$$p(\mathbf{x}|\mathbf{m}_k) = \frac{1}{(2\pi)^{D/2}} \exp\left\{-\frac{1}{2}\|\mathbf{x} - \mathbf{m}_k\|^2\right\}$$

A mixture model:

$$p(\mathbf{x}_n|\{\mathbf{m}_k\}) = \sum_k w_k \, p(\mathbf{x}_n|\mathbf{m}_k)$$

where w_k is the mixing proportion (e.g. set $w_k = 1/K$).

Imagine we observed which data points came from which Gaussians (i.e. we knew $\{s_{nk}\}$), then:

$$p(\mathbf{x}_n, \mathbf{s}_n|\{\mathbf{m}_k\}) = \prod_k \left[w_k \, p(\mathbf{x}_n|\mathbf{m}_k)\right]^{s_{nk}}$$

Likelihood:

$$p(\mathbf{X}, \mathbf{S}|\{\mathbf{m}_k\}) = \prod_n p(\mathbf{x}_n, \mathbf{s}_n|\{\mathbf{m}_k\}) = \prod_{nk} \left[w_k \, p(\mathbf{x}_n|\mathbf{m}_k)\right]^{s_{nk}}$$

A probabilistic interpretation of K-means

Multivariate Gaussian density with mean $\mathbf{m}_k$ and identity covariance matrix I.

$$p(\mathbf{x}|\mathbf{m}_k) = \frac{1}{(2\pi)^{D/2}} \exp\left\{-\frac{1}{2}\|\mathbf{x} - \mathbf{m}_k\|^2\right\}$$

Likelihood: $p(\mathbf{X}, \mathbf{S}|\{\mathbf{m}_k\}) = \prod_n p(\mathbf{x}_n, \mathbf{s}_n|\{\mathbf{m}_k\}) = \prod_{nk} [w_k \, p(\mathbf{x}_n|\mathbf{m}_k)]^{s_{nk}}$

Log Likelihood if we set $w_k = 1/K$:

$$\begin{aligned}
\ln p(\mathbf{X}, \mathbf{S}|\{\mathbf{m}_k\}) &= \sum_{nk} s_{nk} \left[\log w_k + \log p(\mathbf{x}_n|\mathbf{m}_k)\right] \\
&= \sum_{nk} s_{nk} \log p(\mathbf{x}_n|\mathbf{m}_k) - N \log K \\
&= -\frac{1}{2}\left[\sum_{nk} s_{nk} \|\mathbf{x}_n - \mathbf{m}_k\|^2\right] - \frac{ND}{2} \log(2\pi) - N \log K
\end{aligned}$$

Maximizing $\ln p(\mathbf{X}, \mathbf{S}|\{\mathbf{m}_k\})$ with respect to $\{s_{nk}\}$ and $\{\mathbf{m}_k\}$ is equivalent to minimizing the K-means cost function.

Mixtures Models and Latent Variables

The general distribution for a mixture model is

$$P(\mathbf{x}_n|\boldsymbol{\theta}) = \sum_{k=1}^{K} P(s_n = k|\mathbf{w})\, P(\mathbf{x}_n|s_n = k, \boldsymbol{\theta}_k)$$

where $s_n = k$ means that data point n was generated by mixture component k.

The prior probability that data point n was generated from component k is w_k

$$P(s_n = k|\mathbf{w}) = w_k$$

These mixing proportions satisfy $\sum w_k = 1$.

The component can have any distribution $P(\mathbf{x}_n|s_n = k, \boldsymbol{\theta}_k)$.

What is the posterior probability that data point $\mathbf{x}_n$ came from component k? By Bayes rule:

$$r_{nk} = \frac{w_k\, P(\mathbf{x}_n|s_n = k, \boldsymbol{\theta}_k)}{\sum_{k'} w_{k'}\, P(\mathbf{x}_n|s_n = k', \boldsymbol{\theta}_{k'})}$$

This is a soft version of the hard assignments in K-means.

From K-means to EM...

Consider the following K-means like algorithm:

- Fix $\{\mathbf{w}, \{\boldsymbol{\theta}_k\}\}$ and compute $\{r_{nk}\}$
- Fix $\{r_{nk}\}$ and recompute $\{\mathbf{w}, \{\boldsymbol{\theta}_k\}\}$ from data weighted by $\{r_{nk}\}$.

For mixtures of Gaussians:

$$w_k = \frac{\sum_n r_{nk}}{\sum_{nk'} r_{nk'}} = \frac{\sum_n r_{nk}}{N}$$

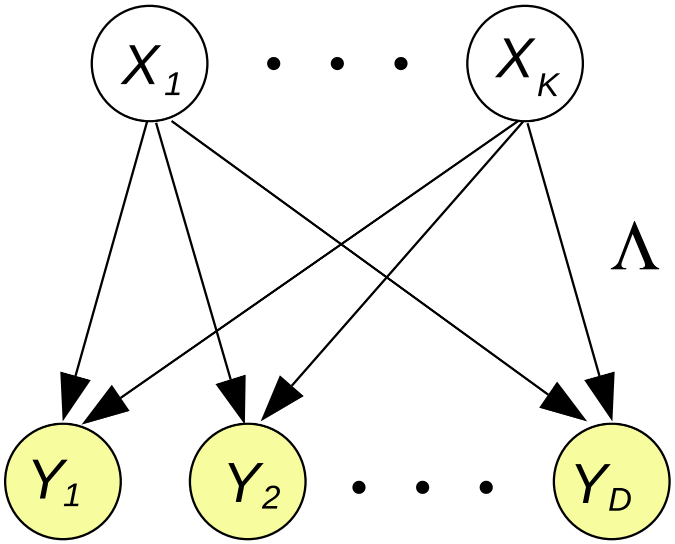
$$\mu_k = \frac{\sum_n r_{nk} \mathbf{x}_n}{\sum_n r_{nk}}$$

$$\boldsymbol{\Sigma}_k = \frac{\sum_n r_{nk} (\mathbf{x}_n - \mu_k)(\mathbf{x}_n - \mu_k)^\top}{\sum_n r_{nk}}$$

This is the Expectation-Maxization (EM) algorithm. We will give a theoretical justification soon...

Factor Analysis

Factor analysis models high dimensional data $\mathbf{y}$ in terms of a linear transformation of some smaller number of latent factors, $\mathbf{x}$.



Linear generative model: $y_d = \sum_{k=1}^{K} \Lambda_{dk}\, x_k + \epsilon_d$

- x_k are independent $\mathcal{N}(0,1)$ Gaussian factors
- ϵ_d are independent $\mathcal{N}(0,\Psi_{dd})$ Gaussian noise
- $K < D$

Properties:

- $p(\mathbf{x}) = \mathcal{N}(0, I)$ and $\mathbf{y} = \Lambda\mathbf{x} + \epsilon$
- Since $p(\epsilon) = \mathcal{N}(0, \Psi)$, we get that $p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\Lambda\mathbf{x}, \Psi)$
- $p(\mathbf{y}) = \int p(\mathbf{x})p(\mathbf{y}|\mathbf{x})d\mathbf{x} = \mathcal{N}(0, \Lambda\Lambda^\top + \Psi)$ where Λ is a $D \times K$ matrix, and Ψ is diagonal.

latent = hidden = unobserved = missing

Ways of thinking about Factor Analysis (FA)

- FA models high dimensional data in terms of a linear transformation of some smaller number of latent factors.

- FA is a method for parameterizing a $D \times D$ covariance matrix Σ in terms of $D \times K + D$ parameters, $\Lambda\Lambda^\top + \Psi$. Since K can be chosen by the user, this means that factor analysis can be applied to *very* high dimensional datasets.

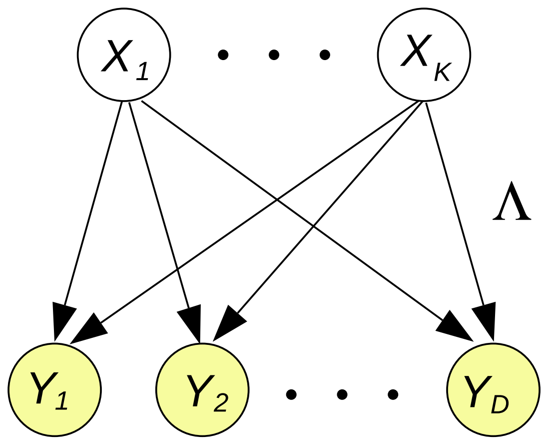
- FA is a method for modelling correlations among the observed variables.

- FA is a linear regression model, where the inputs are assumed to be hidden.

- FA is a method for doing dimensionality reduction. Given $\mathbf{y}$ we can represent it by the mean of $\mathbf{x}$. FA finds a low-dimensional projection of high dimensional data that captures the correlation structure of the data.

$$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{x})p(\mathbf{y}|\mathbf{x})}{p(\mathbf{y})} = \mathcal{N}(\beta\mathbf{y}, I - \beta\Lambda) \quad \text{where} \quad \beta = \Lambda^\top(\Lambda\Lambda^\top + \Psi)^{-1}$$

Factor Analysis



$$\mu = 0$$

$$\Sigma \approx \Lambda\Lambda^\top + \Psi$$

- ML learning for FA aims to fit Λ and Ψ given data. There is no closed form solution for ML parameters.

- Number of free parameters (corrected for symmetries):

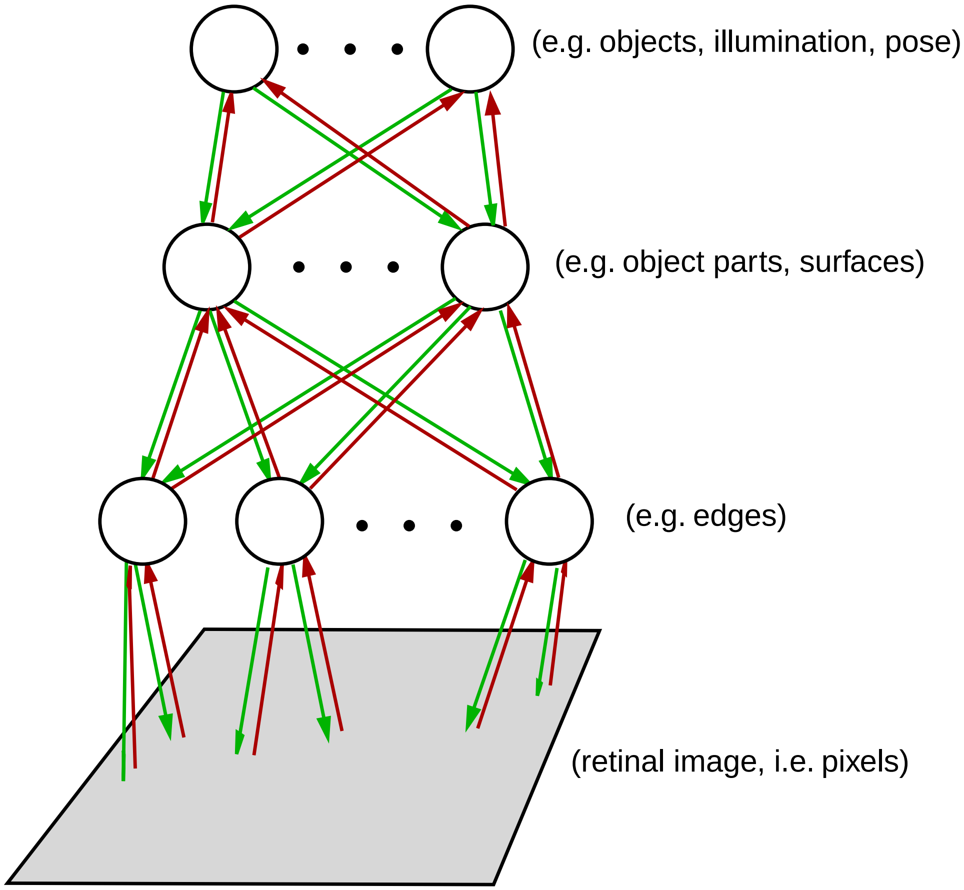
$$DK + D - \frac{K(K-1)}{2} < \frac{D(D+1)}{2}$$

- A Bayesian treatment would start with priors over Λ and Ψ and infer their posterior given the data.

$$p(\Lambda, \Psi|\mathcal{D}) = \frac{p(\mathcal{D}|\Lambda, \Psi)p(\Lambda, \Psi)}{p(\mathcal{D})}$$

Latent Variable Models

Explain correlations in $\mathbf{y}$ by assuming some latent variables $\mathbf{x}$



$$\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta}_x)$$

$$\mathbf{y}|\mathbf{x} \sim p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}_y)$$

$$p(\mathbf{x}, \mathbf{y}|\boldsymbol{\theta}_x, \boldsymbol{\theta}_y) = p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}_y) p(\mathbf{x}|\boldsymbol{\theta}_x)$$

$$p(\mathbf{y}|\theta_x, \boldsymbol{\theta}_y) = \int d\mathbf{x}\; p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}_y) p(\mathbf{x}|\boldsymbol{\theta}_x)$$

The EM Algorithm

- Latent variable models:[1] model data $\mathbf{y}_n$ in terms of latent variables $\mathbf{x}_n$.
- Data set $\mathcal{D} = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$, likelihood: $p(\mathcal{D}|\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\mathbf{y}_n|\boldsymbol{\theta}) = \prod_n \int p(\mathbf{y}_n, \mathbf{x}_n|\boldsymbol{\theta}) d\mathbf{x}_n$
- Goal: learn maximum likelihood (ML) parameter values
- The maximum likelihood procedure finds parameters $\boldsymbol{\theta}$ such that:

$$\boldsymbol{\theta}_{\mathrm{ML}} = \mathrm{argmax}_{\boldsymbol{\theta}}\, p(\mathcal{D}|\boldsymbol{\theta})$$

- Because of the integral (or sum) over latent variables, the likelihood can be a very complicated, and hard to optimize function of $\boldsymbol{\theta}$.
- The Expectation–Maximization (EM) algorithm is a method for ML learning of parameters in latent varible models.
- Basic intuition of EM: iterate between inferring latent variables and fitting parameters.

[1] Examples of latent variable models: factor analysis, probabilistic PCA, ICA, mixture models, hidden Markov models, linear-Gaussian state-space models...

The Expectation Maximisation (EM) algorithm

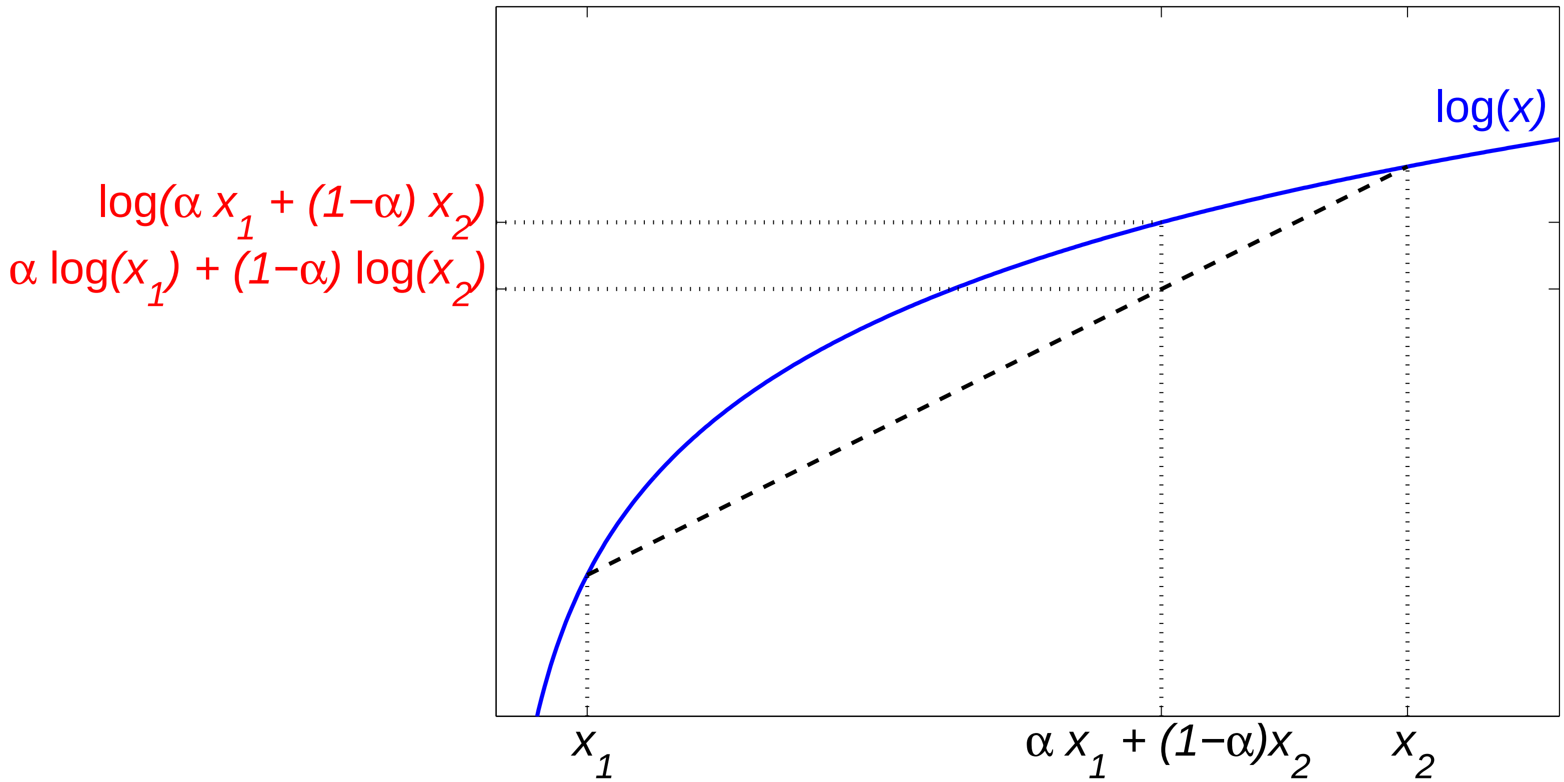
The EM algorithm finds a (local) maximum of a latent variable model likelihood. It starts from arbitrary values of the parameters, and iterates two steps:

E step: Fill in values of latent variables according to posterior given data.

M step: Maximise likelihood as if latent variables were not hidden.

- Useful in models where learning would be easy if hidden variables were, in fact, observed (e.g. FA turns into linear regression).
- Decomposes difficult problems into series of tractable steps.
- No learning rate.
- Framework lends itself to principled approximations.

Jensen's Inequality



For $\alpha_i \geq 0$, $\sum \alpha_i = 1$ and any $\{x_i > 0\}$

$$\log\left(\sum_i \alpha_i x_i\right) \geq \sum_i \alpha_i \log(x_i)$$

Equality if and only if $\alpha_i = 1$ for some i (and therefore all others are 0).

Lower Bounding the Log Likelihood

Observed data $\mathcal{D} = \{\mathbf{y}_n\}$; Latent variables $\mathcal{X} = \{\mathbf{x}_n\}$; Parameters $\boldsymbol{\theta}$.

Goal: Maximize the log likelihood (i.e. ML learning) wrt $\boldsymbol{\theta}$:

$$\mathcal{L}(\boldsymbol{\theta}) = \log P(\mathcal{D}|\boldsymbol{\theta}) = \log \int P(\mathcal{X}, \mathcal{D}|\boldsymbol{\theta}) d\mathcal{X},$$

Any distribution, $q(\mathcal{X})$, over the hidden variables can be used to obtain a lower bound on the log likelihood using Jensen's inequality:

$$\mathcal{L}(\boldsymbol{\theta}) = \log \int q(\mathcal{X}) \frac{P(\mathcal{X}, \mathcal{D}|\boldsymbol{\theta})}{q(\mathcal{X})} \, d\mathcal{X} \geq \int q(\mathcal{X}) \log \frac{P(\mathcal{X}, \mathcal{D}|\boldsymbol{\theta})}{q(\mathcal{X})} \, d\mathcal{X} \stackrel{\text{def}}{=} \mathcal{F}(q, \boldsymbol{\theta}).$$

$$\begin{aligned} \mathcal{F}(q, \boldsymbol{\theta}) &= \int q(\mathcal{X}) \log P(\mathcal{X}, \mathcal{D}|\boldsymbol{\theta}) \, d\mathcal{X} - \int q(\mathcal{X}) \log q(\mathcal{X}) \, d\mathcal{X} \\ &= \int q(\mathcal{X}) \log P(\mathcal{X}, \mathcal{D}|\boldsymbol{\theta}) \, d\mathcal{X} + \mathbf{H}[q], \end{aligned}$$

where $\mathbf{H}[q]$ is the entropy of $q(\mathcal{X})$.
So:

$$\mathcal{F}(q, \boldsymbol{\theta}) = \langle \log P(\mathcal{X}, \mathcal{D}|\boldsymbol{\theta}) \rangle_{q(\mathcal{X})} + \mathbf{H}[q] \leq \mathcal{L}(\boldsymbol{\theta})$$

Notation and Terminology

$$\langle f(x) \rangle_{p(x)} \stackrel{\text{def}}{=} \int f(x) p(x) dx$$

$$\mathbf{H}[p] = - \int p(x) \log p(x) dx$$

Links between statistical physics and machine learning:

- negative log probabilities correspond to the "energy" of a system
- $- \langle \log P(\mathcal{X}, \mathcal{D} | \boldsymbol{\theta}) \rangle_{q(\mathcal{X})}$ is the average energy
- $\mathcal{F}(q, \boldsymbol{\theta})$ is the negative free energy

Physical systems tend to converge to a distribution of states with low free energy $\approx$ Learning systems should find a distribution of parameters and hidden variables with low free energy

The E and M steps of EM

The lower bound on the log likelihood is given by:

$$\mathcal{F}(q, \boldsymbol{\theta}) = \langle \log P(\mathcal{X}, \mathcal{D}|\boldsymbol{\theta}) \rangle_{q(\mathcal{X})} + \mathbf{H}[q],$$

EM alternates between:

E step: optimize $\mathcal{F}(q, \boldsymbol{\theta})$ wrt distribution over hidden variables holding params fixed:

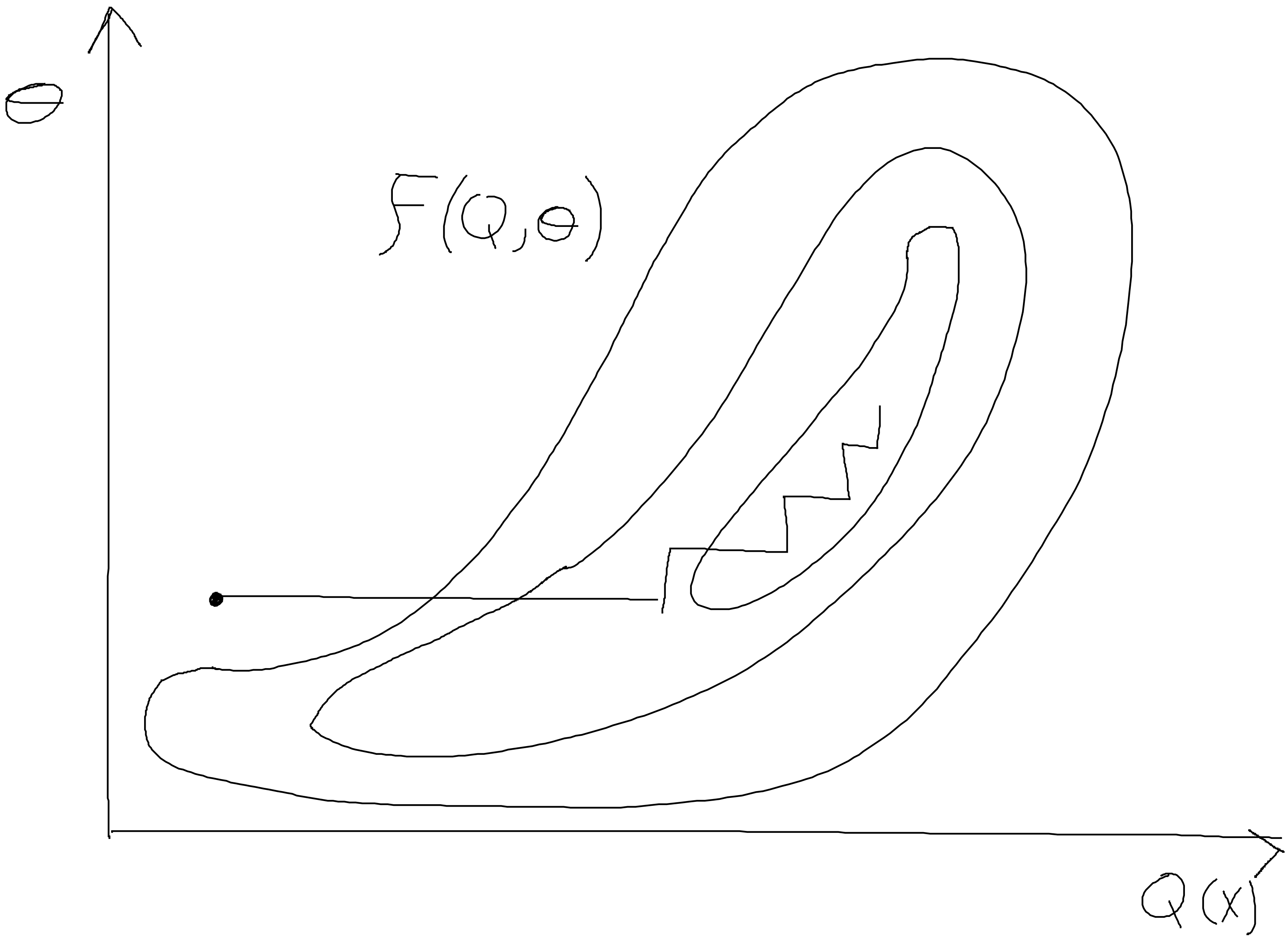
$$q^{(k)}(\mathcal{X}) := \underset{q(\mathcal{X})}{\operatorname{argmax}} \ \mathcal{F}\big(q(\mathcal{X}), \boldsymbol{\theta}^{(k-1)}\big).$$

M step: maximize $\mathcal{F}(q, \boldsymbol{\theta})$ wrt parameters holding hidden distribution fixed:

$$\boldsymbol{\theta}^{(k)} := \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \ \mathcal{F}\big(q^{(k)}(\mathcal{X}), \boldsymbol{\theta}\big) = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \ \langle \log P(\mathcal{X}, \mathcal{D}|\boldsymbol{\theta}) \rangle_{q^{(k)}(\mathcal{X})}$$

The second equality comes from fact that entropy of $q(\mathcal{X})$ does not depend directly on $\boldsymbol{\theta}$.

EM as Coordinate Ascent in $\mathcal{F}$



The E Step

The free energy can be re-written

$$\begin{aligned}
\mathcal{F}(q, \boldsymbol{\theta}) &= \int q(\mathcal{X}) \log \frac{P(\mathcal{X}, \mathcal{D}|\boldsymbol{\theta})}{q(\mathcal{X})} \, d\mathcal{X} \\
&= \int q(\mathcal{X}) \log \frac{P(\mathcal{X}|\mathcal{D}, \boldsymbol{\theta}) P(\mathcal{D}|\boldsymbol{\theta})}{q(\mathcal{X})} \, d\mathcal{X} \\
&= \int q(\mathcal{X}) \log P(\mathcal{D}|\boldsymbol{\theta}) \, d\mathcal{X} + \int q(\mathcal{X}) \log \frac{P(\mathcal{X}|\mathcal{D}, \boldsymbol{\theta})}{q(\mathcal{X})} \, d\mathcal{X} \\
&= \mathcal{L}(\boldsymbol{\theta}) - \textbf{KL}[q(\mathcal{X}) \| P(\mathcal{X}|\mathcal{D}, \boldsymbol{\theta})]
\end{aligned}$$

The second term is the Kullback-Leibler divergence.

This means that, for fixed $\boldsymbol{\theta}$, $\mathcal{F}$ is bounded above by $\mathcal{L}$, and achieves that bound when $\textbf{KL}[q(\mathcal{X}) \| P(\mathcal{X}|\mathcal{D}, \boldsymbol{\theta})] = 0$.

But $\textbf{KL}[q \| p]$ is zero if and only if $q = p$.

So, the E step simply sets

$$q^{(k)}(\mathcal{X}) = P(\mathcal{X}|\mathcal{D}, \boldsymbol{\theta}^{(k-1)})$$

and, after an E step, the free energy equals the likelihood.

The M Step

$$\mathcal{F}(q, \boldsymbol{\theta}) = \int q(\mathcal{X}) \log \frac{P(\mathcal{X}, \mathcal{D}|\boldsymbol{\theta})}{q(\mathcal{X})} \, d\mathcal{X}$$

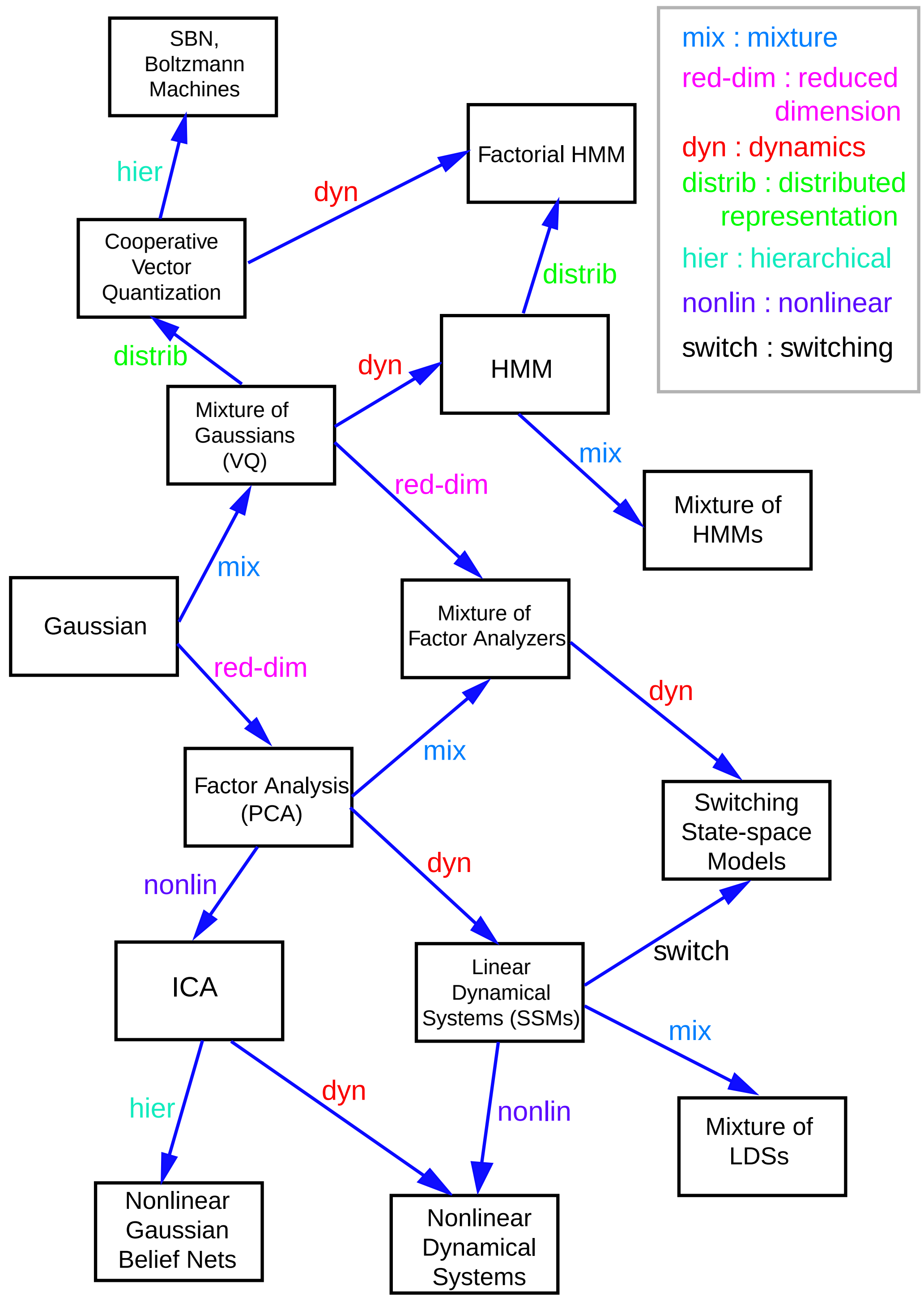
M step: maximize $\mathcal{F}(q, \boldsymbol{\theta})$ wrt parameters holding hidden distribution fixed:

$$\begin{aligned} \boldsymbol{\theta}^{(k)} &:= \operatorname*{argmax}_{\boldsymbol{\theta}} \; \mathcal{F}\big(q^{(k)}(\mathcal{X}), \boldsymbol{\theta}\big) && (1) \\ &= \operatorname*{argmax}_{\boldsymbol{\theta}} \; \int q^{(k)}(\mathcal{X}) \, \log P(\mathcal{X}, \mathcal{D}|\boldsymbol{\theta}) \, d\mathcal{X} && (2) \end{aligned}$$

The second equality comes from fact that entropy of $q(\mathcal{X})$ does not depend directly on $\boldsymbol{\theta}$.

The specific form of the M step depends on the model.

Often the maximum wrt $\boldsymbol{\theta}$ can be found analytically.


SBN, Boltzmann Machines
Factorial HMM
mix : mixture
red-dim : reduced dimension
dyn : dynamics
distrib : distributed representation
hier : hierarchical
nonlin : nonlinear
switch : switching
hier
dyn
Cooperative Vector Quantization
distrib
distrib
HMM
dyn
Mixture of Gaussians (VQ)
mix
red-dim
Mixture of HMMs
mix
Gaussian
Mixture of Factor Analyzers
red-dim
dyn
mix
Factor Analysis (PCA)
Switching State-space Models
dyn
nonlin
switch
ICA
Linear Dynamical Systems (SSMs)
mix
dyn
hier
nonlin
Mixture of LDSs
Nonlinear Gaussian Belief Nets
Nonlinear Dynamical Systems