EECS 491
Probabilistic Graphical Models

Spring 2019

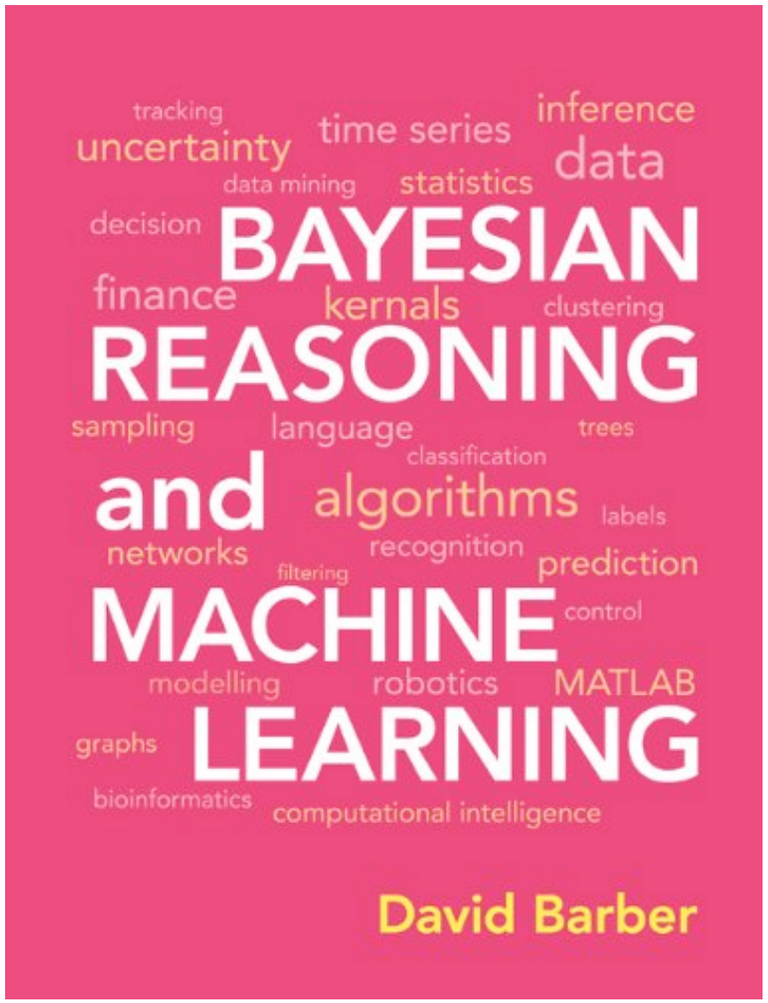
Introduction and Overview

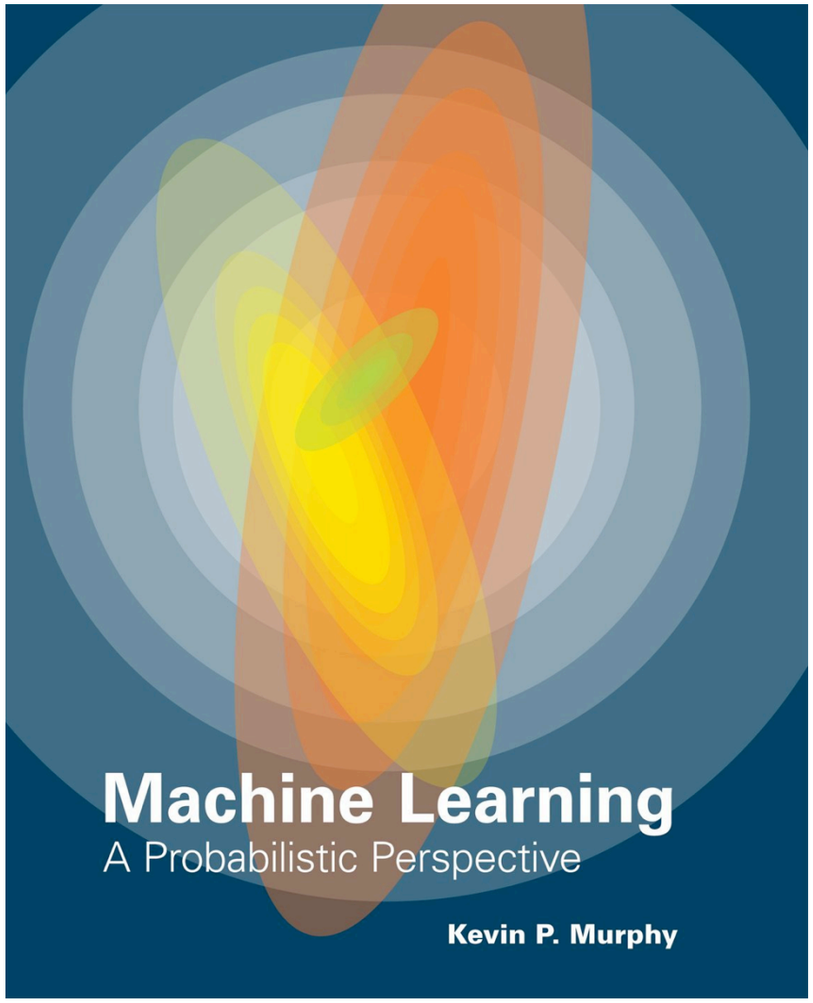
Course information

- Course ID: EECS 491
- Course Title: Probabilistic Graphical Models
- Credits: 3
- Instructor:
  Mike Lewicki, Professor
  Electrical Engineering and Computer Science
  Case Western Reserve University
  office: Olin 508
  office hours: MW 2:00-3:00 (next door in Olin 314 for next few weeks)
  email: msl88@case.edu
- Classroom and meeting times for Spring 2019:
  Olin 313 MW 12:45 - 2:00
- Web: https//canvas.case.edu
- teaching assistants: TBA

Course textbooks (no, we're not covering everything)

- *Bayesian Reasoning and Machine Learning* by Barber, Cambridge 2012
  Available online: http://www.cs.ucl.ac.uk/staff/d.barber/brml/
- Also recommended:
  Machine Learning: A Probabilistic Perspective by Murphy, MIT Press, 2012





Course goals

- Learn general methods and models representing probabilistic information
- Learn a range of inference and learning methods for these models
- Implement and apply these techniques to solve practical problems
- Use notebooks to explore, explain, and illustrate
- Concept areas:
  - probabilistic reasoning
  - belief networks and graphical models
  - inference techniques
  - learning in probabilistic models
  - clustering, PCA, and dimensionality reduction
  - linear probabilistic models
  - dynamic models

Prerequisites: consider carefully

- EECS 233, EECS 391, MATH 408, MATH 207
- familiar with basic concepts in Artificial Intelligence and Machine Learning (EECS 391 and/or EECS 440)
- programming components involve implementing and using algorithms (EECS 233, but programs are short exercises)
- more math than programming
- languages:
  - recommend: jupyter notebook (python or julia but not R) ↦ *must submit* pdf
  - also: matlab + latex ↦ *must submit* pdf (with notebook, code, *small* data)
- math:
  - will make extensive use of probability theory (MATH 408)
  - univariate and multivariate calculus (e.g. MATH 121,122 and 223)
  - linear algebra (MATH 201 or 207)
- If you are missing substantial background, please see me before continuing course.

Assignments and Grading

- No exams. No final.
- only assignments and course project
- course emphasizes creative exploration from basic starting points
- preferred form for assignments will be notebooks (default: python in jupyter)
  can also do code + latex + pdf.
- The notebooks you will create for each exercise in an assignment will consist of
  - conceptual background (text)
  - mathematical background (equations, must use latex)
  - code (e,g., python)
  - graphs, plots, or tables of results

Grading

- seven assignments (everyone)
- final project (grad students)
- lowest assignment score will be dropped
- undergrads:
  - assignments are weighted evenly
  - each is 100/6 = 16 ⅔% of total grade.
- grad students:
  - assignments: 78% (each is 13%, still dropping lowest)
  - project: 22%

Collaboration and Cheating

- Collaboration to discuss the assignments and problems is encouraged.
- You *must* generate and write up the solutions yourself
- We expect that you can write on the explanation and code yourself, starting from a blank page without reference.
- Anything less is cheating.
- You are not allowed to copy code.

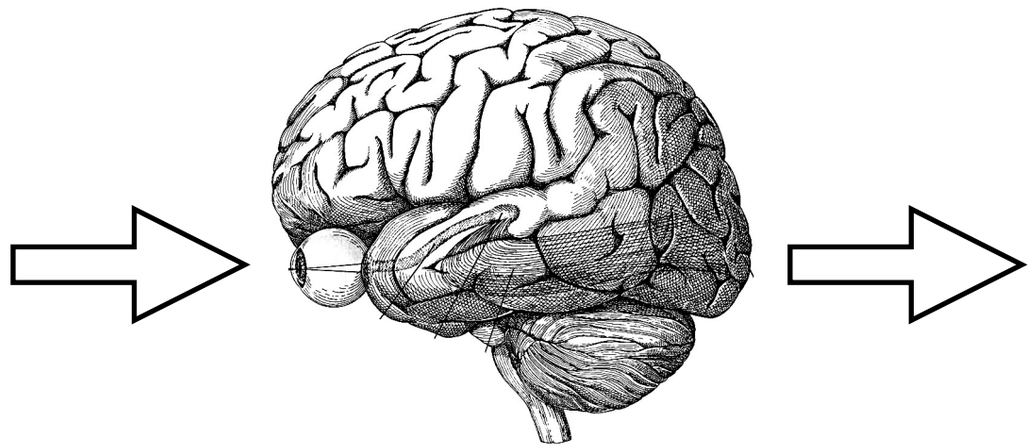
(see schedule)

How this is different from 391 and 440?

- EECS 391 is designed as a introductory course that covers a broad range of topics (e.g. Russell and Norvig):
  - problem solving by search
  - game theory, game trees, alpha-beta pruning
  - constraint satisfaction
  - logic, decision making, Bayes nets
  - reinforcement learning (Prof. Ray's 391)
- EECS 440
  - decision trees, classification
  - neural networks, support vector machines
  - learning theory, relational learning
- EECS 491 expands on subset of these topics: reasoning and probabilistic modeling (but not machine learning, which is EECS 440)

Reasoning

Reasoning can be thought of as constructing an accurate world model.



perception cognition action

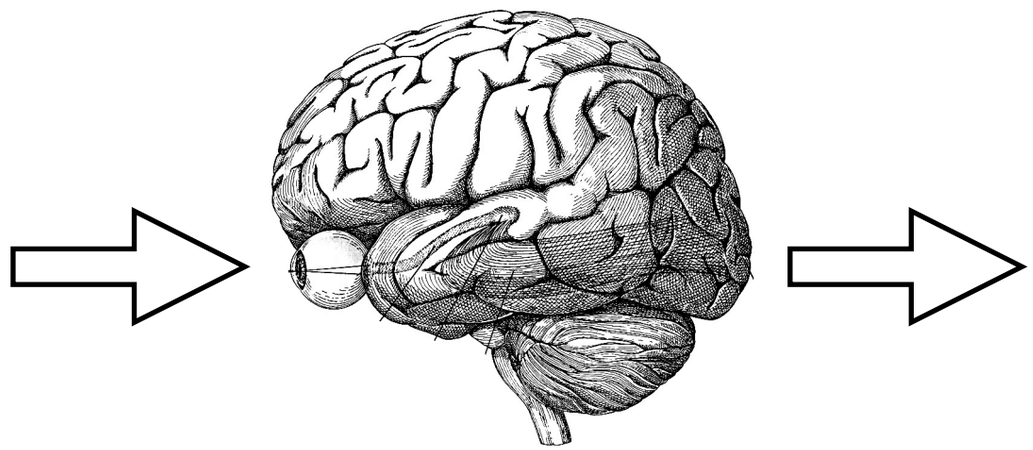
- facts
- observations
- "wet ground"

- logical consequences
- inferences
- "it rained" or "sprinkler" ?

Rational inference:
What can be logically inferred give available information?

Reasoning with uncertain information

Most facts are not concrete and are not known with certainty.



perception cognition action

- facts
- observations
- “fever”
- “aches”
- platelet count=N

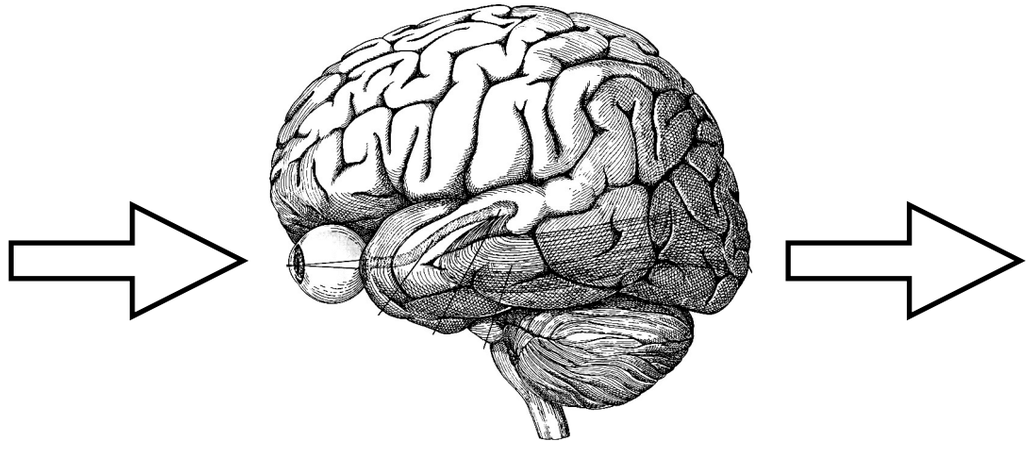
- inferences
- What disease?
- What causes?

Probabilistic inference:
How do we give the proper weight to each observation?

What is ideal?

Learning

What if your world is changing? How do we maintain an accurate model?



perception cognition action

- chess board
- maze
- text
- object
- room
- sound
- visual scene

Learning:
adapt internal representation so that it is as accurate as possible.

Can also adapt our models of other agents.

The process of probabilistic inference

1. *define* model of problem
2. *derive* posterior distributions and estimators
3. e*stimate* parameters from data
4. *evaluate* model accuracy

		Probabilistic Reasoning (Barber Ch. 1)		
2	Wed, Jan 16	**Probabilistic Reasoning** - basic probability review, reasoning with Bayes' rule	B.1.1-2; M. 2.1-2	A1 out
	Mon, Jan 21	*(No class - Martin Luther King Jr. Holiday)*		
3	Wed, Jan 23	**Reasoning with Continuous Variables** - probability distribution functions, prior, likelihood, and posterior, model-based inference	B.1.3, B.8, B. 9.1; M.2.1-4	
4	Mon, Jan 28	**More Probabilistic Reasoning** - more complex examples	B.1.1-2; M. 2.1-2	

Inference using Bayes' rule

Example 1.2 (Hamburgers)**.** Consider the following fictitious scientific information: Doctors find that people with Kreuzfeld-Jacob disease (KJ) almost invariably ate hamburgers, thus $p(Hamburger\ Eater|KJ\ ) = 0.9$. The probability of an individual having KJ is currently rather low, about one in 100,000.

1. Assuming eating lots of hamburgers is rather widespread, say $p(Hamburger\ Eater) = 0.5$, what is the probability that a hamburger eater will have Kreuzfeld-Jacob disease?

   This may be computed as

$$p(KJ\ |Hamburger\ Eater) = \frac{p(Hamburger\ Eater, KJ\ )}{p(Hamburger\ Eater)} = \frac{p(Hamburger\ Eater|KJ\ )p(KJ\ )}{p(Hamburger\ Eater)} \tag{1.2.1}$$

$$= \frac{\frac{9}{10} \times \frac{1}{100000}}{\frac{1}{2}} = 1.8 \times 10^{-5} \tag{1.2.2}$$

2. If the fraction of people eating hamburgers was rather small, $p(Hamburger\ Eater) = 0.001$, what is the probability that a regular hamburger eater will have Kreuzfeld-Jacob disease? Repeating the above calculation, this is given by

$$\frac{\frac{9}{10} \times \frac{1}{100000}}{\frac{1}{1000}} \approx 1/100 \tag{1.2.3}$$

   This is much higher than in scenario (1) since here we can be more sure that eating hamburgers is related to the illness.

Bayesian inference of continuous parameters

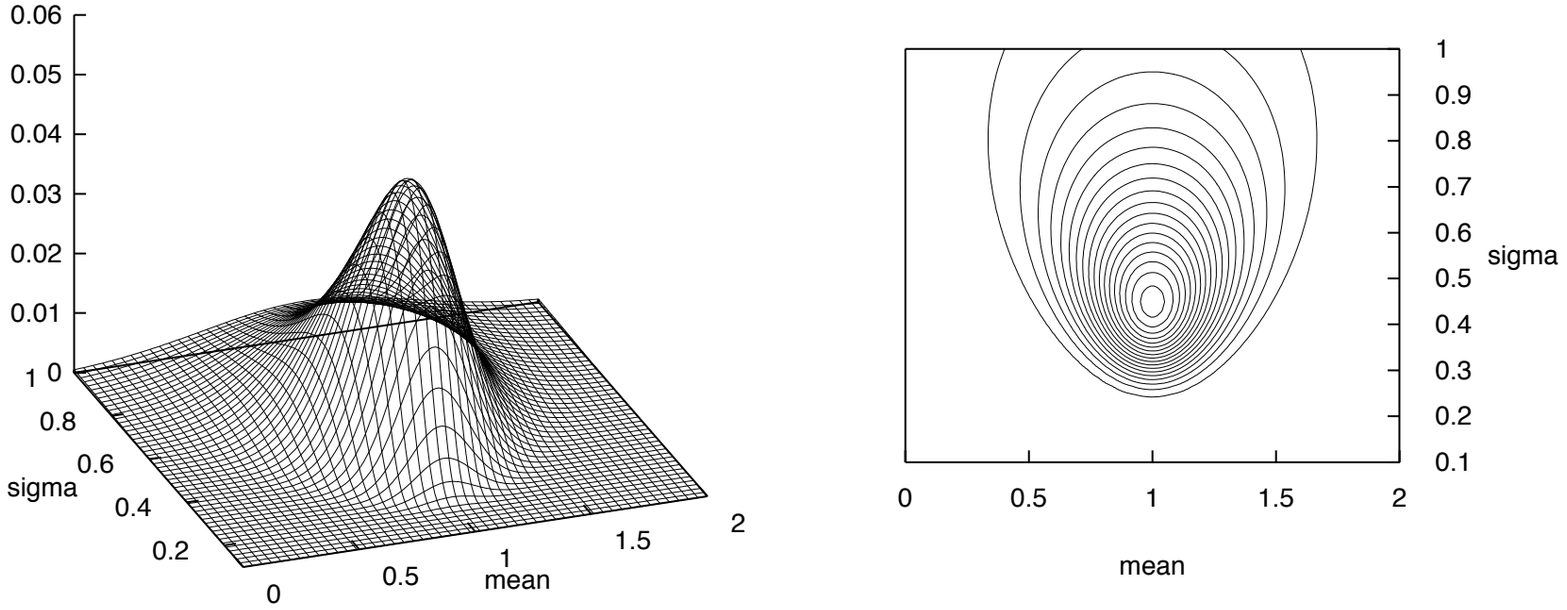


Figure 21.5. The likelihood function for the parameters of a Gaussian distribution.
Surface plot and contour plot of the log likelihood as a function of μ and σ. The data set of $N = 5$ points had mean $\bar{x} = 1.0$ and $S^2 = \sum(x - \bar{x})^2 = 1.0$.

from *Information Theory, Inference, and Learning Algorithms*
by DJC Mackay
http://www.inference.org.uk/itila/book.html

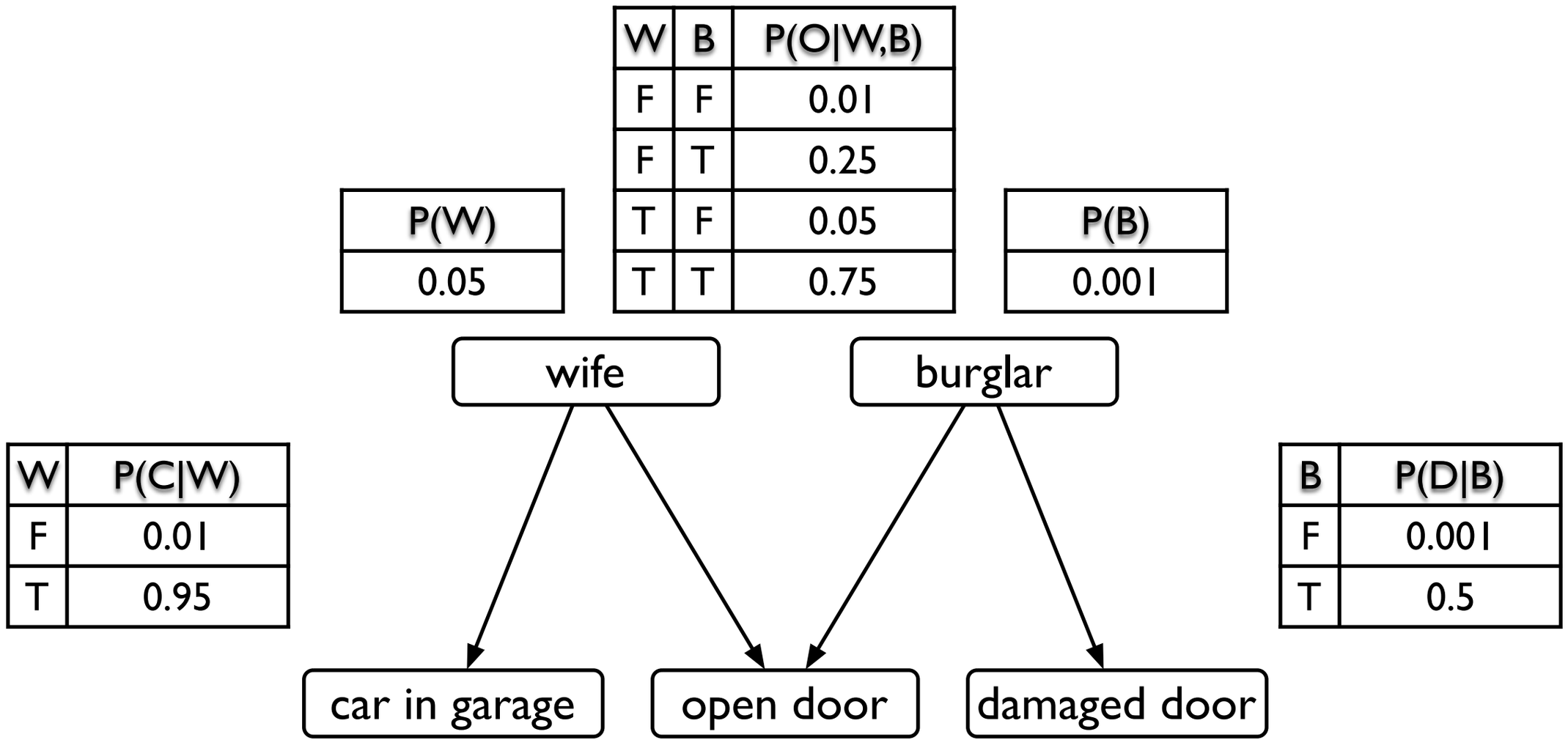
		Belief Networks & Graphical Models (Barber Ch. 3&4)		
5	Wed, Jan 30	**Belief Networks 1** - representing probabilistic relations with directed acyclic graphs, simple belief networks, modeling dependencies	B.3.1-2, (graph background B. 2); M.10.1-2	A1 due; A2 out
6	Mon, Feb 4	**Belief Networks 2** - inference, uncertainty and unreliability in evidence, independence, causality	B.3.1-4; M. 10.5	
7	Wed, Feb 6	**Graphical Models 1** - Markov random fields, expressiveness of graphical models, factor graphs	B.4; M.19.1-4	
8	Mon, Feb 11	**Graphical Models 2** - undirected graphical models, independence relationships, energy functions, Monte Carlo inference in MRFs	B.4; M10, 23, 24	

Bayesian belief networks

- P(o,w,¬b,c,¬d) = P(o|w,¬b)P(c|w)P(¬d|¬b)P(w)P(¬b)

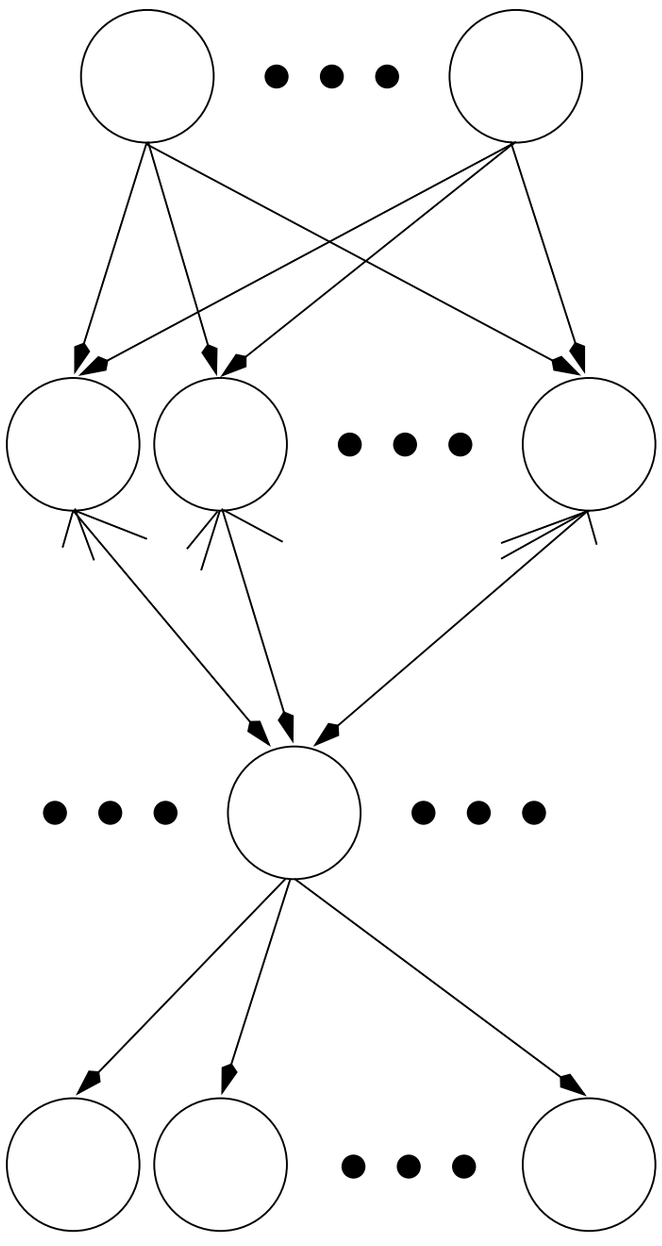
  = 0.05 × 0.95 × 0.999 × 0.05 × 0.999 = 0.0024

- This is essentially the probability that my wife is home and leaves the door open.



Hierarchical Statistical Models

A Bayesian belief network:



The joint probability of binary states is

$$P(\mathbf{S}|\mathbf{W}) = \prod_i P(S_i|\mathrm{pa}(S_i), \mathbf{W})$$

The probability S_i depends only on its parents:

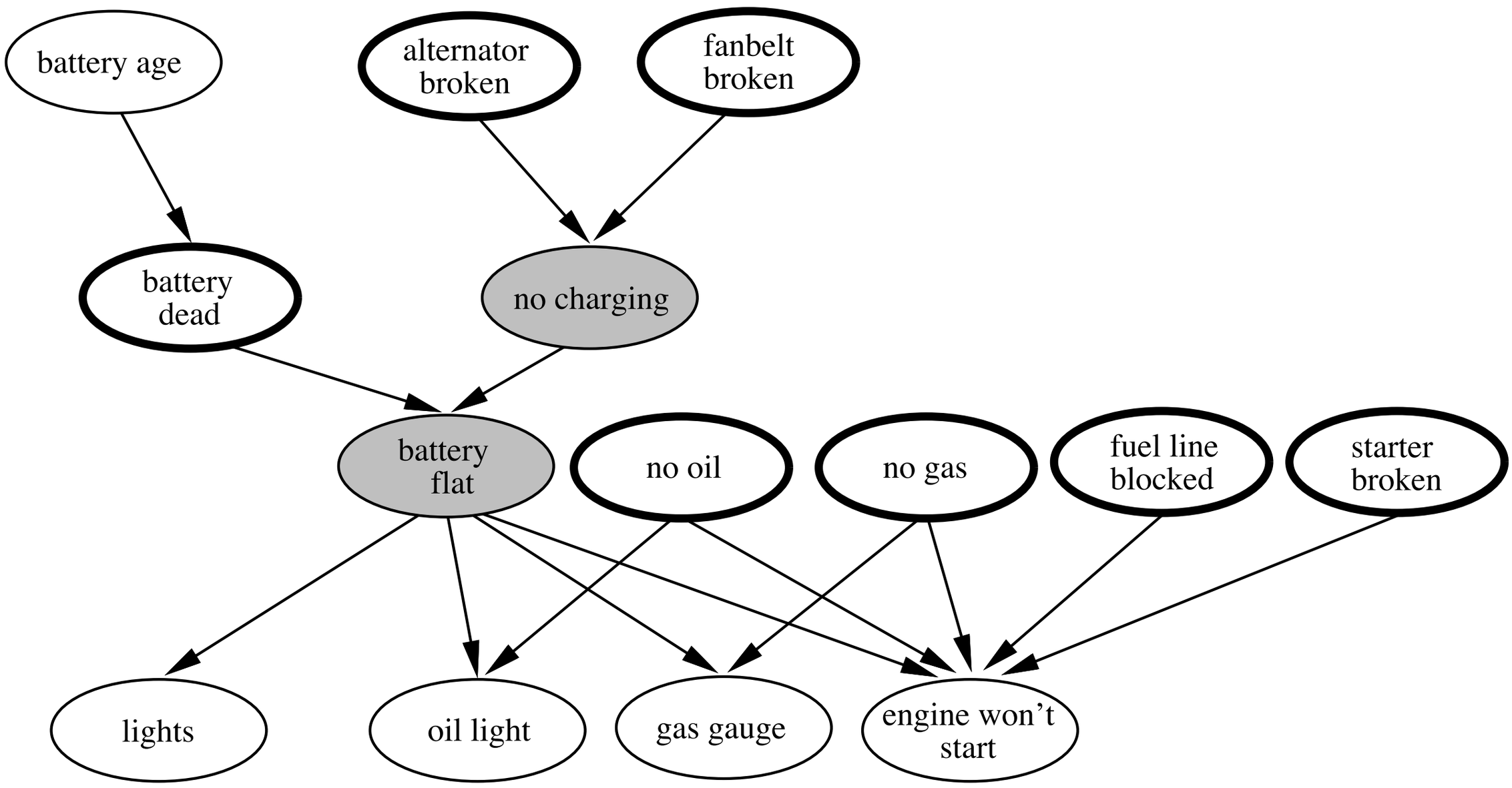
$$P(S_i|\mathrm{pa}(S_i), \mathbf{W}) = \begin{cases} h(\sum_j S_j w_{ji}) & \text{if } S_i = 1 \\ 1 - h(\sum_j S_j w_{ji}) & \text{if } S_i = 0 \end{cases}$$

The function h specifies how causes are combined, $h(u) = 1 - \exp(-u)$, $u > 0$.

Main points:

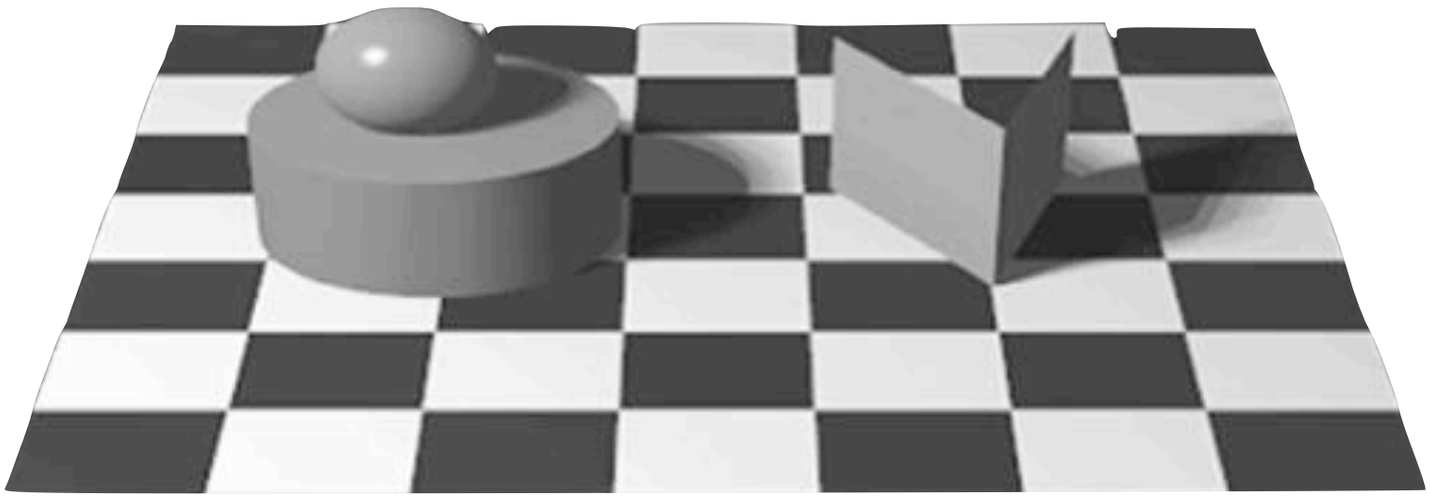
- hierarchical structure allows model to form high order representations
- upper states are priors for lower states
- weights encode higher order features

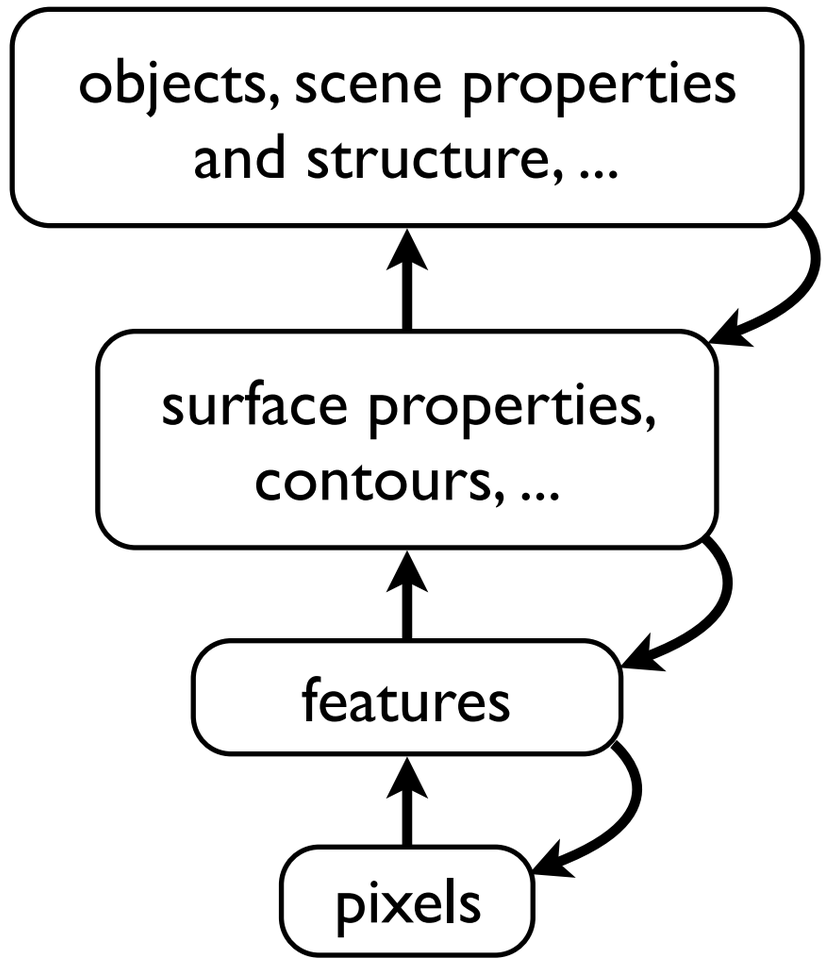
A more complex belief network using noisy-OR nodes



Motivation: learn hierarchical, context dependent representations

- many real-world patterns are hierarchical in structure
- interpretation of patterns depends on context
- essential for complex recognition tasks

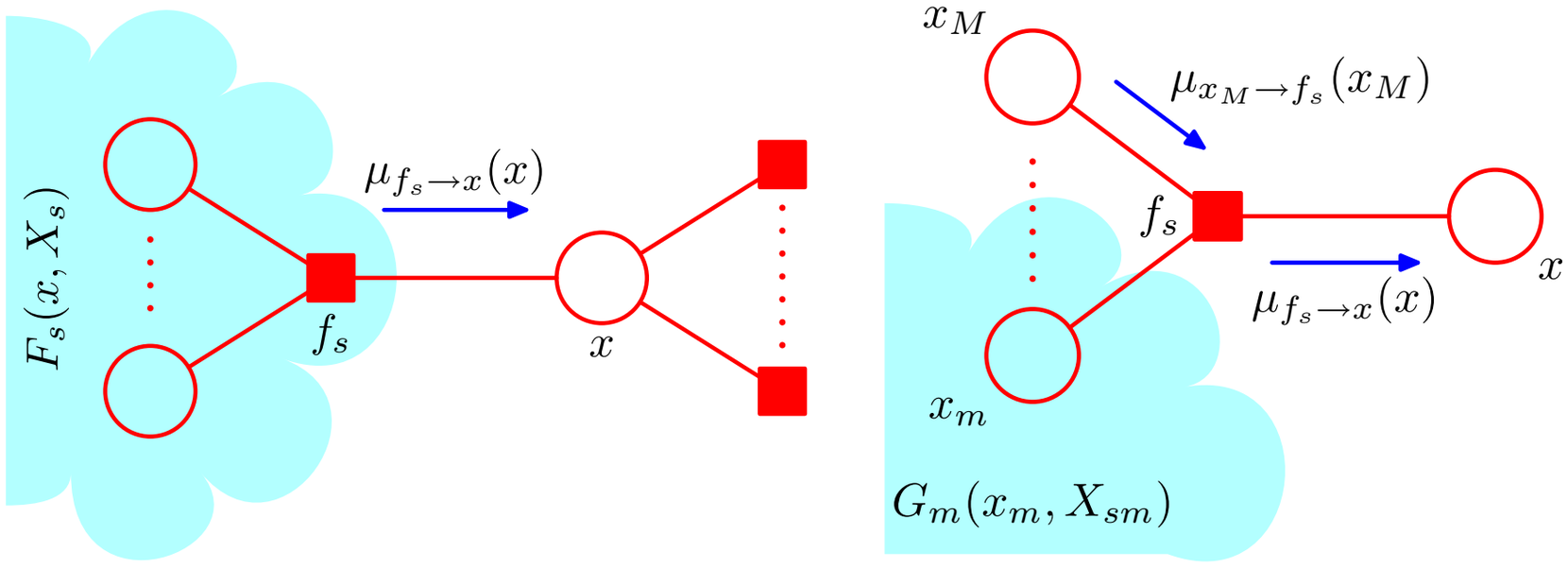




		Inference Techniques (Barber Ch. 5)		
9	Wed, Feb 13	**Introduction to Inference** - variable-elimination, Monte-Carlo methods, junction trees, loops	B.5 & 6; M.20, Bishop Ch.8	A2 due; A3 out
10	Mon, Feb 18	**Efficient Inference 1** - message passing, sum-product algorithm on factor graphs	B.5.1-2; M.20; Bishop Ch.8	
11	Wed, Feb 20	**Efficient Inference 2** - derivation of sum-product algorithm, expectation propagation, message passing for continuous distributions	B.5.2-4; M.20; Bishop Ch.8	

Messages in the sum-product algorithm

- $\mu_{f_s \to x}(x)$ represents a message going from a factor node f_s to variable node x



- Like before the equation is recursive (derivation omitted):

$$\mu_{f_s \to x}(x) = \sum_{x_1} \cdots \sum_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m \in \mathrm{ne}(f_s) \setminus x} \left[\sum_{X_{xm}} G_m(x_m, X_{sm}) \right]$$

$$= \sum_{x_1} \cdots \sum_{x_M} f_s(x, x_1, \ldots, x_M) \prod_{m \in \mathrm{ne}(f_s) \setminus x} \mu_{x_m \to f_s}(x_m) \qquad (8.66)$$

Derivation of algorithms

Goal:

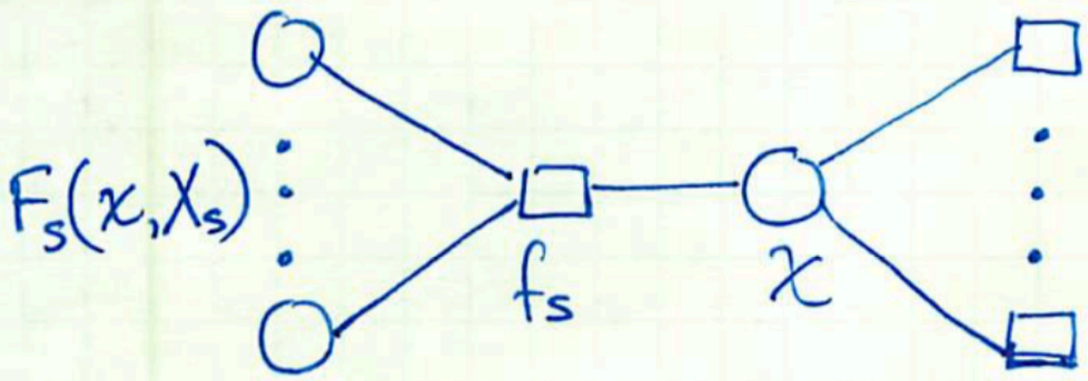
1) efficient, exact inference alg for computing marginals
2) share computations when computing several marginals

Again, start with.

$$p(\underline{x}) = \prod_s f_s(\underline{x}_s)$$

$$p(x_i) = \sum_{\underline{x} \setminus x_i} p(\underline{x}) = \sum_{\underline{x} \setminus x_i} \prod_s f_s(x_s)$$

$\underline{x} \setminus x_i$ → all x_j except $j = i$



Can partition the factors of the joint distr. into groups:

$$p(\underline{x}) = \prod_{s \in ne(x)} F_s(x, X_s)$$

$ne(x)$ = set of factor nodes that are neighbors of x

		Learning in Probabilistic Models (Barber Ch. 8-11)		
12	Mon, Feb 25	**Learning in Probabilistic Models** - represented data, statistics for learning, common probability distributions, learning distributions, maximum likelihood, Gaussian models	B.8.1-3,6-7; M.3.1	A4 out
13	Wed, Feb 27	**Representation and Probabilistic Models** - probabilistic models as belief networks, continuous parameters, training belief networks.	B.9.1-3	A3 due
14	Mon, Mar 4	**More on Generative Models** - Bayesian concept learning, Dirichlet multinomial model, bag of words model, Naïve Bayes, Bayesian model selection	drawn from B. 9, 10, 12; M. 3.1-4	
15	Wed, Mar 6	**Learning with Hidden Variables** - hierarchical models, missing data, expectation maximization (EM), EM.for belief nets, variational Bayes, gradient methods, deep belief nets	B. 9.4, 11.1-2; M.10.4	
	Mar 12-16	*Spring break - no class*		

Bayes theorem for Gaussian variables

- Can we derive

$$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{x})}$$

where **y** is a linear function of **x**:

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1})$$
$$p(\mathbf{y}|\mathbf{x}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\mathbf{x} + \mathbf{b}, \mathbf{L}^{-1})$$

- This is an example of a *linear Gaussian model.*

- Since the prior and likelihood are Gaussian, the posterior distribution is also Gaussian:

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{A}\boldsymbol{\mu} + \mathbf{b}, \mathbf{L}^{-1} + \mathbf{A}\boldsymbol{\Lambda}^{-1}\mathbf{A}^{\mathrm{T}})$$
$$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{x}|\boldsymbol{\Sigma}\{\mathbf{A}^{\mathrm{T}}\mathbf{L}(\mathbf{y} - \mathbf{b}) + \boldsymbol{\Lambda}\boldsymbol{\mu}\}, \boldsymbol{\Sigma})$$

where

$$\boldsymbol{\Sigma} = (\boldsymbol{\Lambda} + \mathbf{A}^{\mathrm{T}}\mathbf{L}\mathbf{A})^{-1}$$

Hierarchical linear Gaussian graphical models

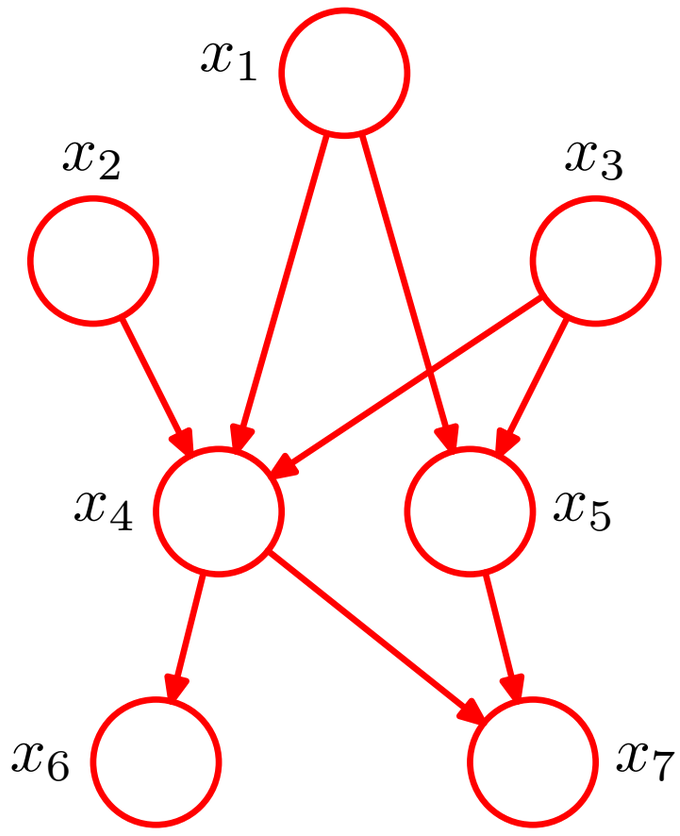
- First consider the 1-D case:

$$p(x_i|\mathrm{pa}(x_i) = j_p) = \mathcal{N}\left(x_i \middle| \sum_{j_p} w_{ij} x_j + b_i, v_i\right)$$

- x_i is a linear function of it's parents, which are also Gaussian random variables.
- More generally, we can let the node variables be vectors:

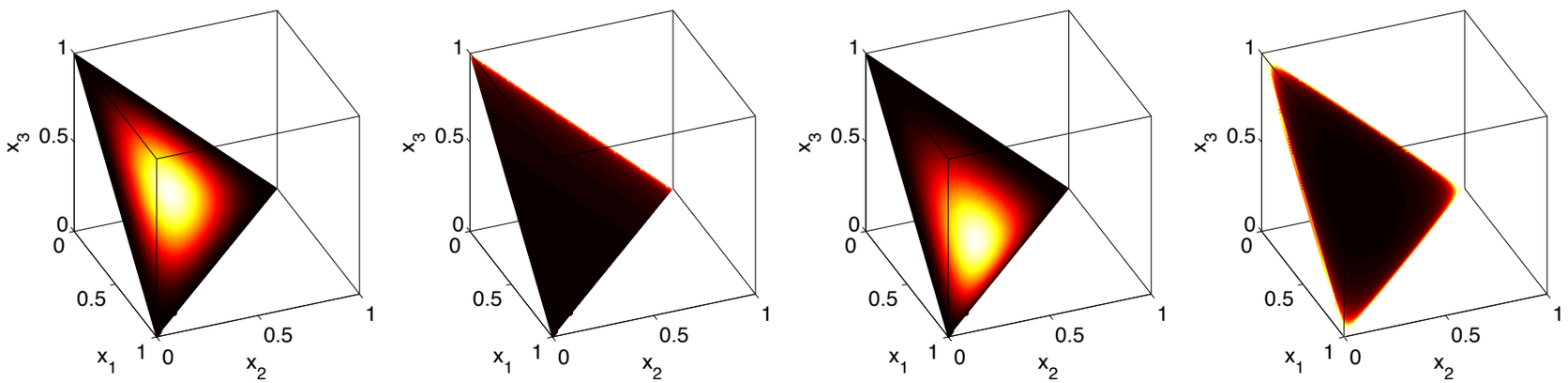
$$p(\mathbf{x}_i|\mathrm{pa}(\mathbf{x}_i) = j_p) = \mathcal{N}\left(\mathbf{x}_i \middle| \sum_{j_p} \mathbf{w}_{ij} \mathbf{x}_j + \mathbf{b}_i, \mathbf{\Sigma}_i\right)$$

- Because the variables are Gaussian, the relevant quantities for inference can be computed analytically.



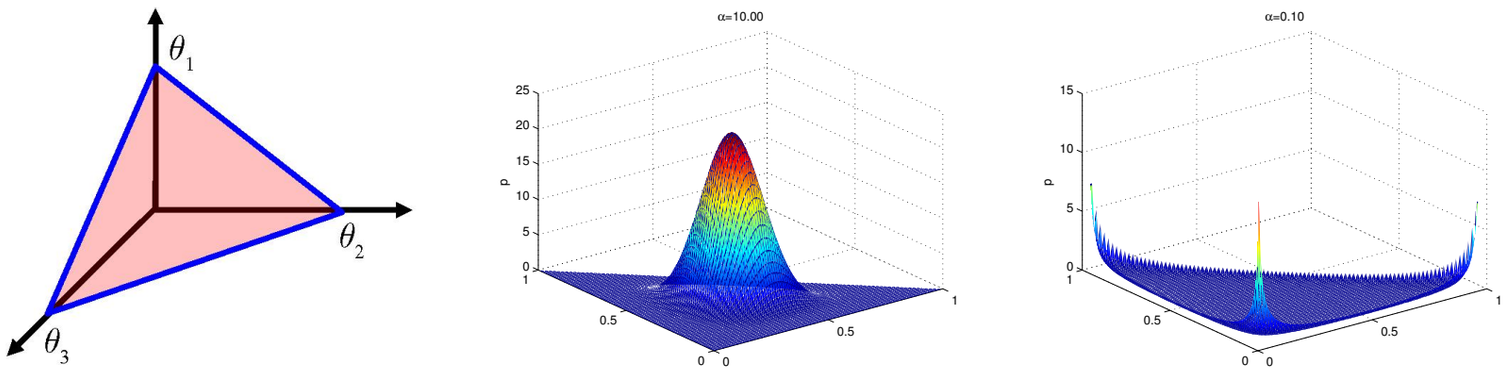
$\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_Q)$, $\alpha_i \geq 0$, $\sum_i \alpha_i = 1$:

$$p(\boldsymbol{\alpha}) = \frac{1}{Z(\mathbf{u})} \delta\left(\sum_{i=1}^{Q} \alpha_i - 1\right) \prod_{q=1}^{Q} \alpha_q^{u_q - 1} \mathbb{I}\left[\alpha_q \geq 0\right]$$



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

Figure 8.6: Dirichlet distribution with parameter (u_1, u_2, u_3) displayed on the simplex $x_1, x_2, x_3 \geq 0, x_1 + x_2 + x_3 = 1$. Black denotes low probability and white high probability. **(a)**: $(3, 3, 3)$ **(b)**: $(0.1, 1, 1)$. **(c)**: $(4, 3, 2)$. **(d)**: $(0.05, 0.05, 0.05)$.



(a) (b) (c)

$$p(\mathbf{x}) = \mathrm{Dir}(\mathbf{x}|\boldsymbol{\alpha}) = \frac{1}{B(\boldsymbol{\alpha})} \prod_{k=1}^{K} x_k^{\alpha_k - 1}$$

$$B(\boldsymbol{\alpha}) := \frac{\prod_{i=1}^{K} \Gamma(\alpha_i)}{\Gamma(\alpha_0)}$$

where $\alpha_0 = \sum_{k=1}^{K} \alpha_k$ and $B(\alpha_1, ..$
require $0 \leq x_k \leq 1$ and $\sum_{k=1}^{K} x_k = 1$

Figure 2.14: (a) The Dirichlet distribution when $K = 3$ defines a distribution over the simplex, which can be represented by the triangular surface. Points on this surface satisfy $0 \leq \theta_k \leq 1$ and $\sum_{k=1}^{3} \theta_k = 1$. Based on Figure 2.4 of [Bis06a]. (b) Plot of the Dirichlet density when $\alpha_k = 10$. (c) Plot of the Dirichlet density when $\alpha_k = 0.1$. (The comb-like structure on the edges is a plotting artefact.) Based on Figure 2.5 of [Bis06a]. Produced by `dirichlet3dPlot`. (See also `visDirichletGui` by Jonathan Huang.)

The Dirichlet-multinomial model

- The posterior predictive distribution is

$$
\begin{aligned}
p(X=j|\mathcal{D}) &= \int p(X=j|\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathcal{D})d\boldsymbol{\theta} = \int p(X=j|\theta_j)\left[\int p(\boldsymbol{\theta}_{-j},\theta_j|\mathcal{D})d\boldsymbol{\theta}_{-j}\right]d\theta_j \\
&= \int \theta_j p(\theta_j|\mathcal{D})d\theta_j = \mathbb{E}\left[\theta_j|\mathcal{D}\right] = \frac{\alpha_j + N_j}{N + \sum_k \alpha_k}
\end{aligned}
$$

- This gives the probability of each word given the observed data.
  - Mary had a little lamb, little lamb, little lamb
    Mary had a little lamb, its fleece was white as snow
- Assume $\alpha_i = 1$
- word: mary lamb little big fleece white black snow rain *unknown*

index:	1	2	3	4	5	6	7	8	9	10
count:	2	4	4	0	1	1	0	1	0	4
P(Xi=j|D)	3/27	5/27	5/27	0	2/27	2/27	0	2/27	0	5/27

		Clustering, PCA, and Dimensionality Reduction (Barber Ch. 12, 15, 20)		
16	Mon, Mar 18	**Gaussian Mixture Models** - clustering, nearest neighbor classification, k-means, latent variable models, mixture models, EM.for MMs	B.20; M.11	A5 out
17	Wed, Mar 20	**Bayesian Model Selection** - Occam's razor, Bayesian complexity penalization, Laplace approximation, Bayes information criterion, Bayes factors	B.12.1-5, M. 5.3, M.11.5	A4 due
18	Mon, Mar 25	**Principal Component Analysis** - dimensionality reduction, optimal linear reconstruction, whitening, latent semantic analysis	B.15.1-3; M. 12.2	
19	Wed, Mar 27	**Non-Linear Dimensionality Reduction** - ISOMAP, LLE, deep belief networks	handouts	

The EM algorithm

- EM stands for Expectation-Maximization, and involves two steps that are iterated. For the case of a Gaussian mixture model:

  1. E-step: Compute $p_{n,k} = p(c_k|x^{(n)}, \theta_{1:\mathrm{K}})$. Let $p_k = \sum_n p_{n,k}$
  2. M-step: Compute new mean, covariance, and class prior for each class:

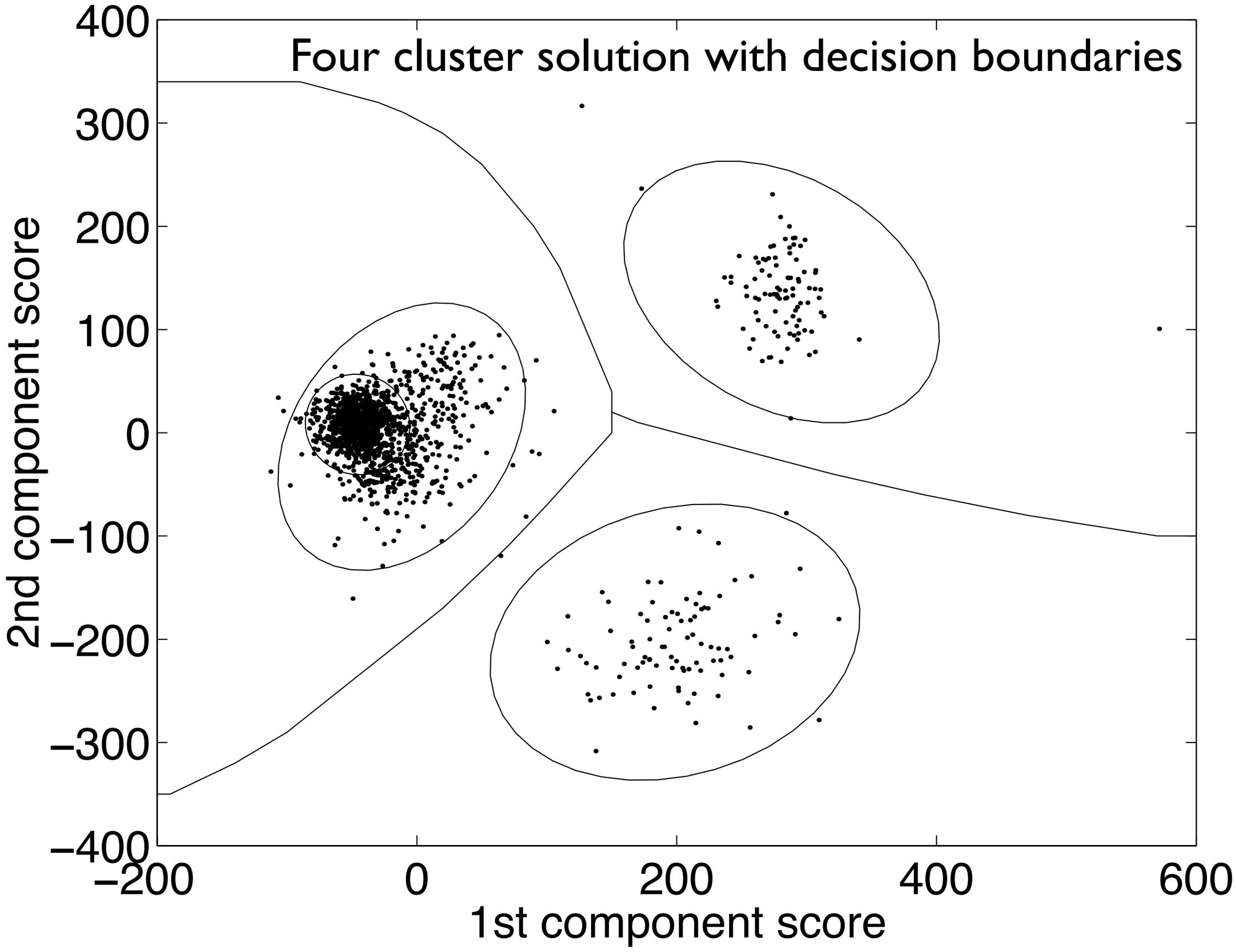
$$\mu_k \leftarrow \sum_n p_{n,k} x^{(n)} / p_k$$

$$\Sigma_k \leftarrow \sum_n p_{n,k} (x^{(n)} - \mu_k)(x^{(n)} - \mu_k)^T / p_k$$

$$p(c_k) \leftarrow p_k$$

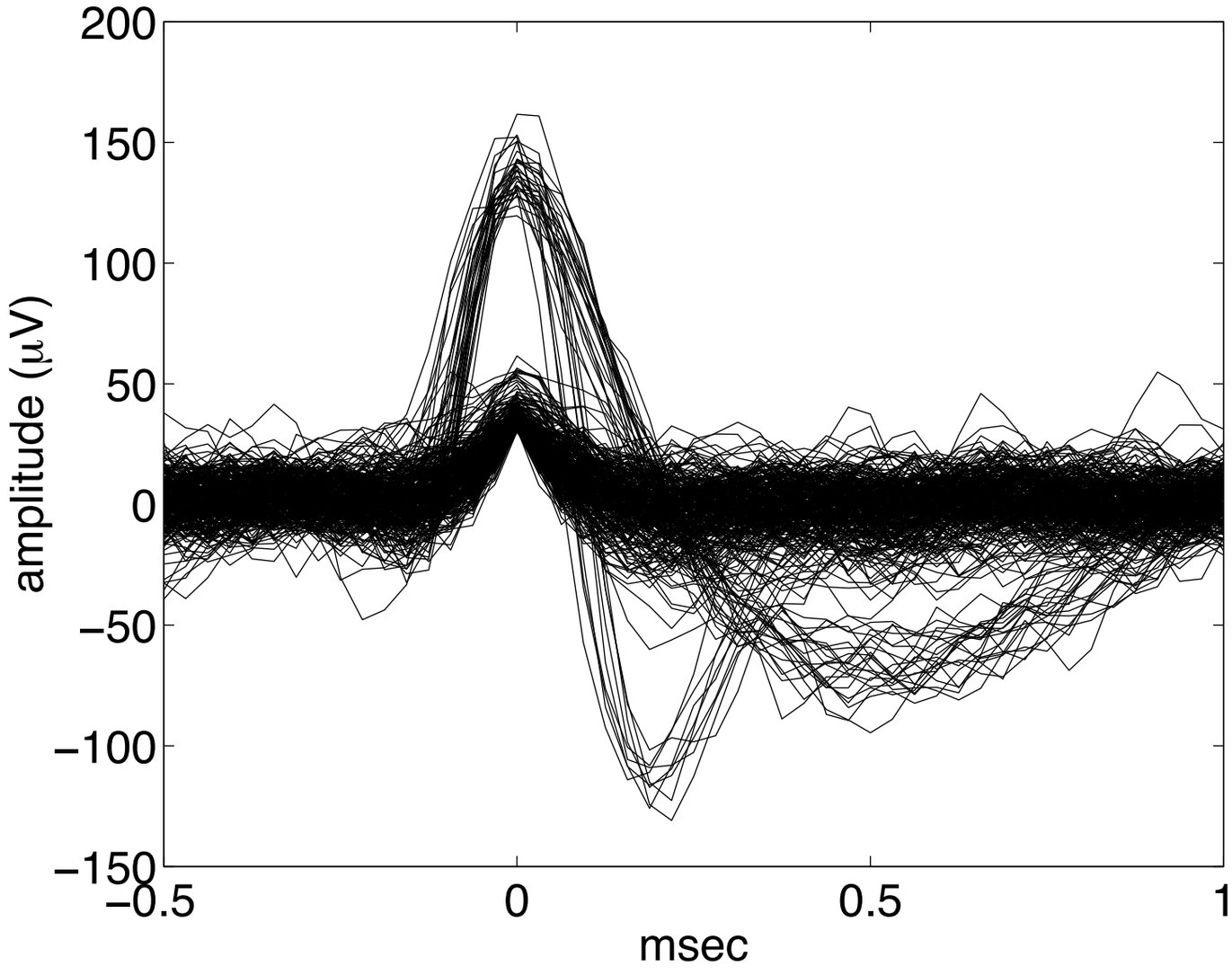
- This is just the sample mean and covariance, weighted by the class conditional probabilities $p_{n,k}$.

- Derived by solving setting log-likelihood gradient to zero (i.e. the maximum).


Four cluster solution with decision boundaries
400
300
200
100
0
−100
−200
−300
−400
2nd component score
−200
0
200
400
600
1st component score

Using principal components to characterize the data

- What are the data?

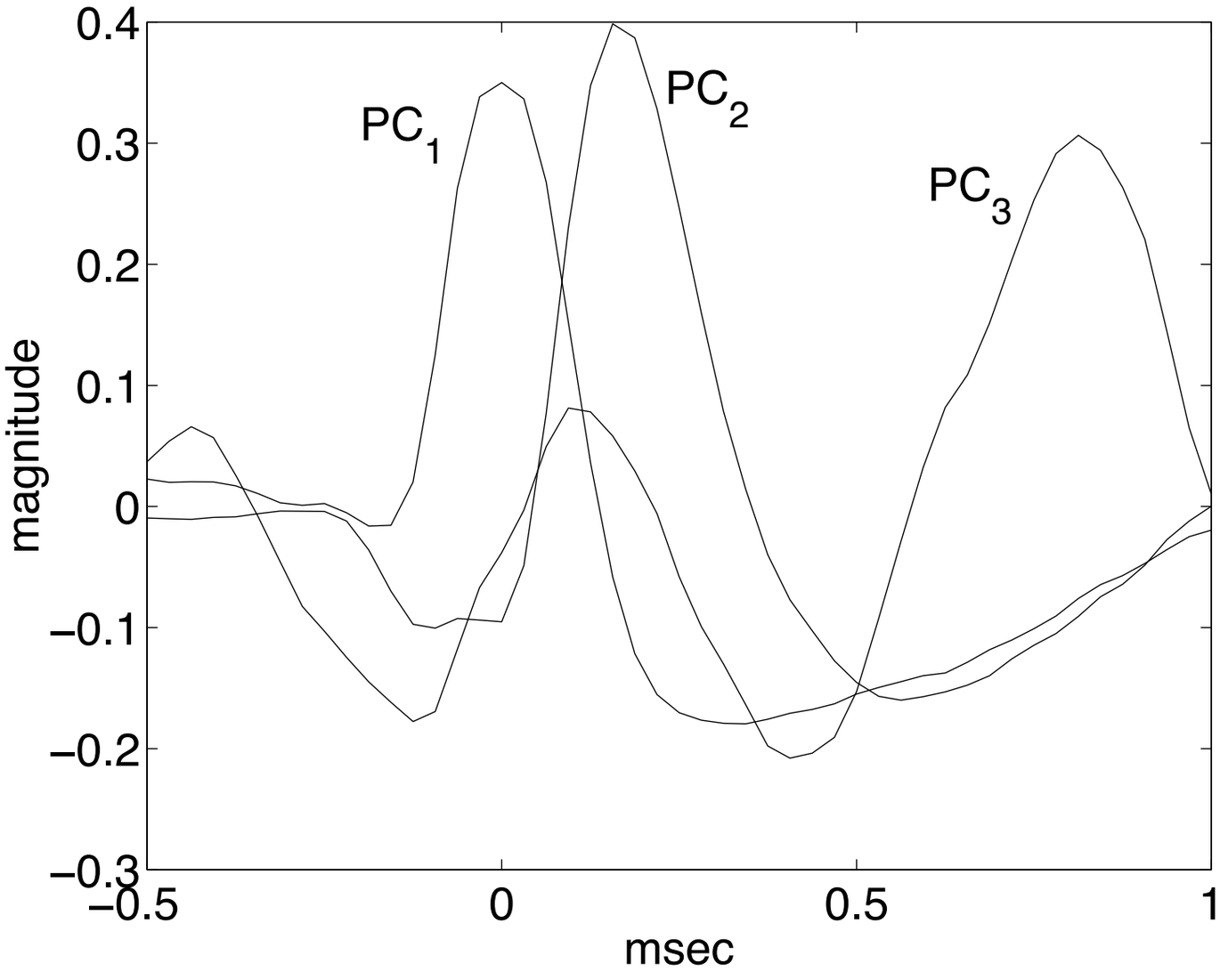


What is the dimensionality of the data?

How many components are there?

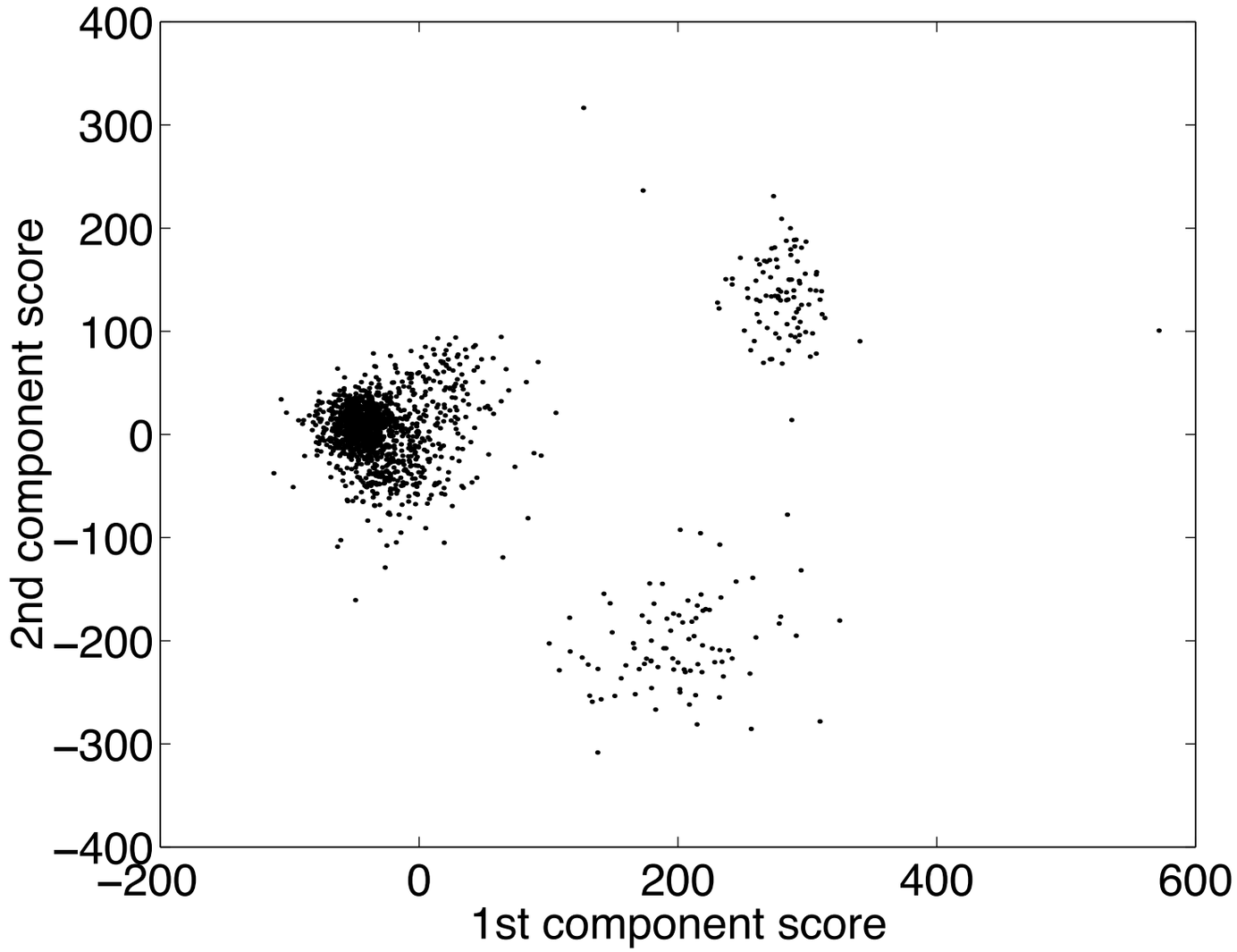
What will the components look like?

The first three principal components of the waveform data



What do you expect when we plot PC1 vs PC2 ?

Scatter plot of the first two principal component scores



Now the data are much better separated.

Could we use more PCs? How many?

Blanz and Vetter (2003)

Characterization of face shape and
texture using principal components

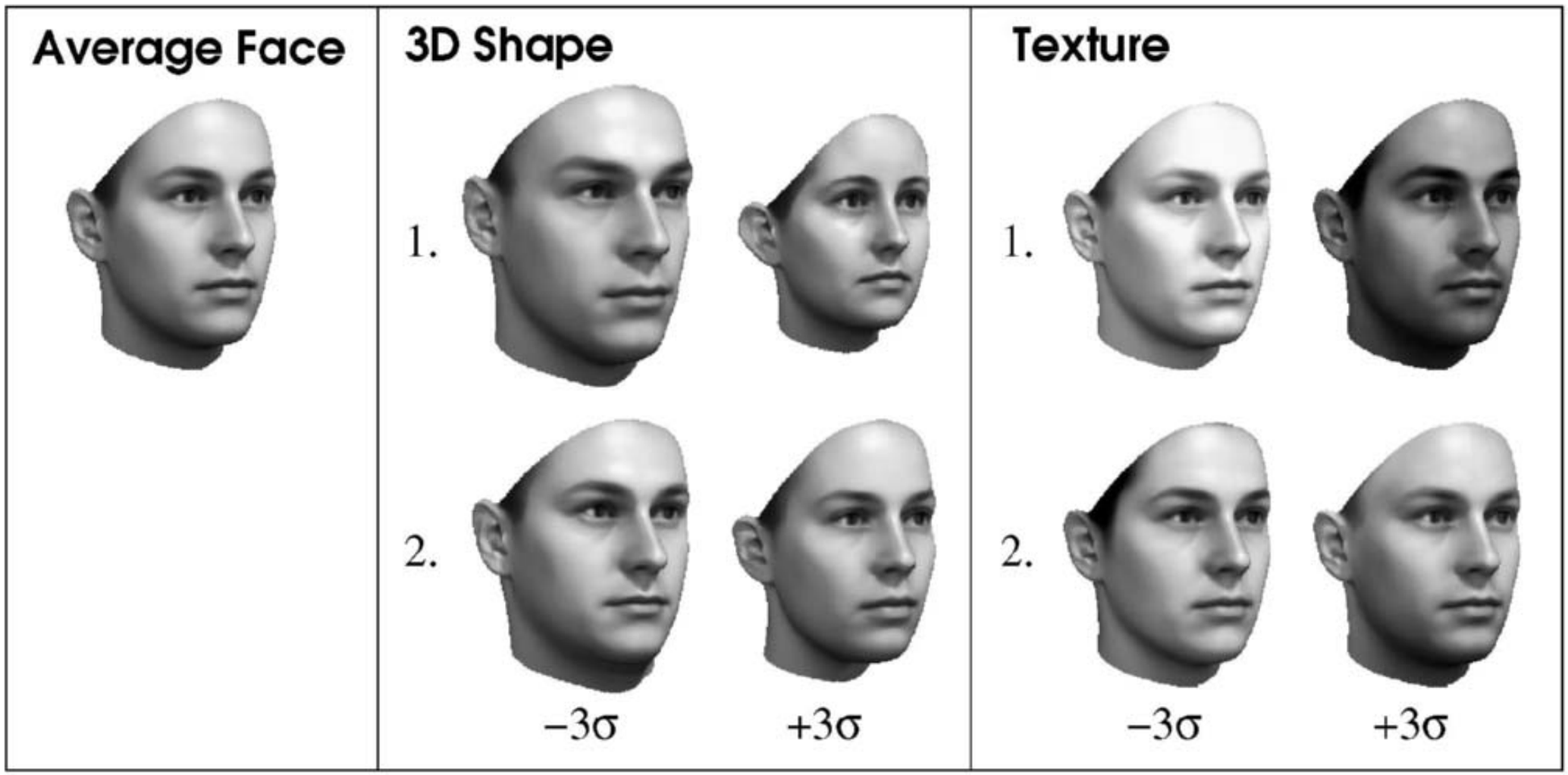
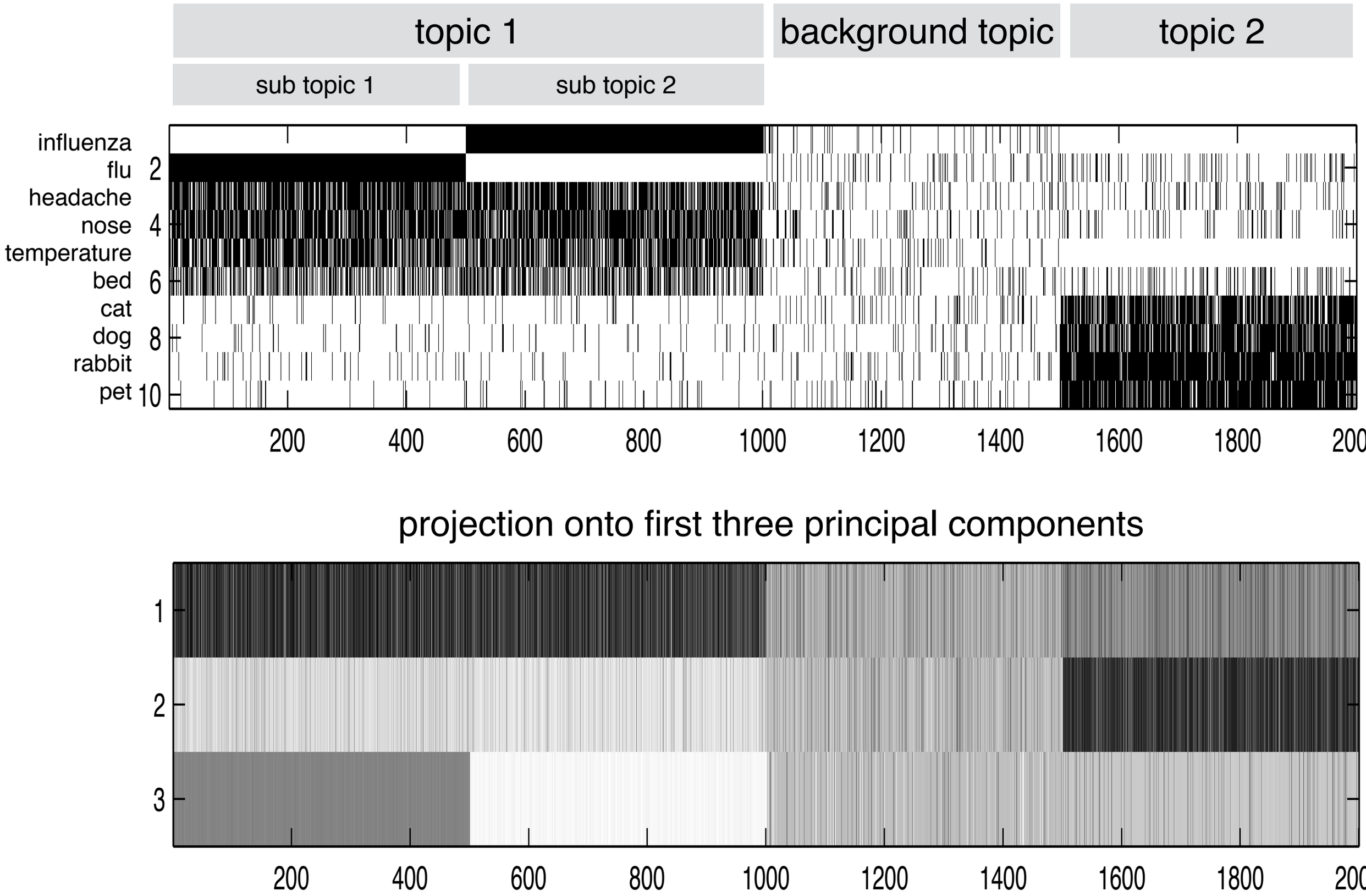


Fig. 4. The average and the first two principal components of a data set of 200 3D face scans, visualized by adding $\pm 3\sigma_{S,i}\mathbf{s}_i$ and $\pm 3\sigma_{T,i}\mathbf{t}_i$ to the average face.

Latent semantic analysis



		Linear Models (Barber Ch. 17&18)		
20	Mon, Apr 1	**Optimal Linear Reconstruction** - derivation, matrix equation techniques, matrix calculus, Lagrangians for matrix constraints	B.15.1-3, 17	A6 out
21	Wed, Apr 3	**Bayesian Interpolation** - non-linear generative models for noisy data, regression, curve fitting, regularization	B.18	A5 due
21	Wed, Apr 3	**Sparse Linear Models** - sparse representation and coding, independent component analysis (ICA), compressed sensing	B.21.6, M.13	
22	Mon, Apr 8	**Sparse Linear Models** (continued) - blind source separation (BSS), ICA learning algorithm		

Coding images with a statistical model

Goal: Encode the patters to desired precision:

$$\begin{aligned}\mathbf{x} &= \vec{a}_1 s_1 + \cdots + \vec{a}_L s_L + \vec{\epsilon} \\ &= \mathbf{A}\mathbf{s} + \boldsymbol{\epsilon}\end{aligned}$$

Posterior:

$$P(\mathbf{s}|\mathbf{x}, \mathbf{A}) = \frac{P(\mathbf{s})P(\mathbf{x}|\mathbf{s}, \mathbf{A})}{P(\mathbf{x}|\mathbf{A})}$$

Prior: s_i's are *independent* and *sparse*:

$$P(\mathbf{s}) = \prod_i P(s_i)$$

$$P(s_i) \propto \exp\left[-\left|\frac{s_i}{\lambda_i}\right|^{q_i}\right]$$

Likelihood: Assume $\boldsymbol{\epsilon} \sim$ Gaussian,

$$P(\mathbf{x}|\mathbf{s}, \Sigma) \propto \exp\left[-\frac{1}{2}\boldsymbol{\epsilon}^T \Sigma^{-1} \boldsymbol{\epsilon}\right]$$

Inference: use the MAP value:

$$\hat{\mathbf{s}} = \arg\max_{\mathbf{s}} P(\mathbf{s}|\mathbf{x}, \mathbf{A})$$

Simple special case: no noise (ICA)

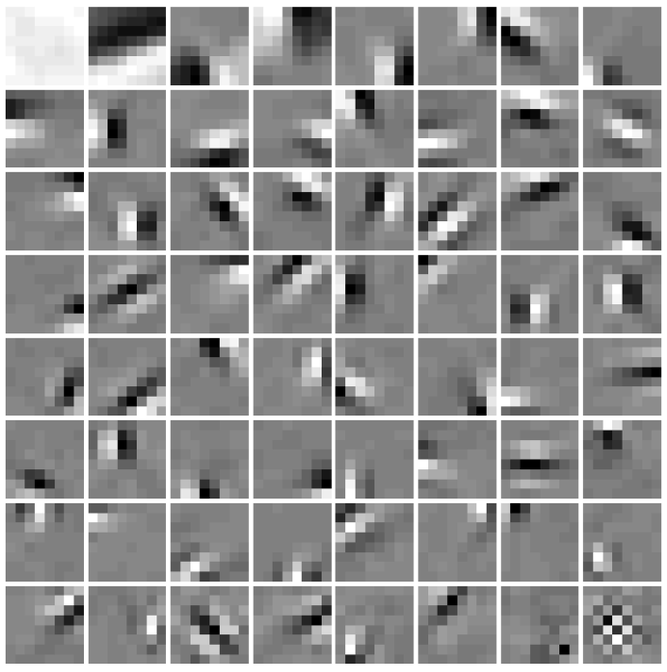
$$\hat{\mathbf{s}} = \mathbf{A}^{-1}\mathbf{x}$$

Inference (or recognition or coding):

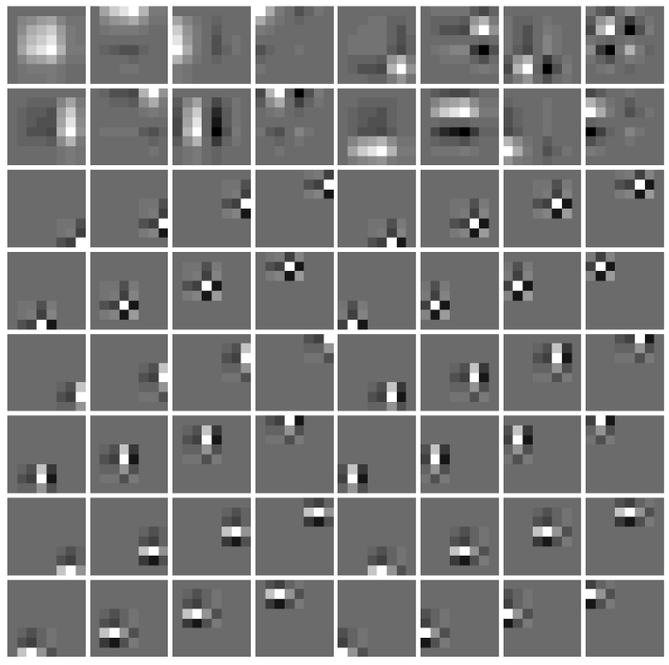
finds most efficient representation of pattern $\mathbf{x}$ *in a given basis* $\mathbf{A}$

Which code is best for natural images?

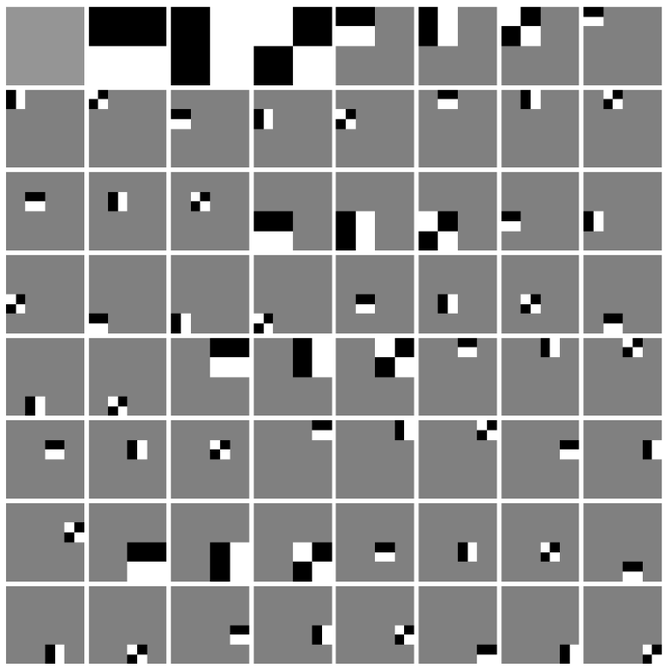
Learned



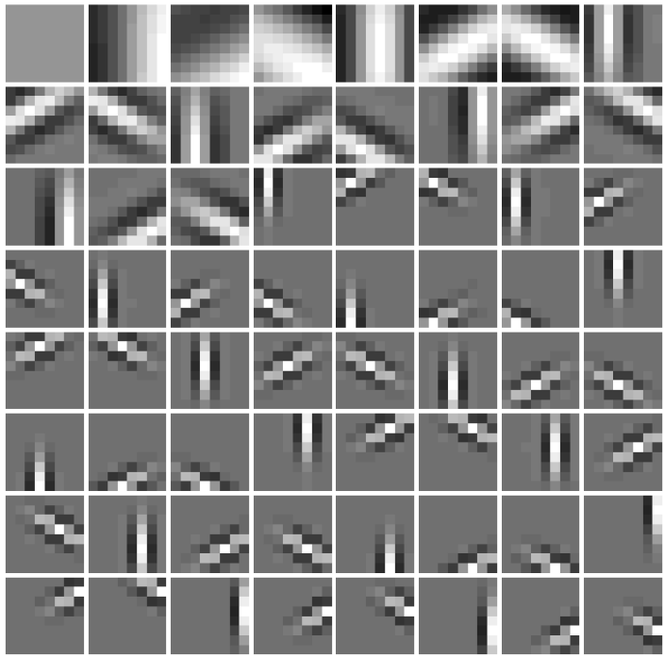
Wavelet



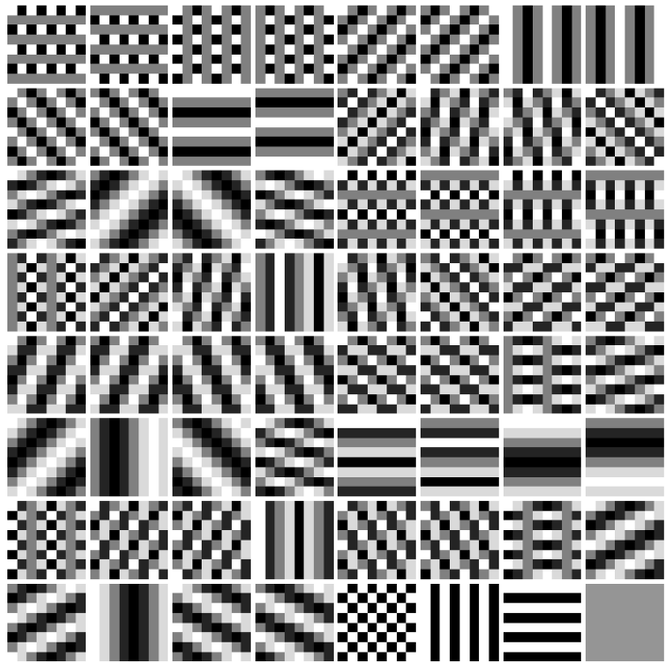
Haar



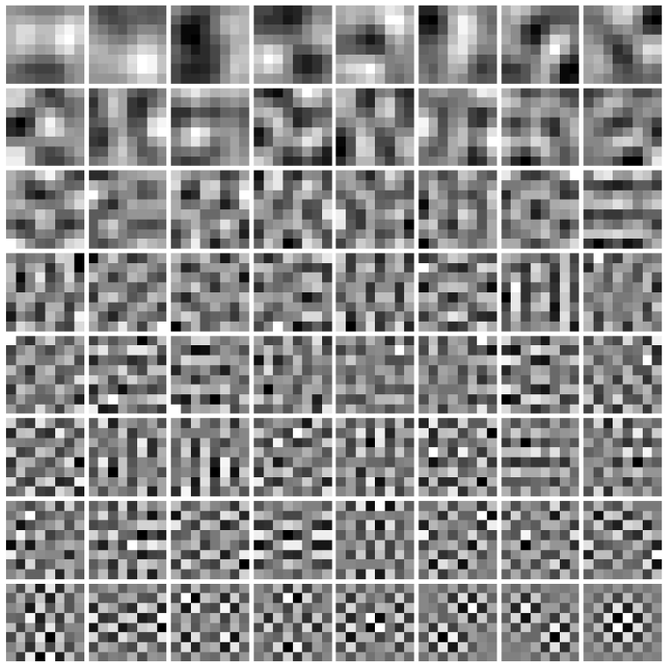
Gabor



Fourier



PCA



Each plot shows the basis functions for different image representations.

Learning non-Gaussian (ICA) mixture models

Natual Images

Scanned Newspapers

ıy Picture-Perfect Slice of the

ing on Route 4 over a dark and winding mountain pass, the visitor suddenly emerges into a "Lost Horizon" world of hot springs, trout streams and meadows of wildflowers, where cattle and the state's largest elk herd graze side by side.

But the same sense of wide-open Western independence evoked by the vistas has prevented the sale of the land for years. And the deal that is being negotiated for the ranch, which has been owned by one family for almost 40 years, is as much about Western attitudes toward public land as it is about money.

The Administration has long supported the purchase of the ranch, which has been called "the hole in the doughnut" because it is an island surrounded by the Santa Fe National Forest. Last February on a visit to New Mexico, President had Air Force One mak fly over the ranch for a l its dominant feature — wide crater of the dorm

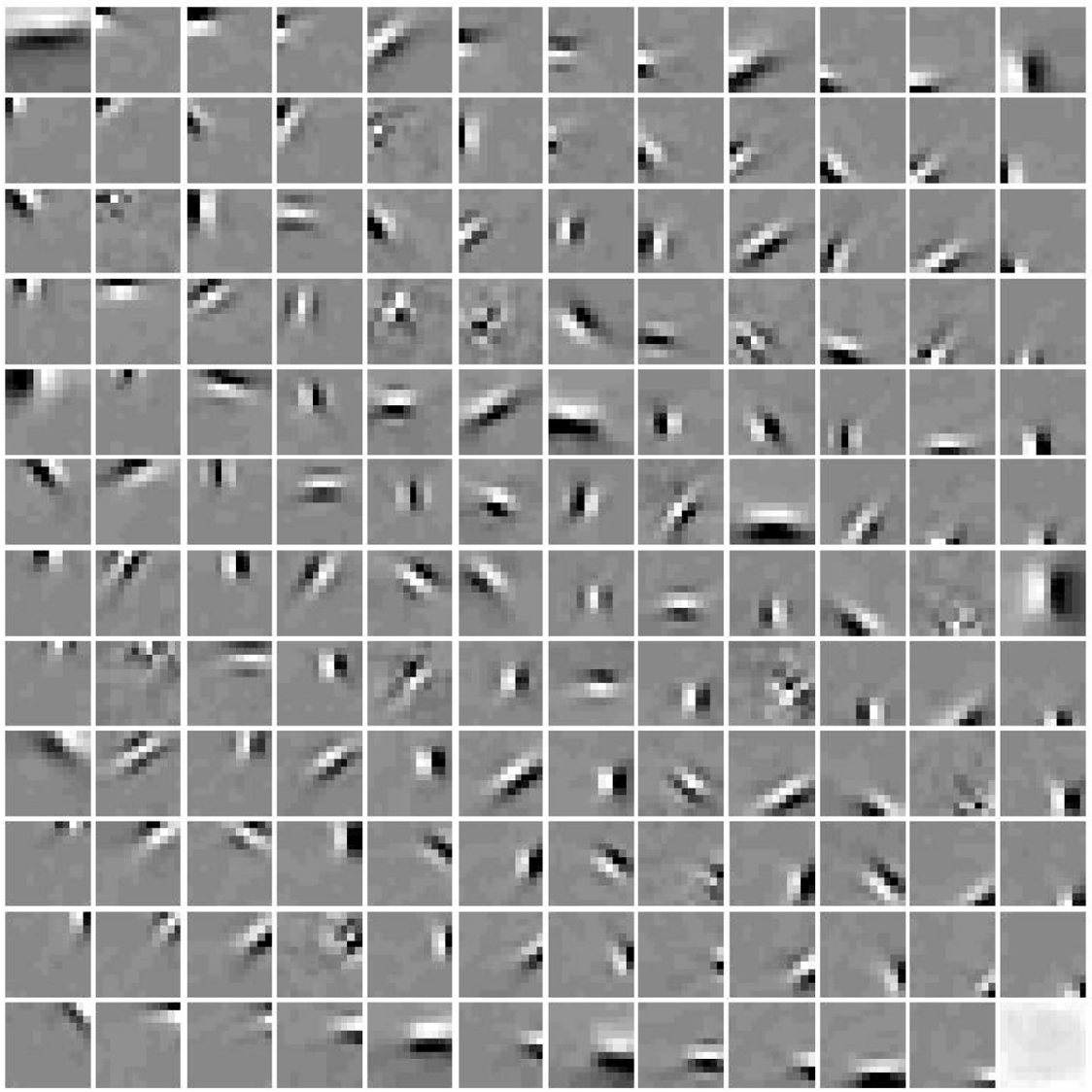
Republicans, noting one-third of New Mexi owned by the Federal have long opposed chases. But in August sentiment began shift cally.

Under legislation dra tor Pete V. Domenici, R New Mexico, the Baca separate unit of the Na system, owned by the Forest Service, but m trust, comprised of n pointed by the Presiden

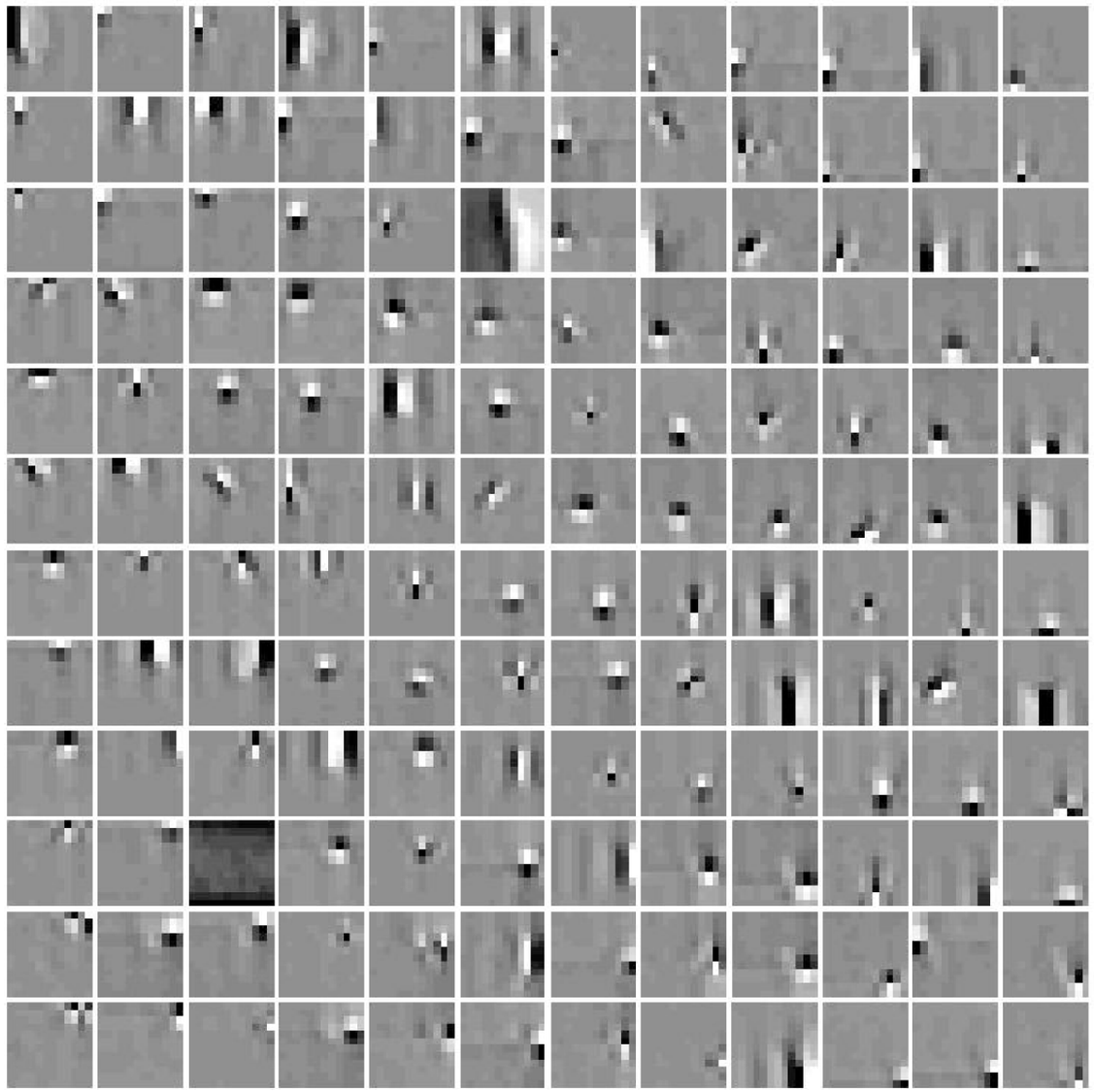
Continued on Pag

Learing ICA mixtures for natural images and newpapers

"Cluster" 1: Natural Image Basis

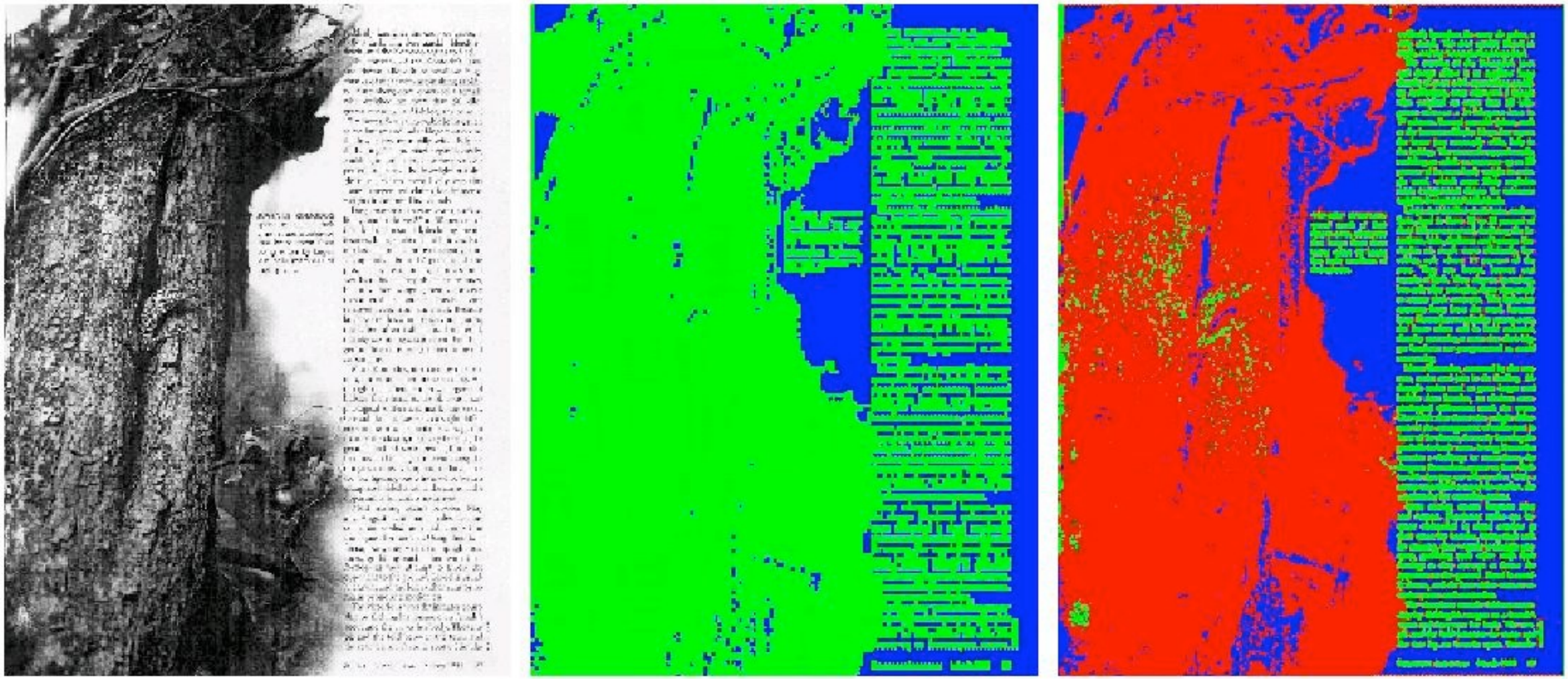


"Cluster" 2: Scanned Newspaper Basis



Lee, Lewicki, and Sejnowski, 2000

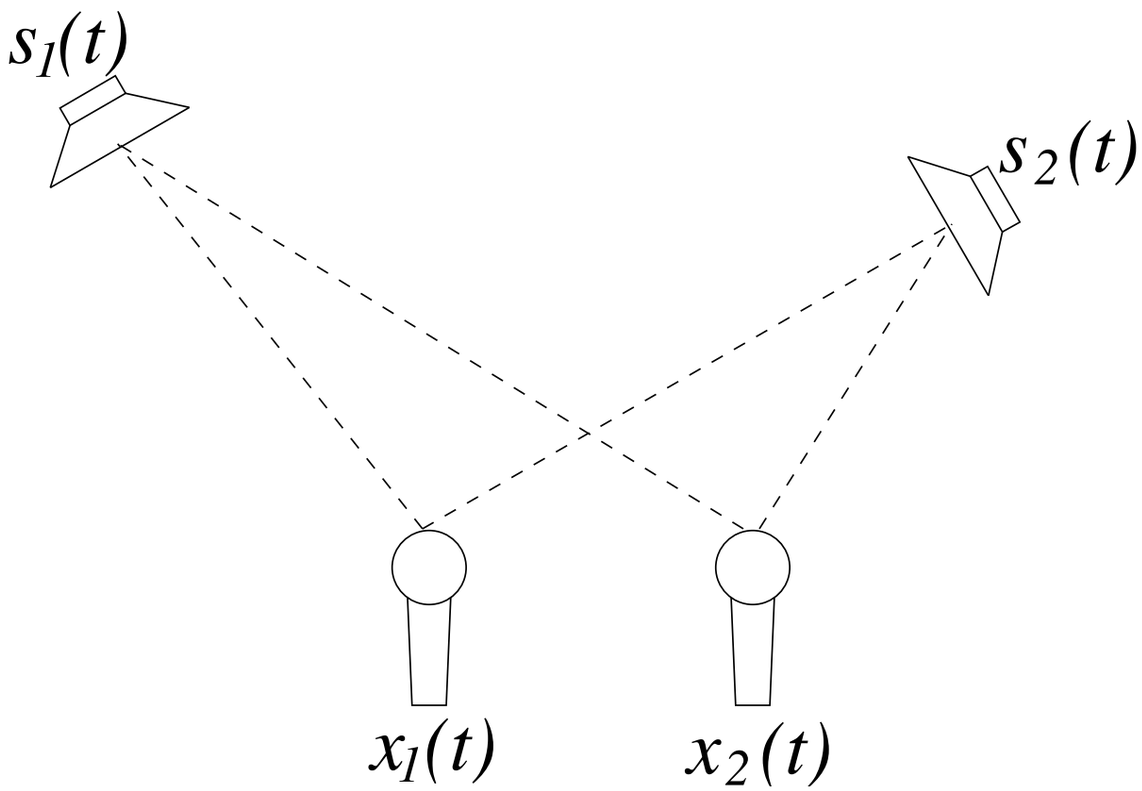
Fine-grained image segmentation



Lee and Lewicki, 2002

Modeling the cocktail party problem

Suppose we have two speakers (sources), $s_1(t)$ and $s_2(t)$ and two microphones (mixtures), $x_1(t)$ and $x_2(t)$:



The general problem is called *blind source separation*. We only observe the mixtures and are "blind" to the sources.

How do we model this in general?

Modeling non-Gaussian distributions

Learning objective: model statistical density of sources:

$\Rightarrow$ maximize $P(\mathbf{x}|\mathbf{A})$ over $\mathbf{A}$.

Probability of pattern ensemble is:

$$P(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N|\mathbf{A}) = \prod_k P(\mathbf{x}_k|\mathbf{A})$$

To obtain $P(\mathbf{x}|\mathbf{A})$ marginalize over $\mathbf{s}$:

$$\begin{aligned} P(\mathbf{x}|\mathbf{A}) &= \int d\mathbf{s}\, P(\mathbf{x}|\mathbf{A}, \mathbf{s}) P(\mathbf{s}) \\ &= \frac{P(\mathbf{s})}{|\det \mathbf{A}|} \end{aligned}$$

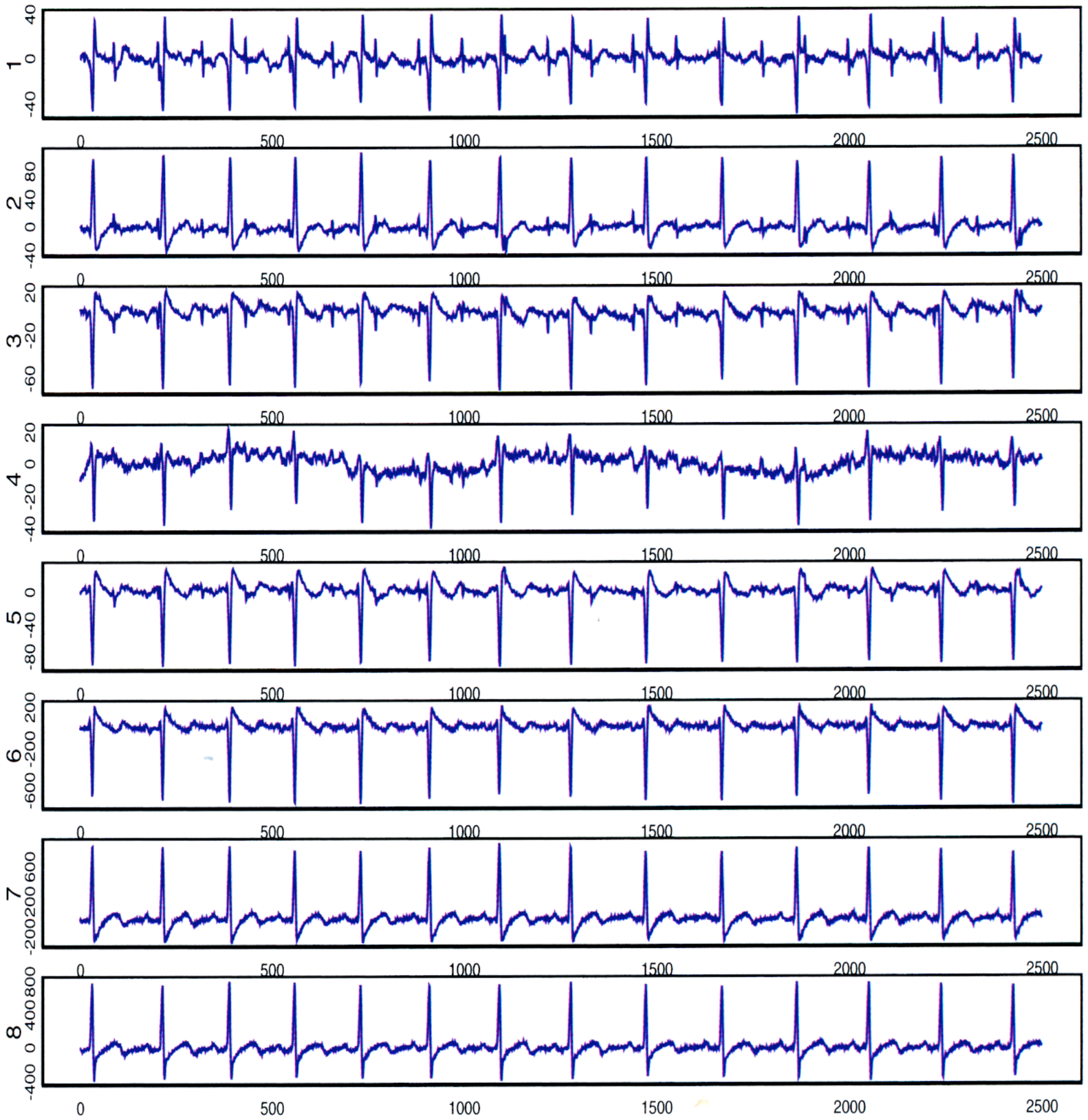
Learning rule (ICA):

$$\begin{aligned} \Delta\mathbf{A} &\propto \mathbf{A}\mathbf{A}^T \frac{\partial}{\partial \mathbf{A}} \log P(\mathbf{x}|\mathbf{A}) \\ &= -\mathbf{A}(\mathbf{z}\mathbf{s}^T - \mathbf{I}), \end{aligned}$$

where $\mathbf{z} = (\log P(\mathbf{s}))'$. Use $P(s_i) \sim \text{ExPwr}(s_i|\mu, \sigma, \beta_i)$.

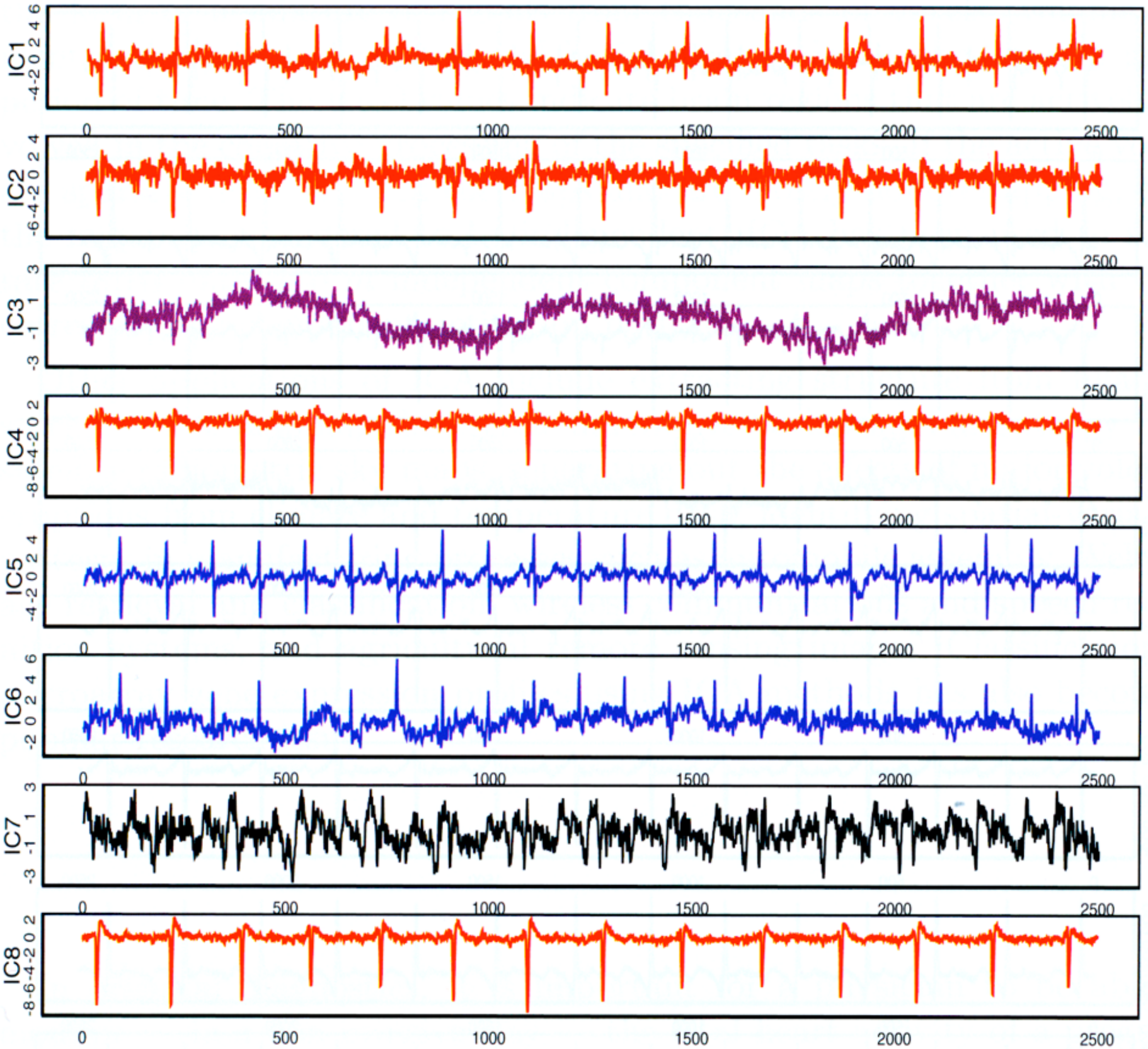
This is identical to the procedure that was used to learn efficient codes, i.e *independent component analysis*.

Blind source separation of cardiac rhythms (De Lathauwer et al, 2000)



- 2,500 ECG points from pregnant women
- eight channels of cutaneous data at 500 Hz

Blind source separation of cardiac rhythms (De Lathauwer et al, 2000)



mother

mother

respiration

mother

fetus

fetus

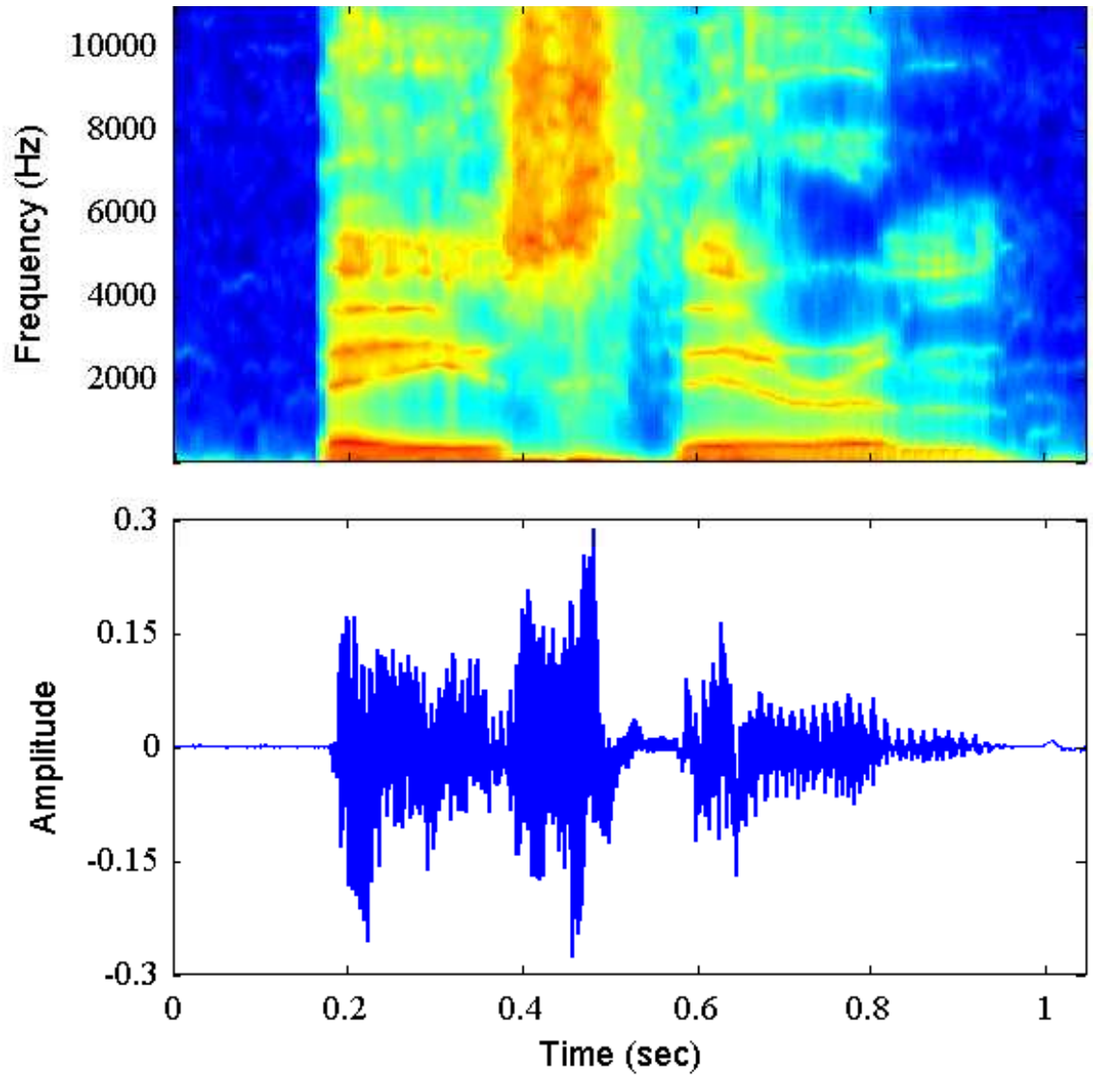
noise

mother

		Dynamic Models (Barber Ch. 23&24)		
23	Wed, Apr 10	**Dynamical Models** - discrete state Markov models, transition matrix, language modeling, MLE, mixture of Markov models, applications: Google PageRank algorithm, gene clustering	B.23.1, M. 17-1-2	A6 due; A7 out
24	Mon, Apr 15	**Hidden Markov Models (HMMs)** - classical inference problems, forward algorithm, forward-backwards algorithm, Viterbi algorithm, sampling, natural language models	B.23.2, M. 17.3-4	
25	Wed, Apr 17	**Learning HMSs** - EM.for HMMs (Baum-Welch algorithm), GMM.emission model, discriminative training, related models and generalizations, dynamic Bayes nets, applications: object tracking, speech recognition, bioinformatics, part-of-speech tagging	B.23.3-5, M. 17.5-6	
26	Mon, Apr 22	**Continuous-state Markov Models 1** - linear dynamical systems (LDS), stationary distributions, autoregressive models, latent linear dynamical systems, Kalman filtering, robotic SLAM	B.24.1-3, M. 18.1-3	
27	Wed, Apr 24	**Continuous-state Markov Models 2** - inference in CSMMs, filtering, smoothing, trajectory analysis, learning LDS, EM.for LDS, approximate online inference	B.24.4-7, M. 18.4-5	A7 due
28	Mon, Apr 29	**Retrospective**		
		No final exam. Grad student projects due Mon, Apr 29.		

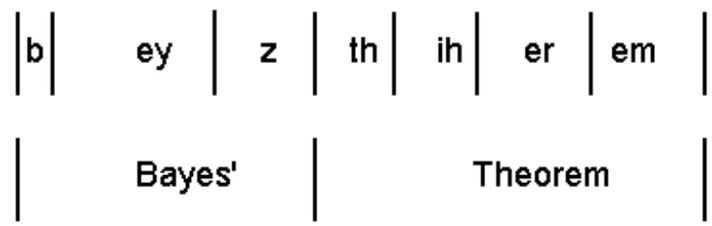
Examples of sequential data

- speech spectrogram
  - sequences of power spectra
  - typically 30 ms with 10 ms overlap



- audio waveform: raw sample values

- sequences of syllables and words



23.2.1 The classical inference problems

The common inference problems in HMMs are summarised below:

Filtering	(Inferring the present)	$p(h_t \vert v_{1:t})$	
Prediction	(Inferring the future)	$p(h_t \vert v_{1:s})$	$t > s$
Smoothing	(Inferring the past)	$p(h_t \vert v_{1:u})$	$t < u$
Likelihood		$p(v_{1:T})$	
Most likely Hidden path	(Viterbi alignment)	$\underset{h_{1:T}}{\text{argmax}}\, p(h_{1:T} \vert v_{1:T})$	

The β recursion

$$p(v_{t:T}|h_{t-1}) = \sum_{h_t} p(v_t, v_{t+1:T}, h_t|h_{t-1})$$

$$= \sum_{h_t} p(v_t|\cancel{v_{t+1:T}}, h_t, \cancel{h_{t-1}})p(v_{t+1:T}, h_t|h_{t-1})$$

$$= \sum_{h_t} p(v_t|h_t)p(v_{t+1:T}|h_t, \cancel{h_{t-1}})p(h_t|h_{t-1})$$

Again use:
p(a,b) = p(a|b)p(b)

Defining $\beta(h_t) \equiv p(v_{t+1:T}|h_t)$ gives the β-recursion

$$\beta(h_{t-1}) = \sum_{h_t} p(v_t|h_t)p(h_t|h_{t-1})\beta(h_t), \qquad 2 \leq t \leq T$$

with $\beta(h_T) = 1$. The smoothed posterior is then given by

$$p(h_t|v_{1:T}) \equiv \gamma(h_t) = \frac{\alpha(h_t)\beta(h_t)}{\sum_{h_t} \alpha(h_t)\beta(h_t)}$$

equivalent to message passing on factor graphs

Together the $\alpha - \beta$ recursions are called the Forward-Backward algorithm.

Hidden Markov models

$\mathbf{X}_t$ is a single, discrete variable (usually $\mathbf{E}_t$ is too)
Domain of X_t is $\{1, \ldots, S\}$

Transition matrix $\mathbf{T}_{ij} = P(X_t = j | X_{t-1} = i)$, e.g., $\begin{pmatrix} 0.7 & 0.3 \\ 0.3 & 0.7 \end{pmatrix}$

Sensor matrix $\mathbf{O}_t$ for each time step, diagonal elements $P(e_t | X_t = i)$

e.g., with $U_1 = true$, $\mathbf{O}_1 = \begin{pmatrix} 0.9 & 0 \\ 0 & 0.2 \end{pmatrix}$

Forward and backward messages as column vectors:

$$\begin{aligned} \mathbf{f}_{1:t+1} &= \alpha \mathbf{O}_{t+1} \mathbf{T}^\top \mathbf{f}_{1:t} \\ \mathbf{b}_{k+1:t} &= \mathbf{T} \mathbf{O}_{k+1} \mathbf{b}_{k+2:t} \end{aligned}$$

Forward-backward algorithm needs time $O(S^2 t)$ and space $O(St)$

Tracking observations over time: Kalman filtering

- Consider the simple linear model assuming constant velocity

$$X_{t+\Delta} = X_t + \dot{X}\Delta$$

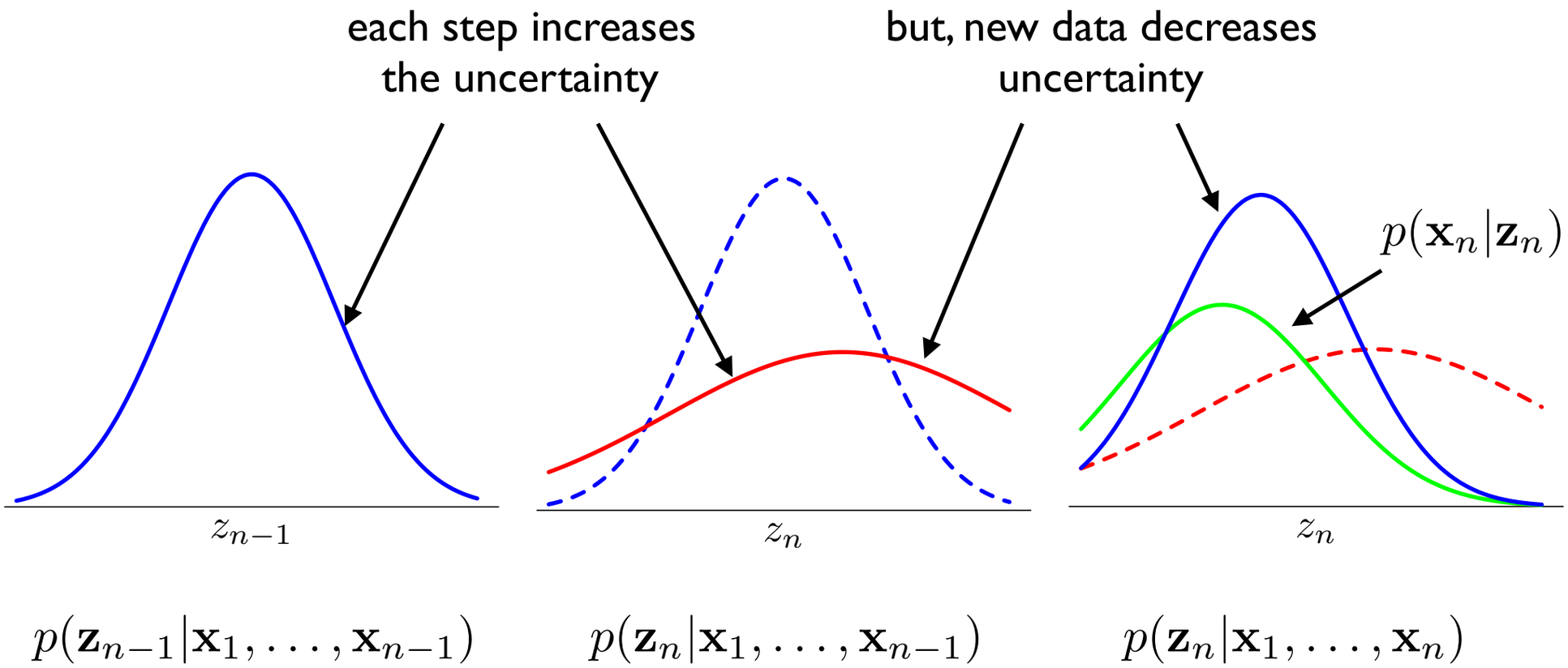
- Assuming Gaussian noise, we have the linear Gaussian transition model:

$$P(X_{t+\Delta} = x_{t+\Delta} | X_t = x_t, \dot{X}_t = \dot{x}_t) = \mathcal{N}(x_{t+\Delta} | x_t + \dot{x}_t\Delta, \sigma)$$

- But, not only is there noise in the transition, there is noise in the observation, ie we don't directly observe X_t, we only observe a noisy version of it:

$$Z_t = X_t + \epsilon_t$$

Linear dynamical systems: the Kalman filter in 1-D



$p(\mathbf{z}_{n-1}|\mathbf{x}_1,\ldots,\mathbf{x}_{n-1})$ $p(\mathbf{z}_n|\mathbf{x}_1,\ldots,\mathbf{x}_{n-1})$ $p(\mathbf{z}_n|\mathbf{x}_1,\ldots,\mathbf{x}_n)$

Particle filtering contd.

Assume consistent at time t: $N(\mathbf{x}_t|\mathbf{e}_{1:t})/N = P(\mathbf{x}_t|\mathbf{e}_{1:t})$

Propagate forward: populations of $\mathbf{x}_{t+1}$ are

$$N(\mathbf{x}_{t+1}|\mathbf{e}_{1:t}) = \sum_{\mathbf{x}_t} P(\mathbf{x}_{t+1}|\mathbf{x}_t) N(\mathbf{x}_t|\mathbf{e}_{1:t})$$ total # samples

Weight samples by their likelihood for $\mathbf{e}_{t+1}$:

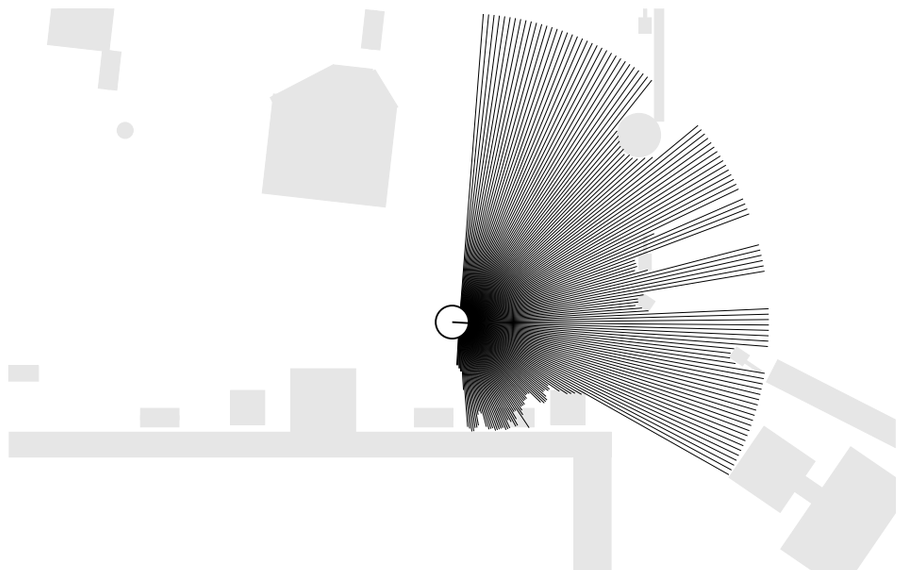
$$W(\mathbf{x}_{t+1}|\mathbf{e}_{1:t+1}) = P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1}) N(\mathbf{x}_{t+1}|\mathbf{e}_{1:t})$$ total weight in samples

Resample to obtain populations proportional to W: #samples prop. to weight

$$\begin{aligned} N(\mathbf{x}_{t+1}|\mathbf{e}_{1:t+1})/N &= \alpha W(\mathbf{x}_{t+1}|\mathbf{e}_{1:t+1}) = \alpha P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1}) N(\mathbf{x}_{t+1}|\mathbf{e}_{1:t}) \\ &= \alpha P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1}) \sum_{\mathbf{x}_t} P(\mathbf{x}_{t+1}|\mathbf{x}_t) N(\mathbf{x}_t|\mathbf{e}_{1:t}) \\ &= \alpha' P(\mathbf{e}_{t+1}|\mathbf{x}_{t+1}) \sum_{\mathbf{x}_t} P(\mathbf{x}_{t+1}|\mathbf{x}_t) P(\mathbf{x}_t|\mathbf{e}_{1:t}) \\ &= P(\mathbf{x}_{t+1}|\mathbf{e}_{1:t+1}) \end{aligned}$$

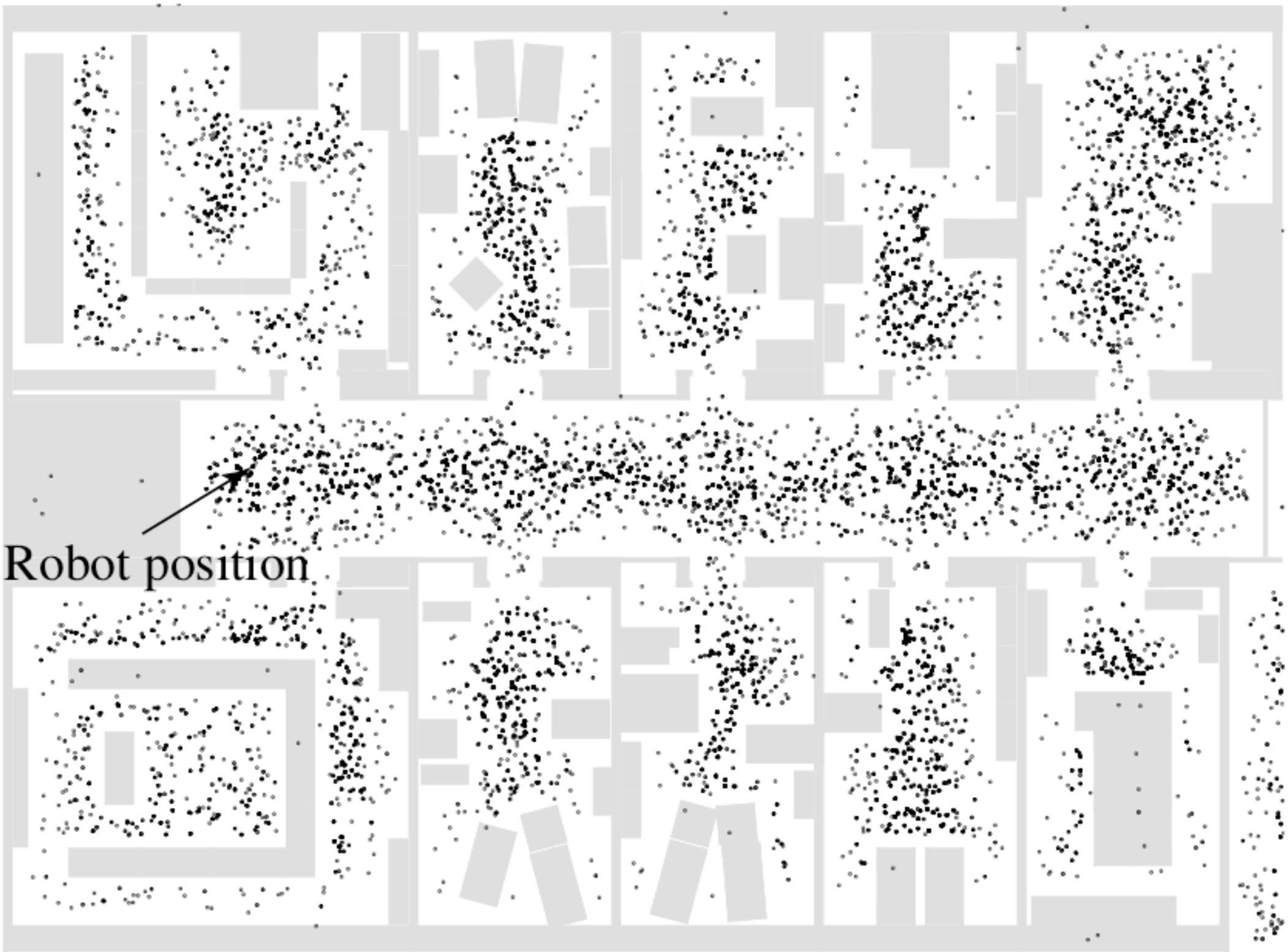
Sensors

Range finders: sonar (land, underwater), laser range finder, radar (aircraft), tactile sensors, GPS



Imaging sensors: cameras (visual, infrared)
Proprioceptive sensors: shaft decoders (joints, wheels), inertial sensors, force sensors, torque sensors

Initial state: particles uniformly distributed based on prior



Second state: first measurements arrive