Bayesian Methods for Unsupervised Learning

Zoubin Ghahramani
Gatsby Computational Neuroscience Unit
University College London, UK

zoubin@gatsby.ucl.ac.uk
http://www.gatsby.ucl.ac.uk

Mathematical Psychology Conference
July 2003

What is Unsupervised Learning?

Unsupervised learning: given some data, learn a probabilistic model of the data:

- clustering models (e.g. k-means, mixture models)
- dimensionality reduction models (e.g. factor analysis, PCA)
- generative models which relate the hidden causes or sources to the observed data (e.g. ICA, hidden markov models)
- models of conditional independence between variables (e.g. graphical models)
- other models of the data density

Supervised learning: given some input and target data, learn a mapping from inputs to targets:

- classification/discrimination (e.g. perceptron, logistic regression)
- function approximation (e.g. linear regression)

Reinforcement learning: systems learns to produce actions while interacting in an environment so as to maximize its expected sum of long-term rewards. Formally equivalent to **sequential decision theory** and **optimal adaptive control theory**.

Bayesian Learning

Consider a data set $\mathcal{D}$, and a model m with parameters θ.

Prior over model parameters: $p(\theta|m)$
Likelihood of model parameters for data set $\mathcal{D}$: $p(\mathcal{D}|\theta, m)$
Prior over model class: $p(m)$

The likelihood and parameter priors are combined into the posterior for a particular model; batch and online versions:

$$p(\theta|\mathcal{D}, m) = \frac{p(\mathcal{D}|\theta, m)p(\theta|m)}{p(\mathcal{D}|m)} \qquad p(\theta|\mathcal{D}, x, m) = \frac{p(x|\theta, \mathcal{D}, m)p(\theta|\mathcal{D}, m)}{p(x|\mathcal{D}, m)}$$

Predictions are made by integrating over the posterior:

$$p(x|\mathcal{D}, m) = \int d\theta \; p(x|\theta, \mathcal{D}, m) \; p(\theta|\mathcal{D}, m).$$

Bayesian Model Comparison

A data set $\mathcal{D}$, and a model m with parameters θ.

Prior over model parameters: $p(\theta|m)$
Likelihood of model parameters for data set $\mathcal{D}$: $p(\mathcal{D}|\theta, m)$
Prior over model class: $p(m)$

To compare models, we again use Bayes' rule and the prior on models

$$p(m|\mathcal{D}) \propto p(\mathcal{D}|m)\, p(m)$$

This also requires an integral over θ:

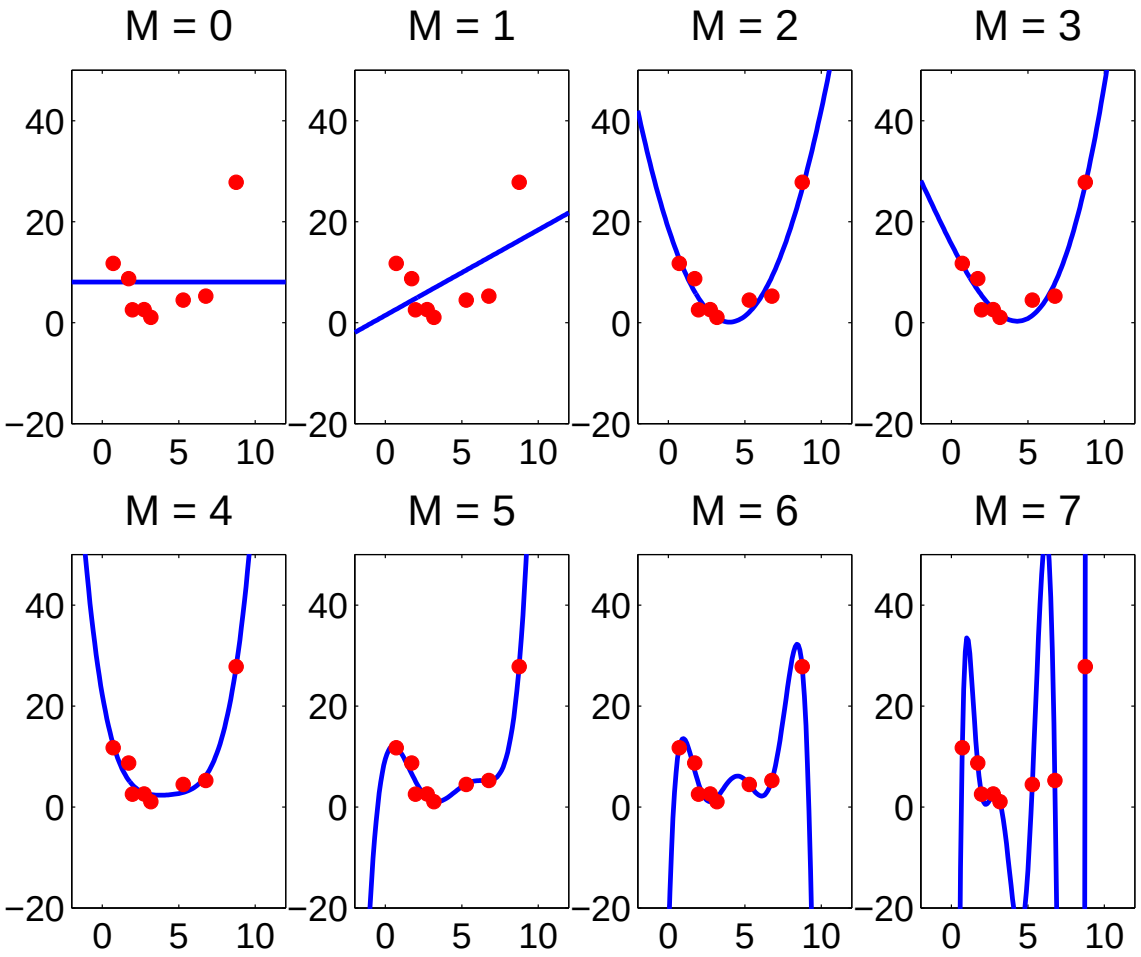
$$p(\mathcal{D}|m) = \int d\theta\, p(\mathcal{D}|\theta, m)\, p(\theta|m)$$

For interesting models, these integrals may be difficult to compute.

Why be Bayesian?

- Cox Axioms lead to the following:
  If you want to represent uncertain beliefs numerically then, given some basic desiderata, Bayes Rule is the only coherent way to manipulate them.
- The Dutch Book Theorem:
  If you are willing to accept bets with odds proportional to your beliefs, then unless your beliefs satisfy Bayes rule, there exists some set of simultaneous bets which you would accept, but which are guaranteed to lose you money, no matter what the outcome!
- Asymptotic Convergence and Consensus: You will converge to the truth if you assigned it non-zero prior; and different Bayesians will converge to the same posterior as long as they assigned non-zero prior to the same set.
- Automatic: no arbitrary learning rates, no overfitting (no fitting!), honest about ignorance, principled framework for model selection and decision making,...
- Ubiquitous/Fashionable: vision, cue integration, concept learning in humans, language learning, robotics, movement control, machine learning,...

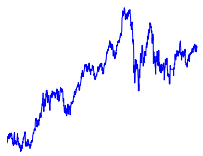
Model structure and overfitting: a simple example



Learning Model Structure

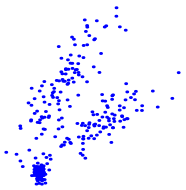
- **Feature Selection**

  Is some input relevant to predicting some output ?



- **Cardinality of Discrete Latent Variables**

  How many clusters in the data?

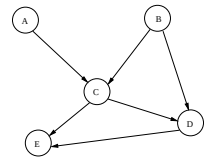


  How many states in a hidden Markov model?

```
SVYDAAAQLTADVKKDLRDSWKVIGSDKKGNGVALMTTY
```

- **Dimensionality of Real Valued Latent Vectors**
  What choice of dimensionality in a PCA/FA model of the data?
  How many state variables in a linear-Gaussian state-space model?

- **Conditional Independence Structure**
  What is the structure of a probabilistic graphical model/Bayesian network (i.e. what $\perp\!\!\!\perp$ relations hold)?



Using Bayesian Occam's Razor to Learn Model Structure

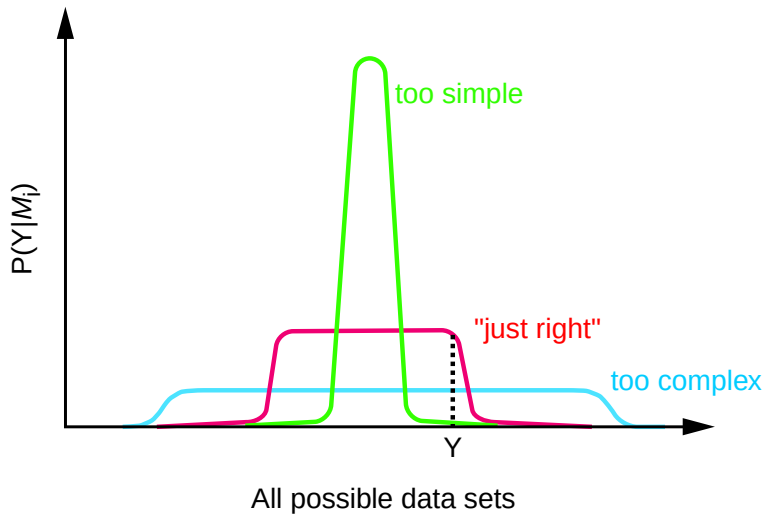
Select the model class, m_i, with the highest probability given the data, $\mathbf{y}$:

$$p(m_i|\mathbf{y}) = \frac{p(\mathbf{y}|m_i)p(m_i)}{p(\mathbf{y})}, \qquad p(\mathbf{y}|m_i) = \int p(\mathbf{y}|\boldsymbol{\theta}, m_i)\, p(\boldsymbol{\theta}|m_i)\, d\boldsymbol{\theta}$$

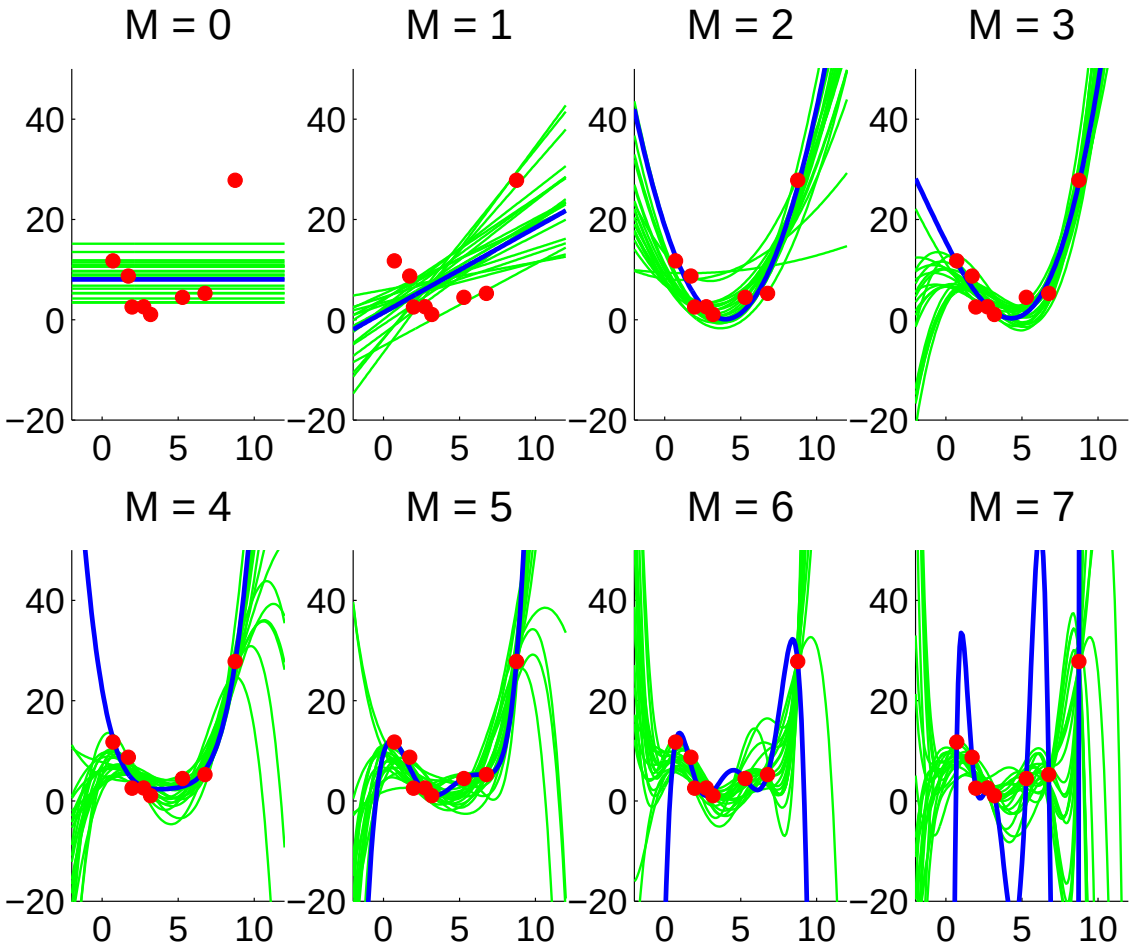
Interpretation of the Marginal Likelihood ("evidence"): The probability that *randomly selected* parameters from the prior would generate $\mathbf{y}$.

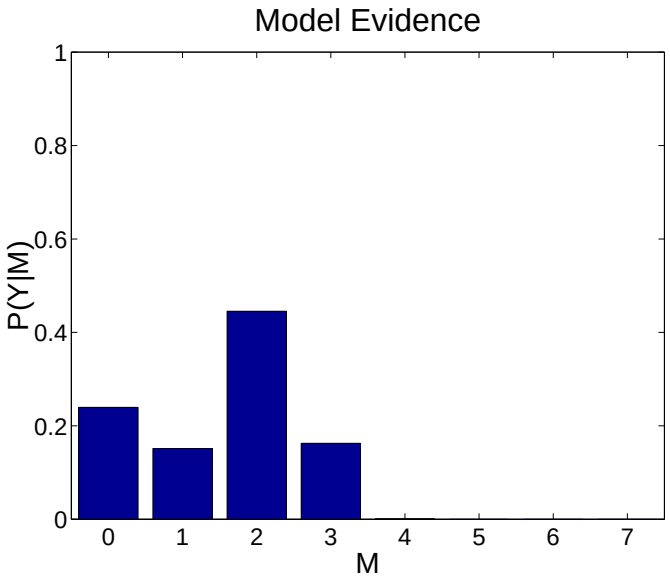
Model classes that are too simple are unlikely to generate the data set.

Model classes that are too complex can generate many possible data sets, so again, they are unlikely to generate that particular data set at random.



Bayesian Model Selection: Occam's Razor at Work





`demo: polybayes`

Subtleties of Occam's Hill

Latent Variable Models

General Setup:

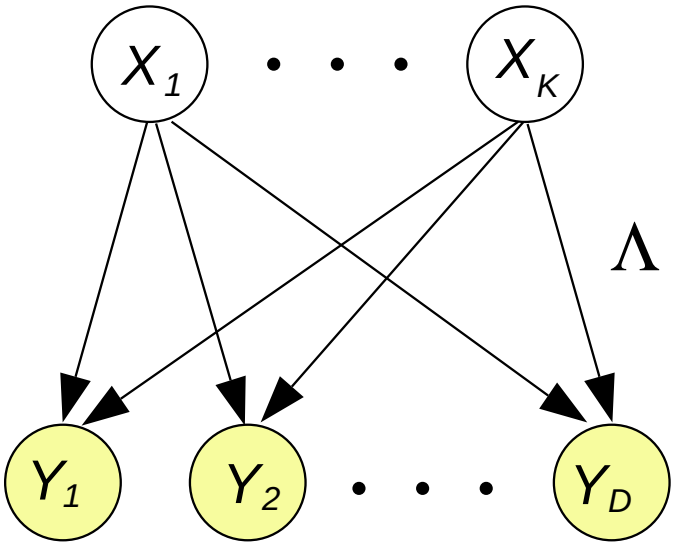
Model parameters: θ
Observed data set: $\mathcal{D} = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$
Latent variables: $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$

$$p(\mathcal{D}|\theta, m) = \prod_n p(\mathbf{y}_n|\theta, m) = \prod_n \left(\int p(\mathbf{y}_n|\mathbf{x}_n, \theta, m)\; p(\mathbf{x}_n|\theta, m)\; d\mathbf{x}_n \right)$$

Examples:

- Factor analysis and PCA
- Mixture models (e.g. mixtures of Gaussians)
- Hidden Markov models
- Linear Dynamical Systems
- Bayesian networks with hidden variables...

Factor Analysis



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
- fewer latent factors than observations: $K < D$

Under this model, $\mathbf{y}$ is Gaussian with:

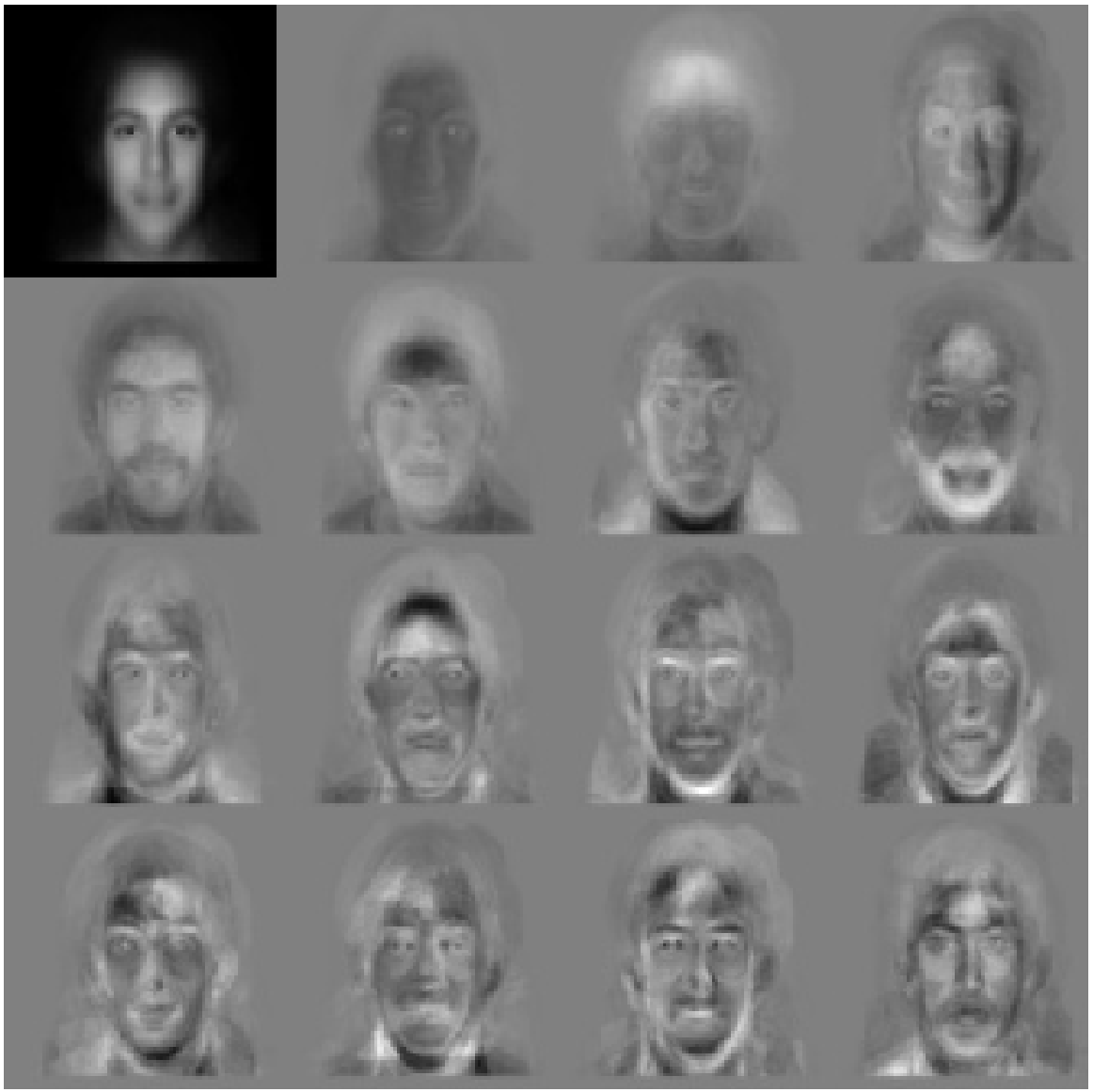
$$p(\mathbf{y}|\theta) = \int p(\mathbf{x})p(\mathbf{y}|\mathbf{x},\theta)d\mathbf{x} = \mathcal{N}(0, \Lambda\Lambda^\top + \Psi)$$

with parameters $\theta = \{\Lambda, \Psi\}$, where Λ is a $D \times K$ matrix, and Ψ is diagonal.

Dimensionality Reduction: Finds a low-dimensional representation of high dimensional data that captures most of the correlation structure of the data.

Principal Components Analysis (PCA) can be derived as a special case of FA where $\Psi = \lim_{\sigma^2 \to 0} \sigma^2 I$.

Example of PCA: Eigenfaces



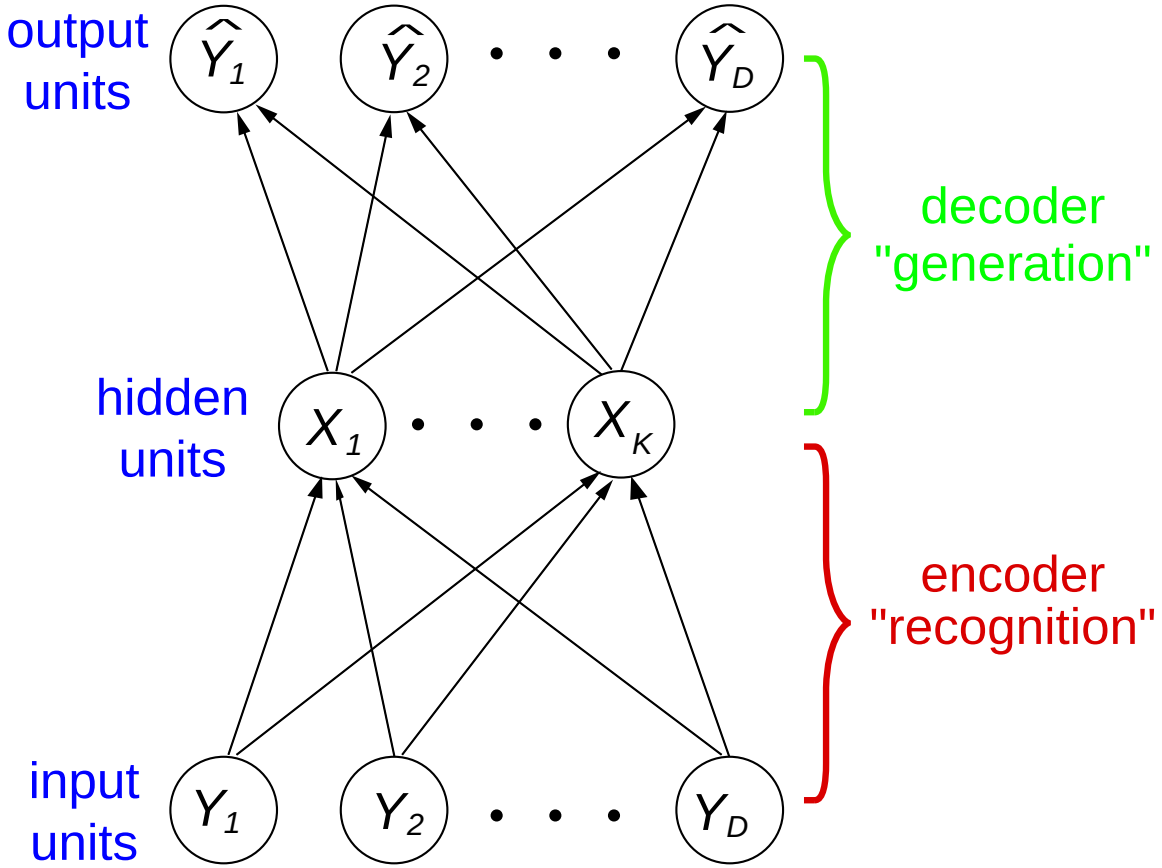
from www-white.media.mit.edu/vismod/demos/facerec/basic.html

FA vs PCA

- PCA is rotationally invariant; FA is not
- FA is measurement scale invariant; PCA is not
- FA defines a probabilistic model; PCA does not*

*But it is possible to define probabilistic PCA (PPCA).

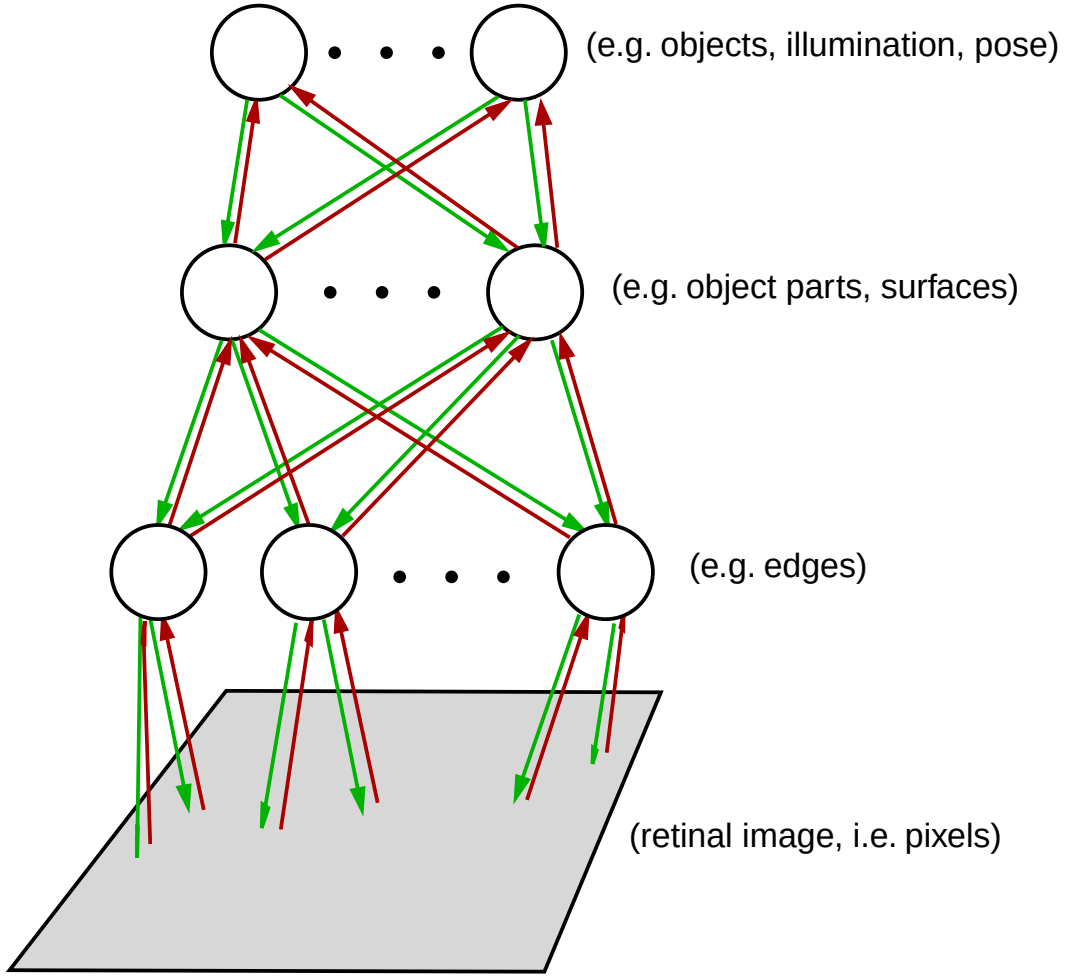
Neural Network Interpretations and Encoder-Decoder Duality



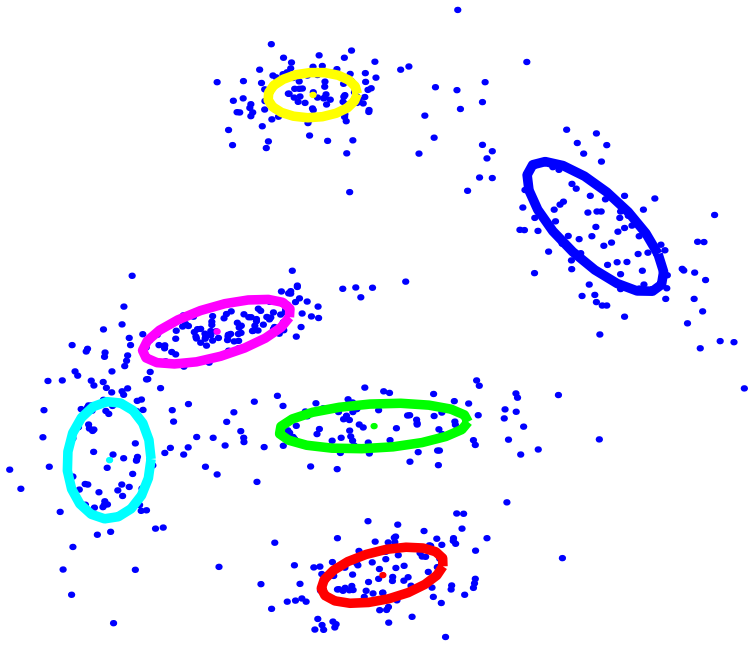
A linear autoencoder neural network trained to minimise squared error learns to perform PCA (Baldi & Hornik, 1989).
Other regularized cost functions lead to PPCA and FA (Roweis & Ghahramani, 1999).

Latent Variable Models

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



Mixtures of Gaussians



Model with discrete hidden variable s_n, parameters $\theta = \{\pi, \mu, \Sigma\}$:

$$p(\mathbf{y}_n|\theta) = \sum_{s_n=1}^{K} p(\mathbf{y}_n|s_n, \theta)p(s_n|\theta)$$

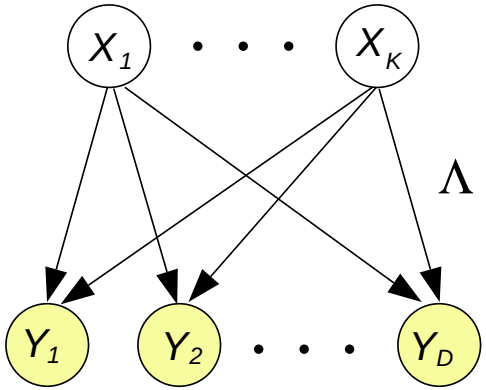
$$p(s_n = k|\pi) = \pi_k \qquad \text{and} \qquad p(\mathbf{y}_n|s_n = k, \mu, \Sigma) = \mathcal{N}(\mu_k, \Sigma_k)$$

k-means algorithm is a special case of learning a mixture of Gaussians using EM where $\Sigma_k = \lim_{\sigma^2 \to 0} \sigma^2 I$.

Blind Source Separation and Independent Components Analysis (ICA)



Independent Components Analysis (ICA)

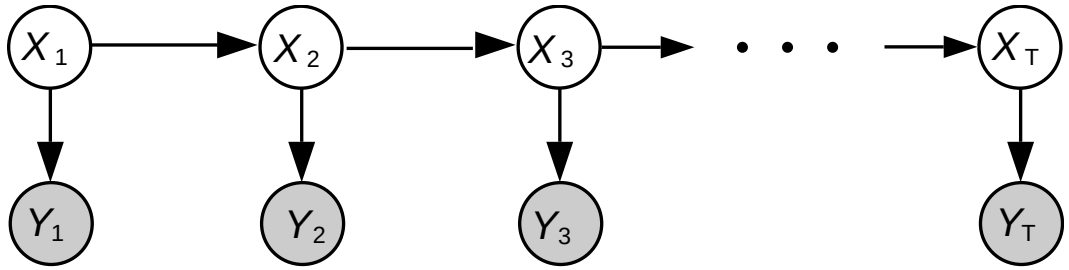


- $p(x_k)$ **is non-Gaussian.**
- Equivalently $p(x_k)$ is Gaussian, with a nonlinearity $g(\cdot)$: $y_d = \sum_{k=1}^{K} \Lambda_{dk}\, g(x_k) + \epsilon_d$
- For $K = D$, and observation noise assumed to be zero, inference and learning are easy (standard ICA).
- Many extensions are possible:
  - Fewer or more sources than "microphones" $(K \neq D)$
  - Allowing noise on microphones
  - Fitting source distributions
  - Time series versions with convolution by linear filter
  - Time-varying mixing matrix
  - Discovering number of sources

How ICA Relates to Factor Analysis and Other Models

- **Factor Analysis (FA)**: Assumes the factors are Gaussian.
- **Principal Components Analysis (PCA)**: Assumes no noise on the observations: $\Psi = \lim_{\epsilon \to 0} \epsilon I$
- **Independent Components Analysis (ICA)**: Assumes the factors are non-Gaussian (and no noise).
- **Mixture of Gaussians**: A single discrete-valued "factor": $x_k = 1$ and $x_j = 0$ for all $j \neq k$.
- **Mixture of Factor Analysers**: Assumes the data has several clusters, each of which is modeled by a single factor analyser.
- **Linear Dynamical Systems**: Time series model in which the factor at time t depends linearly on the factor at time $t - 1$, with Gaussian noise.

Linear-Gaussian State-space models (SSMs)



$$P(\mathbf{x}_{1:T}, \mathbf{y}_{1:T}) = P(\mathbf{x}_1)P(\mathbf{y}_1|\mathbf{x}_1)\prod_{t=2}^{T} P(\mathbf{x}_t|\mathbf{x}_{t-1})P(\mathbf{y}_t|\mathbf{x}_t)$$

where $\mathbf{x}_t$ and $\mathbf{y}_t$ are both real-valued vectors

Output equation: $$y_{t,i} = \sum_j C_{ij}\, x_{t,j} + v_{t,i}$$

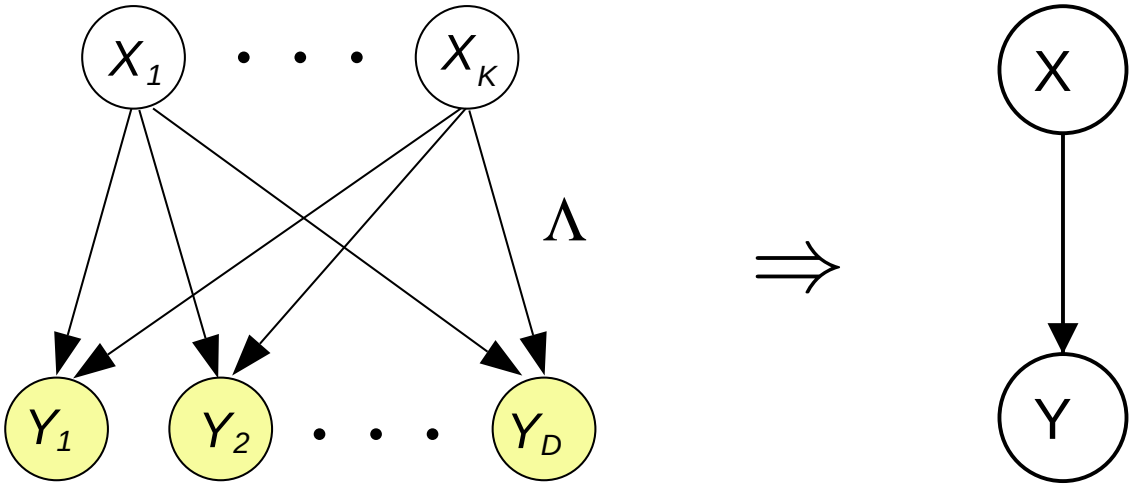
which is, in matrix form: $$\mathbf{y}_t = C\mathbf{x}_t + \mathbf{v}_t$$

State dynamics equation: $$\mathbf{x}_t = A\mathbf{x}_{t-1} + \mathbf{w}_t$$

where $\mathbf{v}$ and $\mathbf{w}$ are uncorrelated zero-mean Gaussian noise vectors.

These models are a.k.a. stochastic linear dynamical systems, Kalman filter models.

From Factor Analysis to State Space Models



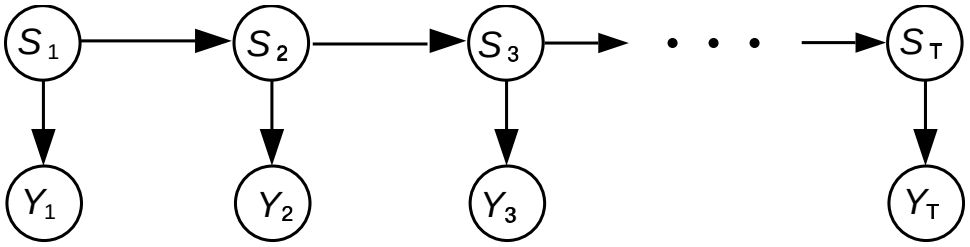
Linear generative model:

$$y_i = \sum_{j=1}^{K} C_{ij}\, x_j + v_i$$

- x_j are independent $\mathcal{N}(0,1)$ Gaussian factors/state variables
- v_i are independent $\mathcal{N}(0,\Psi_{ii})$ Gaussian noise

State-space models are a dynamical generalization of factor analysis where $x_{t,j}$ can depend linearly on $x_{t-1,\ell}$. Also, possibly $K \geq D$ and Ψ not diagonal.

Hidden Markov Models



- Discrete hidden states $\mathbf{s}_t \in \{1 \ldots, K\}$, and observations $\mathbf{y}_t$ (discrete or continuous). Joint probability factorizes:

$$P(\mathbf{s}_1, \ldots, \mathbf{s}_T, \mathbf{y}_1 \ldots, \mathbf{y}_T) = P(\mathbf{s}_1)P(\mathbf{y}_1|\mathbf{s}_1) \prod_{t=2}^{T} P(\mathbf{s}_t|\mathbf{s}_{t-1})P(\mathbf{y}_t|\mathbf{s}_t)$$

- Hidden state sequence controlled by transition matrix T and initial state probabilities π: $P(\mathbf{s}_1 = j) = \pi_j$ $P(\mathbf{s}_{t+1} = j|\mathbf{s}_t = i) = T_{ij}$
- Observations controlled by emission distributions: $E_j(y) = P(\mathbf{y}_t = y|\mathbf{s}_t = j)$
- If observations are also discrete then emission *matrix* is $E_{jk} = P(\mathbf{y}_t = k|\mathbf{s}_t = j)$
- If observations are continuous and Gaussian given the state, then we can think of the HMM as a dynamical generalization of a Mixture of Gaussians.

Learning Latent Variable Models

How can we learn the parameters of latent variable models?

The Expectation Maximization (EM) algorithm

Given a set of observed (visible) variables $\mathbf{y}$, a set of unobserved (hidden / latent / missing) variables $\mathbf{x}$, and model parameters θ, optimize the log likelihood:

$$\mathcal{L}(\theta) = \log p(\mathbf{y}|\theta) = \log \int p(\mathbf{x}, \mathbf{y}|\theta)\, d\mathbf{x}.$$

Using *Jensen's inequality*, for any distribution of hidden states $q(\mathbf{x})$ we have:

$$\mathcal{L} = \log \int q(\mathbf{x}) \frac{p(\mathbf{x}, \mathbf{y}|\theta)}{q(\mathbf{x})} d\mathbf{x} \geq \int q(\mathbf{x}) \log \frac{p(\mathbf{x}, \mathbf{y}|\theta)}{q(\mathbf{x})}\, d\mathbf{x} = \mathcal{F}(q, \theta),$$

defining the $\mathcal{F}(q, \theta)$ functional, which is a lower bound on the log likelihood.

In the EM algorithm, we alternately optimize $\mathcal{F}(q, \theta)$ wrt q and θ, and we can prove that this will never decrease $\mathcal{L}$.

The E and M steps of EM

The lower bound on the log likelihood:

$$\mathcal{F}(q,\theta) = \int q(\mathbf{x}) \log \frac{p(\mathbf{x},\mathbf{y}|\theta)}{q(\mathbf{x})} d\mathbf{x} = \int q(\mathbf{x}) \log p(\mathbf{x},\mathbf{y}|\theta) d\mathbf{x} + \mathcal{H}(q),$$

where $\mathcal{H}(q) = -\int q(\mathbf{x}) \log q(\mathbf{x}) d\mathbf{x}$ is the entropy of q. We iteratively alternate:

E step: optimize $\mathcal{F}(q,\theta)$ wrt the distribution over hidden variables given the parameters:

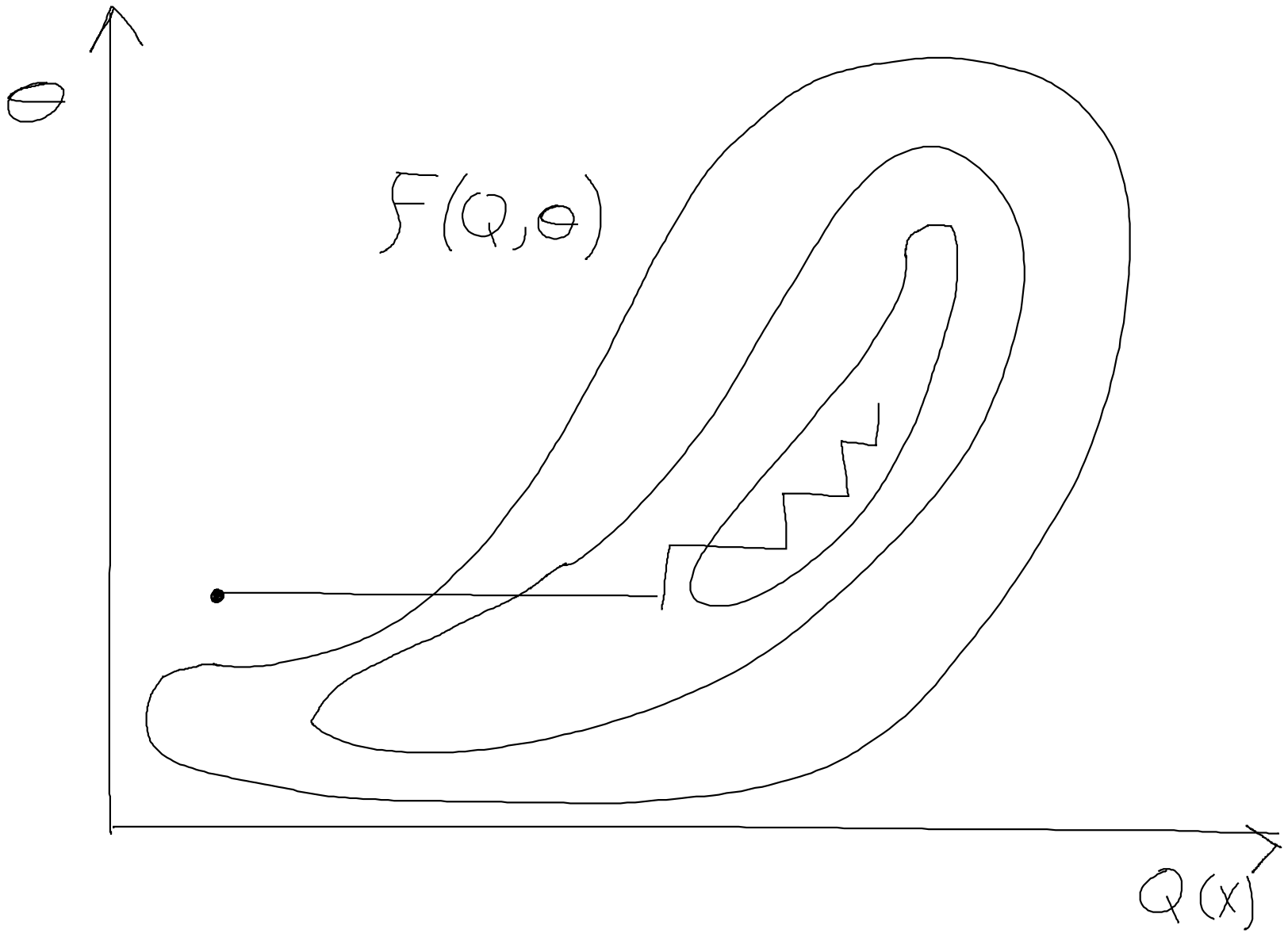
$$q^{(k)}(\mathbf{x}) := \underset{q(\mathbf{x})}{\operatorname{argmax}} \; \mathcal{F}\big(q(\mathbf{x}), \theta^{(k-1)}\big) = p(\mathbf{x}|\mathbf{y}, \theta^{(k-1)}).$$

M step: maximize $\mathcal{F}(q,\theta)$ wrt the parameters given the hidden distribution:

$$\theta^{(k)} := \underset{\theta}{\operatorname{argmax}} \; \mathcal{F}\big(q^{(k)}(\mathbf{x}), \theta\big) = \underset{\theta}{\operatorname{argmax}} \; \int q^{(k)}(\mathbf{x}) \log p(\mathbf{x},\mathbf{y}|\theta) d\mathbf{x},$$

which is equivalent to optimizing the expected complete-data likelihood $p(\mathbf{x},\mathbf{y}|\theta)$, since the entropy of $q(\mathbf{x})$ does not depend on θ.

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



The EM algorithm never decreases the log likelihood

The difference between the cost functions:

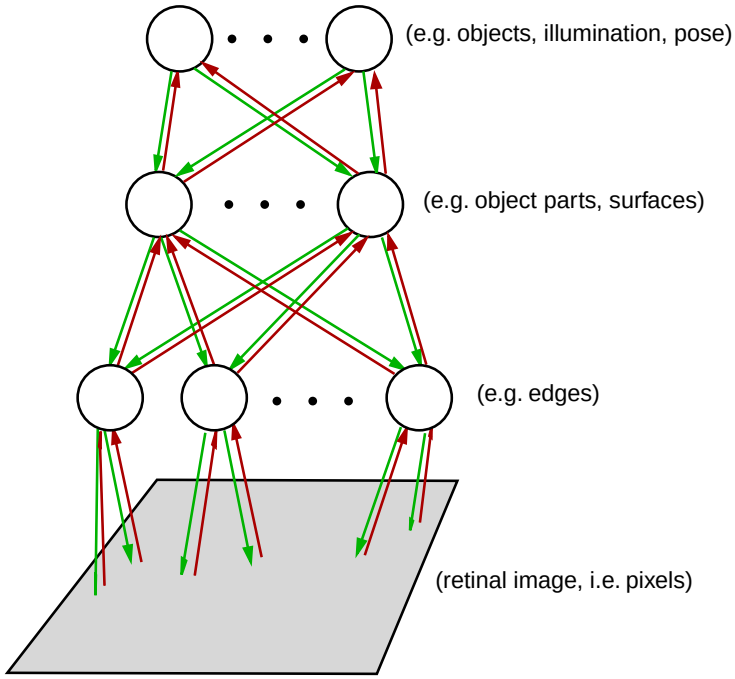
$$\begin{aligned}
\mathcal{L}(\theta) - \mathcal{F}(q,\theta) &= \log p(\mathbf{y}|\theta) - \int q(\mathbf{x}) \log \frac{p(\mathbf{x},\mathbf{y}|\theta)}{q(\mathbf{x})} d\mathbf{x} \\
&= \log p(\mathbf{y}|\theta) - \int q(\mathbf{x}) \log \frac{p(\mathbf{x}|\mathbf{y},\theta)p(\mathbf{y}|\theta)}{q(\mathbf{x})} d\mathbf{x} \\
&= -\int q(\mathbf{x}) \log \frac{p(\mathbf{x}|\mathbf{y},\theta)}{q(\mathbf{x})} d\mathbf{x} = \mathcal{KL}\big(q(\mathbf{x}), p(\mathbf{x}|\mathbf{y},\theta)\big),
\end{aligned}$$

is called the Kullback-Liebler divergence; it is non-negative and only zero if and only if $q(\mathbf{x}) = p(\mathbf{x}|\mathbf{y},\theta)$ (thus this is the E step). Although we are working with a lower bound, the likelihood is still increased in every iteration:

$$\mathcal{L}\big(\theta^{(k-1)}\big) \underset{\text{E step}}{=} \mathcal{F}\big(q^{(k)}, \theta^{(k-1)}\big) \underset{\text{M step}}{\leq} \mathcal{F}\big(q^{(k)}, \theta^{(k)}\big) \underset{\text{Jensen}}{\leq} \mathcal{L}\big(\theta^{(k)}\big),$$

where the first equality holds because of the E step, and the first inequality comes from the M step and the final inequality from Jensen. Usually EM converges to a local optimum of $\mathcal{L}$.

The EM Algorithm: Coupling Perception and Learning



Perception step ("E Step"): find/infer explanations $\mathbf{x}$ that are probable given the observed data, $\mathbf{y}$, and the current generative model, θ. For example, sample from:

$$P(\mathbf{x}|\mathbf{y},\theta) \propto P(\mathbf{x}|\theta)P(\mathbf{y}|\mathbf{x},\theta)$$

This involves combining both top-down prior expectations and bottom-up sensory evidence.

Learning step ("M Step"): modify model parameters to increase the probability of both explanations and data:

$$\theta_{\text{new}} = \text{maximizer of } P(\mathbf{y}|\mathbf{x},\theta)\ P(\mathbf{x}|\theta)$$

This usually involves only **local information**.

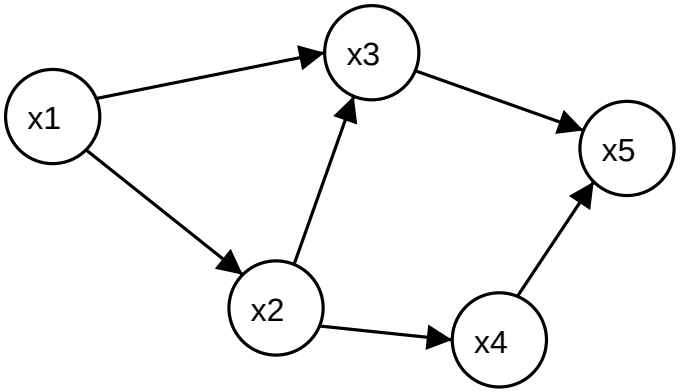
This algorithm is guaranteed to increase the likelihood (Dempster, Laird and Rubin, 1977).

Many many models can be learned using these ideas: e.g. Boltzmann machines, Helmholtz machines, ICA, hidden Markov models, clustering models, dimensionality reduction models, ...

Is the brain a statistical inference engine?

Graphical Models

All the models we have discussed can be represented graphically.
Directed acyclic graph (DAG) where each node corresponds to a random variable.



$$P(\mathbf{x}) = P(\mathbf{x}_1)P(\mathbf{x}_2|\mathbf{x}_1)P(\mathbf{x}_3|\mathbf{x}_1,\mathbf{x}_2) P(\mathbf{x}_4|\mathbf{x}_2)P(\mathbf{x}_5|\mathbf{x}_3,\mathbf{x}_4)$$

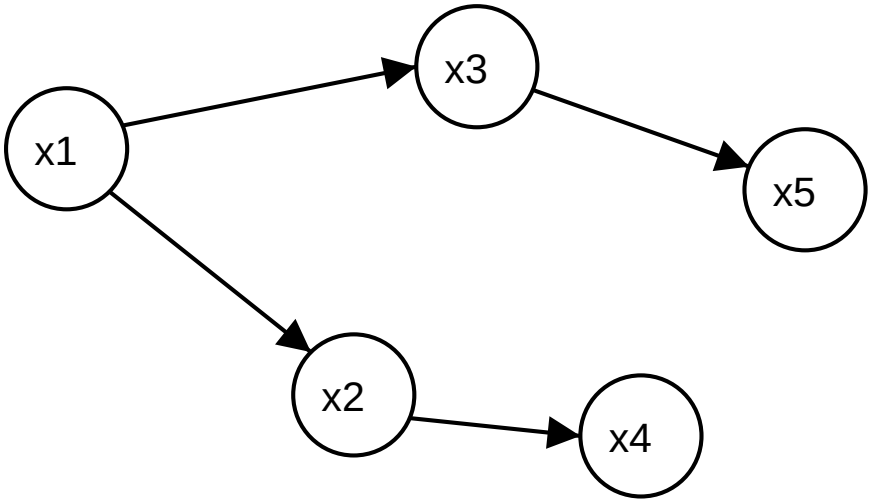
Definitions: *children*, *parents*, *descendents*, *ancestors*
Key quantity: joint probability distribution over nodes: $P(\mathbf{x}) = P(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$
The graph specifies a factorization of this joint pdf: $P(\mathbf{x}) = \prod_i P(\mathbf{x}_i|\text{pa}_i)$
Semantics: Given its parents, each node is *conditionally independent* from its *non-descendents*
Definition: A is *conditionally independent* from B given C if $P(A, B|C) = P(A|C)P(B|C)$ for all A, B, and C s.t. $P(C) \neq 0$.

Also known as Bayesian Networks, Belief Nets and Probabilistic Independence Nets.

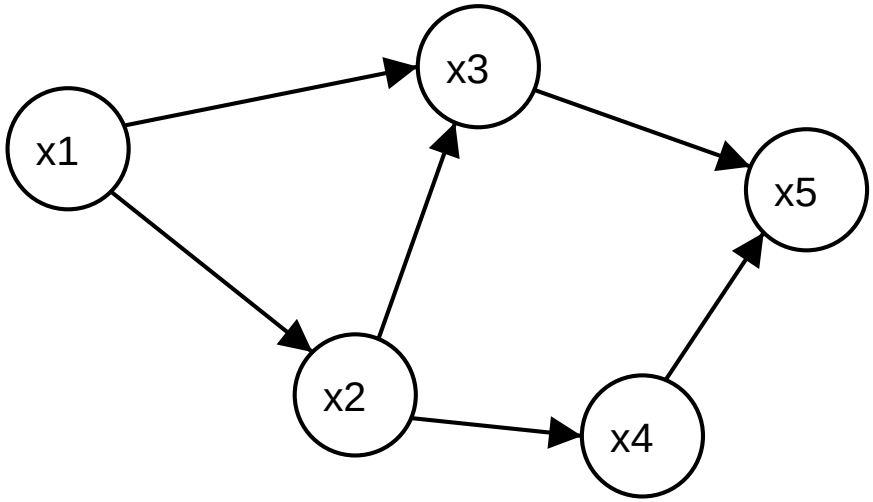
Important advance in statistics and machine learning: not just an intuitive representation of complex multivariate models, but also important for developing local message passing algorithms for inference.

Inference in Graphical Models

Inference is the problem of computing the probability distribution over some unobserved/hidden/latent variable(s) given some observed variable(s), e.g. $P(\mathbf{x}_i|\mathbf{x}_j)$



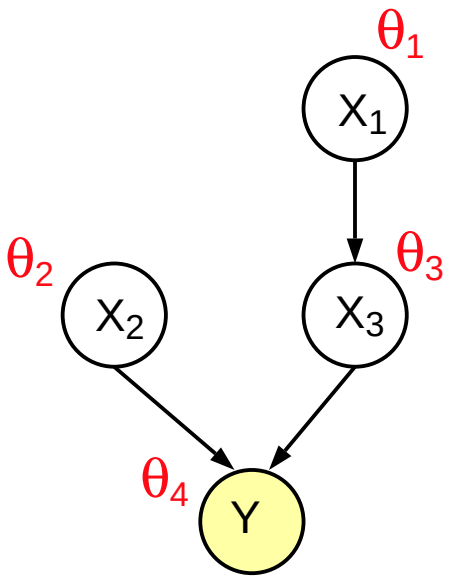
Singly connected nets:
The *belief propagation* algorithm.



Multiply connected nets:
The *junction tree* algorithm.

These are efficient message passing algorithms for applying Bayes rule using the conditional independence relationships implied by the graphical model.

EM in Graphical Models: Learning with Hidden Variables



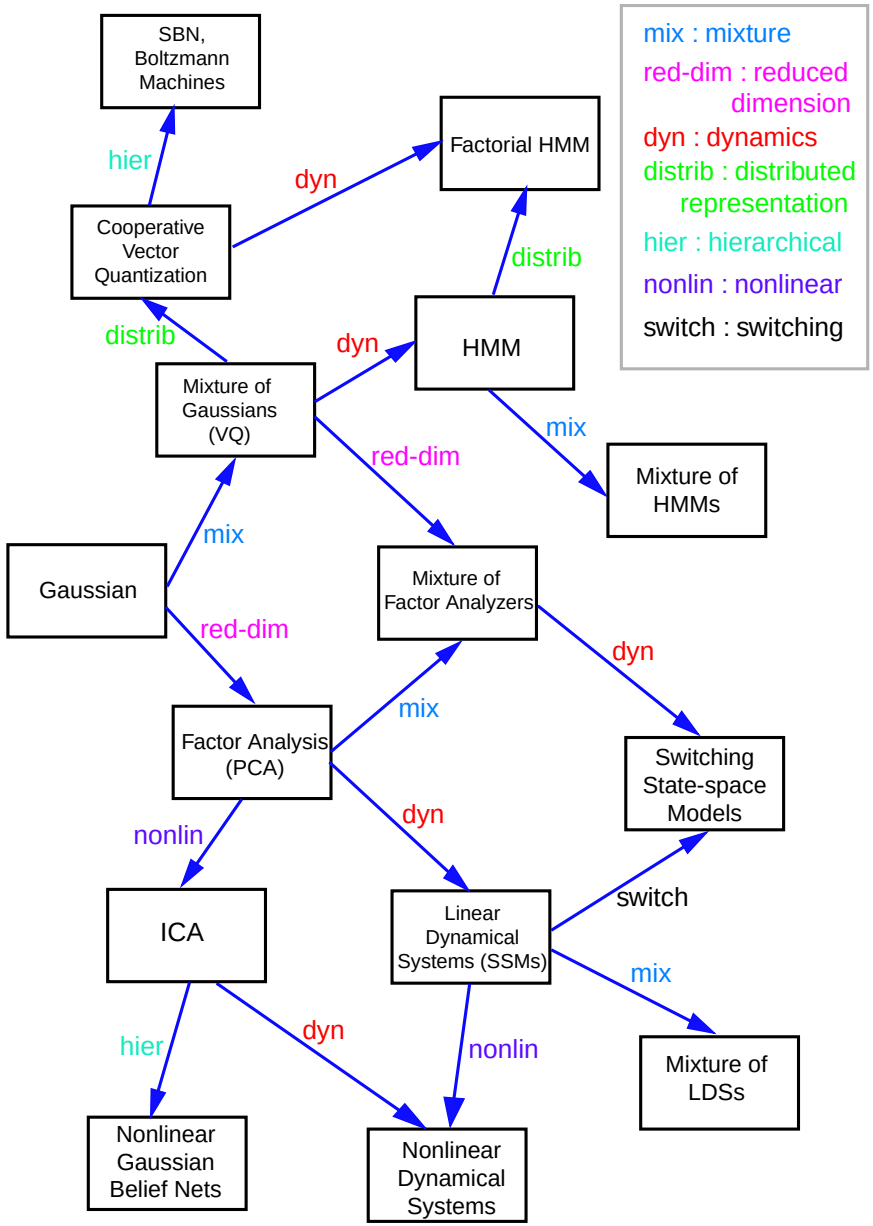
Assume a model parameterised by θ with observable variables $\mathbf{y}$ and hidden variables $\mathbf{x}$

Goal: maximise log likelihood of observables.

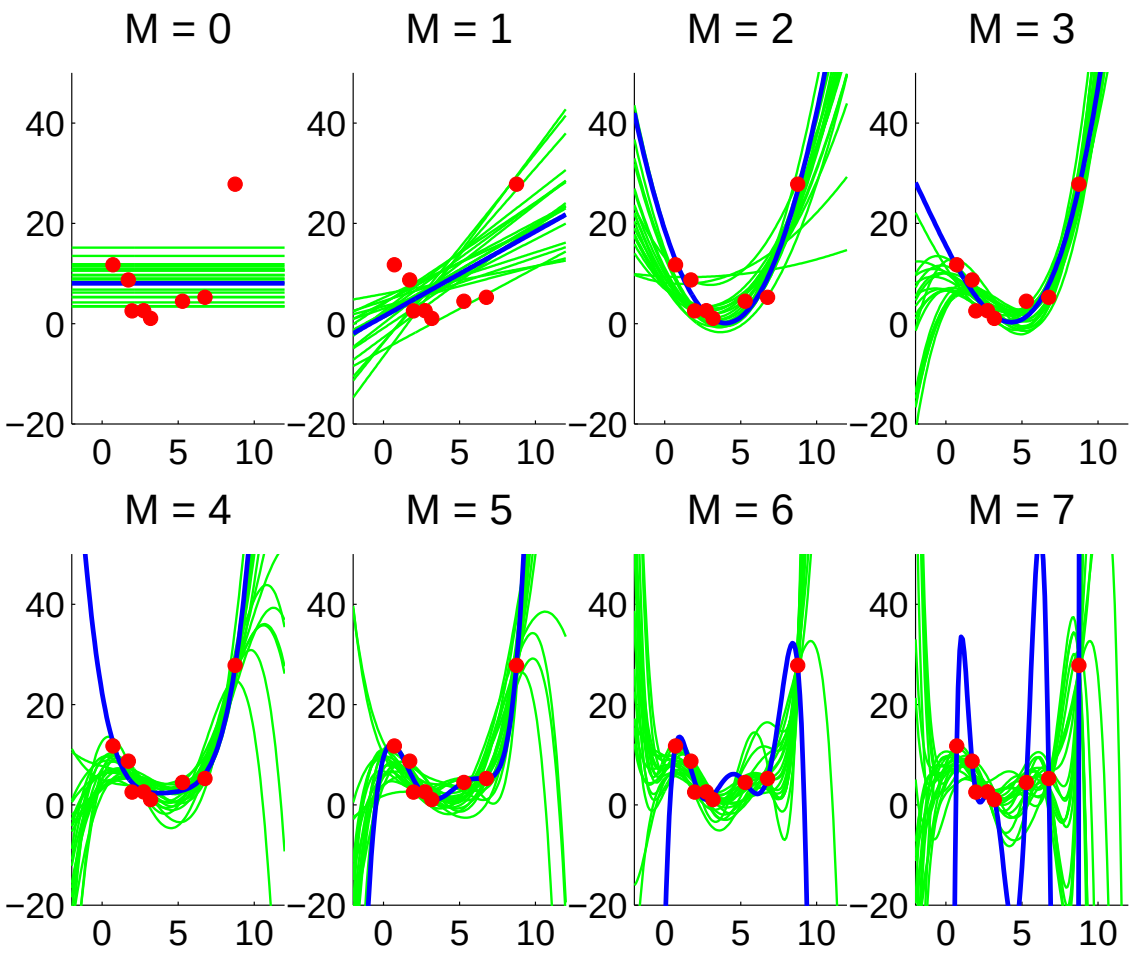
$$\mathcal{L}(\theta) = \ln P(\mathbf{y}|\theta) = \ln \int P(\mathbf{y}, \mathbf{x}|\theta)\, d\mathbf{x}$$

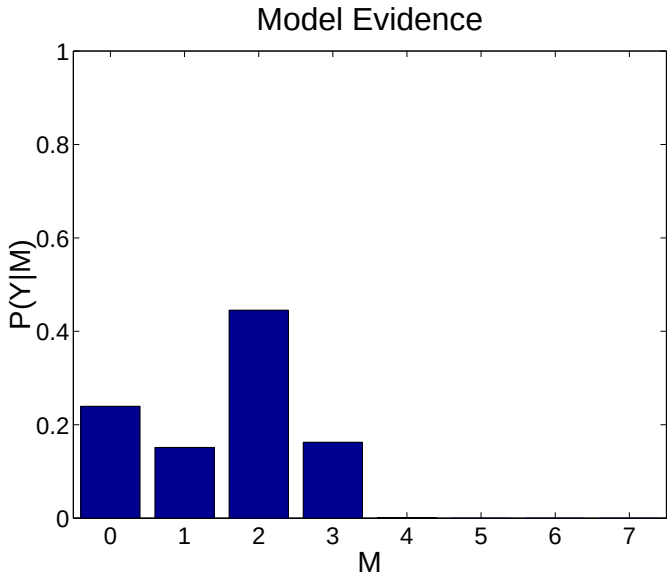
- **E-step**: Infer $P(\mathbf{x}|\mathbf{y}, \theta_{\text{old}})$. This requires solving the *inference* problem: finding "explanations", $\mathbf{x}$, for the data, $\mathbf{y}$ given the current model θ. This step can make use of **belief propagation** or **junction tree** inference algorithms.
  Intuition: The **E-step** "fills in" values for the hidden variables. With no hidden variables, the likelihood is a simpler function of the parameters.
- **M-step**: find θ_{new} using complete data learning.
  Intuition: The **M-step** for the parameters at each node can be computed independently, and depends only on the values of the variables at that node and its parents.

A Generative Model for Generative Models



Back to Bayesian Learning...





Computing Marginal Likelihoods can be Computationally Intractable

$$p(\mathbf{y}|m_i) = \int p(\mathbf{y}|\boldsymbol{\theta}, m_i)\, p(\boldsymbol{\theta}|m_i)\, d\boldsymbol{\theta}$$

- This can be a very **high dimensional integral**.

- The presence of **latent variables** results in additional dimensions that need to be marginalized out.

$$p(\mathbf{y}|m_i) = \int \int p(\mathbf{y}, \mathbf{x}|\boldsymbol{\theta}, m_i)\, p(\boldsymbol{\theta}|m_i)\, d\mathbf{x}\, d\boldsymbol{\theta}$$

- The likelihood term can be **complicated**.

Approximations to the Marginal Likelihood

- **BIC/MDL Penalty**: crude asymptotic approximation for identifiable models: $(d/2)\log N$
- **Laplace's Approximation**: approximate parameter posterior by a Gaussian with mean at a local maximum of the posterior and covariance matrix based on curvature (i.e. Hessian) of the log posterior.
- **Sampling Methods**: Monte carlo methods for estimating marginal likelihoods based on thermodynamic integration / annealed importance sampling, etc.
- **Variational Methods**: maximize a lower bound on the marginal likelihood.

Approximate Inference Methods

- **Sampling**: approximate true distribution over hidden variables by drawing random samples, e.g. Gibbs sampling
- **Linearization**: approximate the transformation on the hidden variables by one which keeps the form of the distribution closed (e.g. Gaussians and linear), e.g. Extended Kalman Filtering
- **Recognition Models**: approximate the true distribution with an approximation that can be computed easily/quickly by an explicit *bottom-up* inference model/network, e.g. Helmholtz Machines
- **Variational Methods**: approximate the true distribution with an approximate form that is tractable; maximise a lower bound on the likelihood with respect to free parameters in this approximation

Lower Bounding the Marginal Likelihood

Variational Bayesian Learning

Let the latent variables be $\mathbf{x}$, data $\mathbf{y}$ and the parameters $\boldsymbol{\theta}$.
We can lower bound the marginal likelihood (Jensen's inequality):

$$\begin{aligned}
\ln p(\mathbf{y}|m) &= \ln \int p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta}|m)\, d\mathbf{x}\, d\boldsymbol{\theta} \\
&= \ln \int q(\mathbf{x}, \boldsymbol{\theta}) \frac{p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta}|m)}{q(\mathbf{x}, \boldsymbol{\theta})}\, d\mathbf{x}\, d\boldsymbol{\theta} \\
&\geq \int q(\mathbf{x}, \boldsymbol{\theta}) \ln \frac{p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta}|m)}{q(\mathbf{x}, \boldsymbol{\theta})}\, d\mathbf{x}\, d\boldsymbol{\theta}.
\end{aligned}$$

Use a simpler, factorised approximation to $q(\mathbf{x}, \boldsymbol{\theta}) \approx q_{\mathbf{x}}(\mathbf{x}) q_{\boldsymbol{\theta}}(\boldsymbol{\theta})$:

$$\begin{aligned}
\ln p(\mathbf{y}|m) &\geq \int q_{\mathbf{x}}(\mathbf{x}) q_{\boldsymbol{\theta}}(\boldsymbol{\theta}) \ln \frac{p(\mathbf{y}, \mathbf{x}, \boldsymbol{\theta}|m)}{q_{\mathbf{x}}(\mathbf{x}) q_{\boldsymbol{\theta}}(\boldsymbol{\theta})}\, d\mathbf{x}\, d\boldsymbol{\theta} \\
&= \mathcal{F}_m(q_{\mathbf{x}}(\mathbf{x}), q_{\boldsymbol{\theta}}(\boldsymbol{\theta}), \mathbf{y}).
\end{aligned}$$

The Variational Bayesian EM algorithm

EM for MAP estimation

Goal: maximize $p(\boldsymbol{\theta}|\mathbf{y}, m)$ w.r.t. $\boldsymbol{\theta}$

E Step: compute

$$q_{\mathbf{x}}^{(t+1)}(\mathbf{x}) = p(\mathbf{x}|\mathbf{y}, \boldsymbol{\theta}^{(t)})$$

M Step:

$$\boldsymbol{\theta}^{(t+1)} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \int q_{\mathbf{x}}^{(t+1)}(\mathbf{x}) \ln p(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}) \, d\mathbf{x}$$

Variational Bayesian EM

Goal: lower bound $p(\mathbf{y}|m)$

VB-E Step: compute

$$q_{\mathbf{x}}^{(t+1)}(\mathbf{x}) = p(\mathbf{x}|\mathbf{y}, \bar{\boldsymbol{\phi}}^{(t)})$$

VB-M Step:

$$q_{\boldsymbol{\theta}}^{(t+1)}(\boldsymbol{\theta}) \propto \exp\left[\int q_{\mathbf{x}}^{(t+1)}(\mathbf{x}) \ln p(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta}) \, d\mathbf{x}\right]$$

Properties:

- Reduces to the EM algorithm if $q_{\boldsymbol{\theta}}(\boldsymbol{\theta}) = \delta(\boldsymbol{\theta} - \boldsymbol{\theta}^*)$.
- $\mathcal{F}_m$ increases monotonically, and incorporates the model complexity penalty.
- Analytical parameter distributions (but not constrained to be Gaussian).
- VB-E step has same complexity as corresponding E step.
- We can use the junction tree, belief propagation, Kalman filter, etc, algorithms in the VB-E step of VB-EM, but **using expected natural parameters**, $\bar{\boldsymbol{\phi}}$.

Variational Bayesian EM

The Variational Bayesian EM algorithm has been used to approximate Bayesian learning in a wide range of models such as:

- probabilistic PCA and factor analysis
- mixtures of Gaussians and mixtures of factor analysers
- hidden Markov models
- state-space models (linear dynamical systems)
- ICA
- discrete graphical models...

The main advantage is that it can be used to **automatically do model selection** and does not suffer from overfitting to the same extent as ML methods do.

Also it is about as computationally demanding as the usual EM algorithm.

See: `www.variational-bayes.org`

demo: `mixture of Gaussians`
demo: `vbhmm_demo`

Another example: Independent Components Analysis

Blind Source Separation: 5 $\times$ 100 msec speech and music sources linearly mixed to produce 11 signals (microphones)

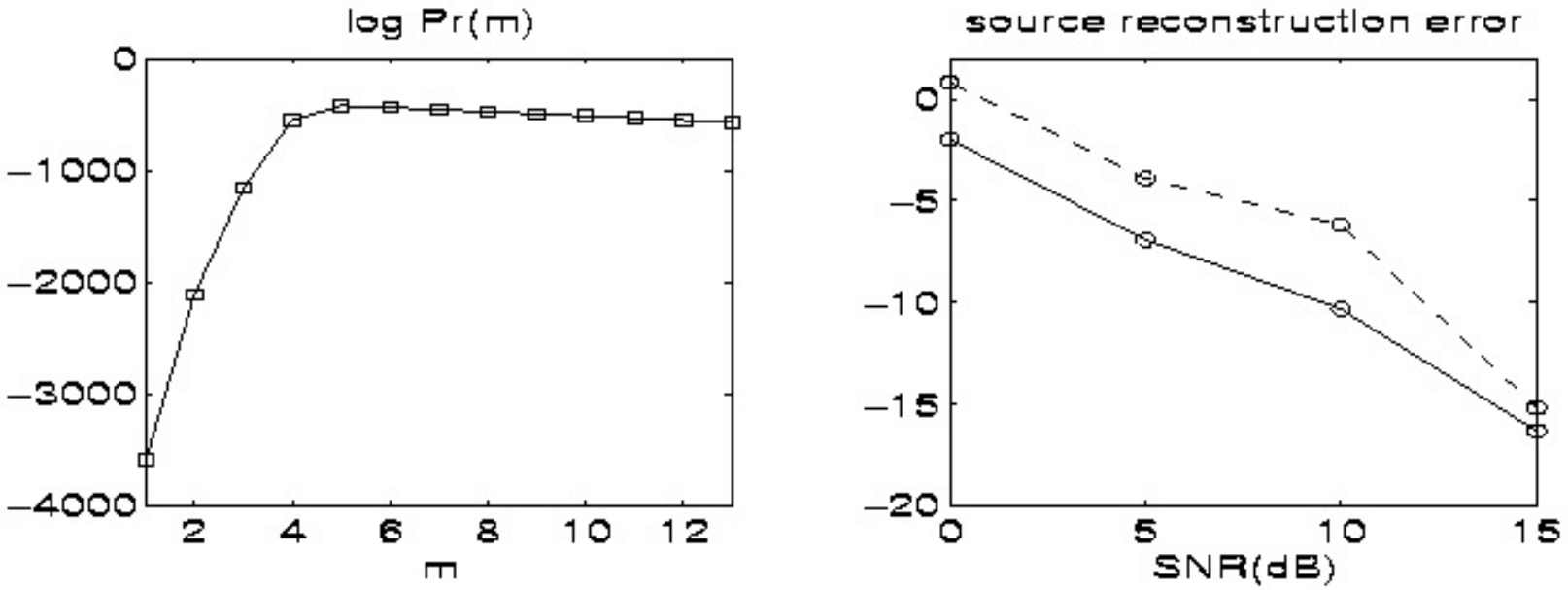


Figure 2. Application of VB to blind source separation algorithm (see text).

from Attias (2000)

Summary

- Unsupervised vs Supervised vs Reinforcement Learning
- Model Selection, Overfitting, Occam's Razor, and the Marginal Likelihood
- Models
  - Factor Analysis, PCA, and Network Interpretations
  - Mixture of Gaussians and k-means
  - The EM Algorithm as a general way of fitting models with hidden variables
  - ICA as nonlinear Factor Analysis
  - From Factor Analysis to state-space models
  - From Mixture of Gaussians to hidden Markov models
- Graphical Models, Inference and EM
- Approximate inference methods
- The Variational Bayesian EM Algorithm

Topics I didn't cover but should have...

- Choosing priors: subjective, objective and empirical Bayesian approaches
- Infinite models and non-parametric Bayesian approaches
- Gaussian processes and other methods for supervised Bayesian learning.
- Other modern approximate inference methods: e.g. loopy belief propagation, expectation propagation, fractional BP

Selected References

Graphical Models and the EM algorithm:

⋆ *Learning in Graphical Models* (1998) Edited by M.I. Jordan. Dordrecht: Kluwer Academic Press. Also available from MIT Press (paperback).

- **Motivation for Bayes Rule**: "Probability Theory - the Logic of Science," E.T.Jaynes, http://www.math.albany.edu:8008/JaynesBook.html
- **EM:**

  Baum, L., Petrie, T., Soules, G., and Weiss, N. (1970). A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics*, 41:164–171;

  Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statistical Society Series B*, 39:1–38;

  Neal, R. M. and Hinton, G. E. (1998). A new view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*.
- **Factor Analysis and PCA:**

  Mardia, K.V., Kent, J.T., and Bibby J.M. (1979). *Multivariate Analysis* Academic Press, London

  Roweis, S. T. (1998). EM algorthms for PCA and SPCA. NIPS98

  Ghahramani, Z. and Hinton, G. E. (1996). The EM algorithm for mixtures of factor analyzers.

Technical Report CRG-TR-96-1 [http://www.gatsby.ucl.ac.uk/∼zoubin/papers/tr-96-1.ps.gz]
Department of Computer Science, University of Toronto.

Tipping, M. and Bishop, C. (1999). Mixtures of probabilistic principal component analyzers. *Neural Computation*, 11(2):435–474.

- **Belief propagation:**

Kim, J.H. and Pearl, J. (1983) A computational model for causal and diagnostic reasoning in inference systems. In *Proc of the Eigth International Joint Conference on AI*: 190-193;

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA.

- **Junction tree:** Lauritzen, S. L. and Spiegelhalter, D. J. (1988).
Local computations with probabilities on graphical structures and their application to expert systems. *J. Royal Statistical Society B*, pages 157–224.

Other graphical models:

- **ICA:**

Comon, P. (1994). Independent component analysis: A new concept.
Signal Processing, 36:287–314;

Bell, A. J. and Sejnowski, T. J. (1995). An information-maximization approach to blind separation and blind deconvolution.
Neural Computation, 7(6):1129–1159.

⋆ Roweis, S.T and Ghahramani, Z. (1999) A unifying review of linear Gaussian models. *Neural Computation* **11**(2): 305–345.

Approximate Inference & Learning

- **MCMC:** Neal, R. M. (1993). Probabilistic inference using Markov chain monte carlo methods. Technical Report CRG-TR-93-1, Department of Computer Science, University of Toronto.
- **Extended Kalman Filtering:** Goodwin, G. and Sin, K. (1984). *Adaptive filtering prediction and control*. Prentice-Hall.
- **Recognition Models:**

  Hinton, G. E., Dayan, P., Frey, B. J., and Neal, R. M. (1995). The wake-sleep algorithm for unsupervised neural networks. *Science*, 268:1158–1161.

  Dayan, P., Hinton, G. E., Neal, R., and Zemel, R. S. (1995) The Helmholtz Machine . *Neural Computation*, 7, 1022-1037.

- **Occam's Razor and Laplace Approximation:**

  Jefferys, W.H., Berger, J.O. (1992)
  Ockham's Razor and Bayesian Analysis. *American Scientist* **80**:64-72;

  MacKay, D.J.C. (1995) Probable Networks and Plausible Predictions - A Review of Practical Bayesian Methods for Supervised Neural Networks.
  Network: Computation in Neural Systems. **6** 469-505

Rasmussen, C. E and Ghahramani, Z. (2001) Occam's Razor. Advances in Neural Information Systems 13, MIT Press.

- **BIC:** Schwarz, G. (1978). Estimating the dimension of a model.
  Annals of Statistics **6**:461-464.

- **MDL and MML:** Wallace, C.S. and Freeman, P.R. (1987) Estimation and inference by compact coding. *J. of the Royal Stat. Society series B* **49**(3): 240-265;
  J. Rissanen (1987)

- **Bayesian Gibbs sampling software:**
  The BUGS Project – http://www.iph.cam.ac.uk/bugs/

- **Variational Bayesian Learning:**

  Hinton, G. E. and van Camp, D. (1993) Keeping Neural Networks Simple by Minimizing the Description Length of the Weights. In *Sixth ACM Conference on Computational Learning Theory*, Santa Cruz.

  Waterhouse, S., MacKay, D., and Robinson, T. (1995) Bayesian methods for Mixtures of Experts. NIPS95.
  (See also several unpublished papers by MacKay).

  Barber D., Bishop C. M., (1998) Ensemble Learning for MultiLayer Networks. Advances in Neural Information Processing Systems 10.

  Bishop, C.M. (1999) Variational PCA. In *Proc. Ninth Int. Conf. on Artificial Neural Networks ICANN99* **1**:509 - 514.

Attias, H. (1999). Inferring parameters and structure of latent variable models by variational Bayes. In *Proc. 15th Conf. on Uncertainty in Artificial Intelligence*;

Ghahramani, Z. and Beal, M.J. (2000) Variational inference for Bayesian mixtures of factor analysers. In Neural Information Processing Systems 12.

Ghahramani, Z. and Beal, M.J. (2000) Propagation algorithms for variational Bayesian learning. In Neural Information Processing Systems 13

Beal, M.J. and Ghahramani, Z. (2003) The Variational Bayesian EM Algorithm for Incomplete Data: with Applicatin to Scoring Graphical Model Structures. Bayesian Statistics 7

See also recent papers on my website: `http://www.gatsby.ucl.ac.uk/~zoubin`

Resources

- **Conferences**

  NIPS - Neural Information Processing Systems - http://www.nips.cc
  UAI - Uncertainty in AI
  AIStats - Workshop on AI and Statistics
  ICML - Int Conf on Machine Learning
  COLT - Conference on Learning Theory

- **Journals**

  Journal of Machine Learning Research - http://www.jmlr.org
  Neural Computation
  Network: Computation in Neural Systems

- **Searching for Online Papers**

  Citeseer - online papers, citation searches
  JSTOR - older scanned in papers