Unsupervised Learning

Nonlinear, Distributed and Hierarchical Models

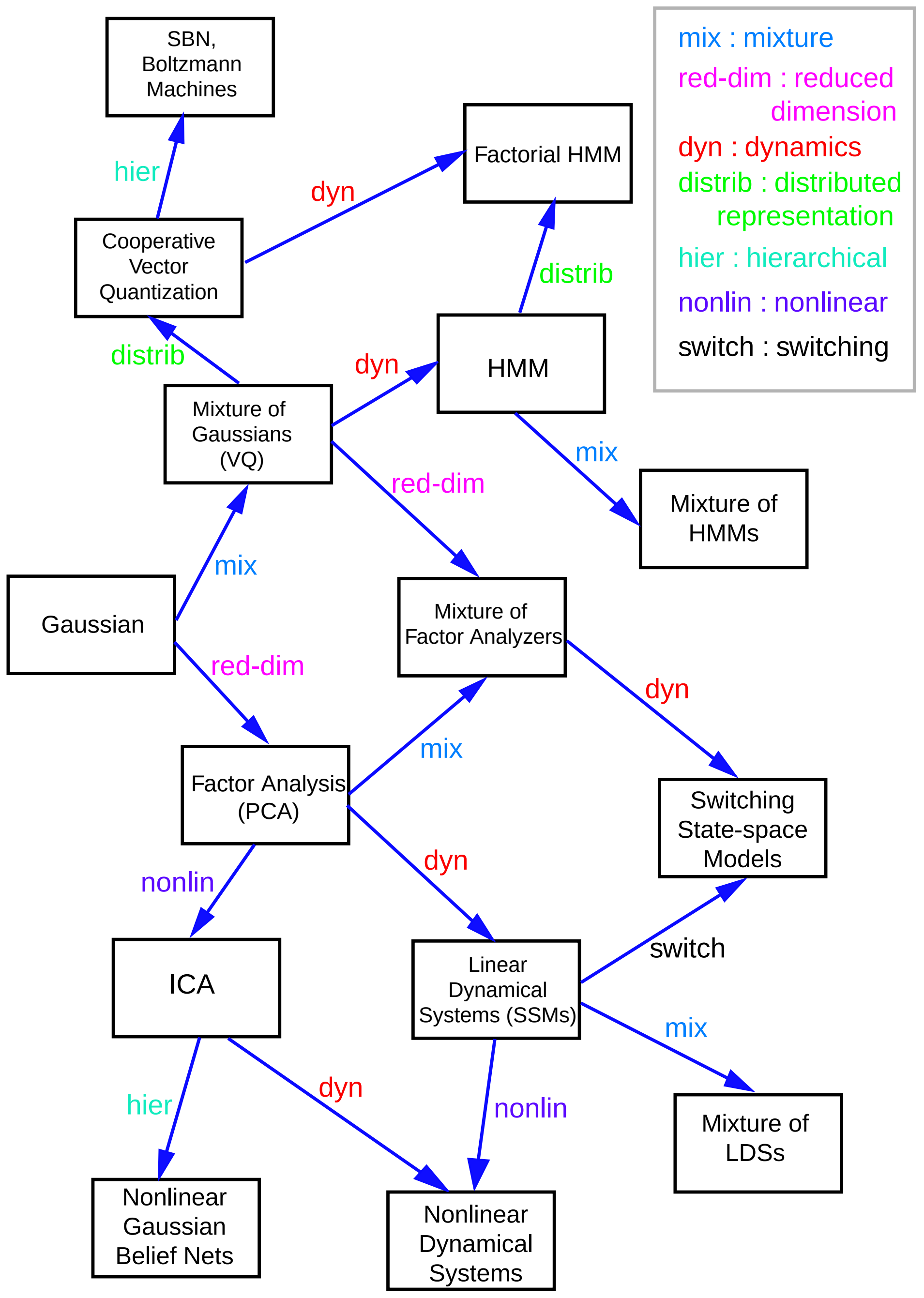
Maneesh Sahani

`maneesh@gatsby.ucl.ac.uk`

Gatsby Computational Neuroscience Unit, and
MSc in Intelligent Systems, Dept Computer Science
University College London

Term 1, Autumn 2006

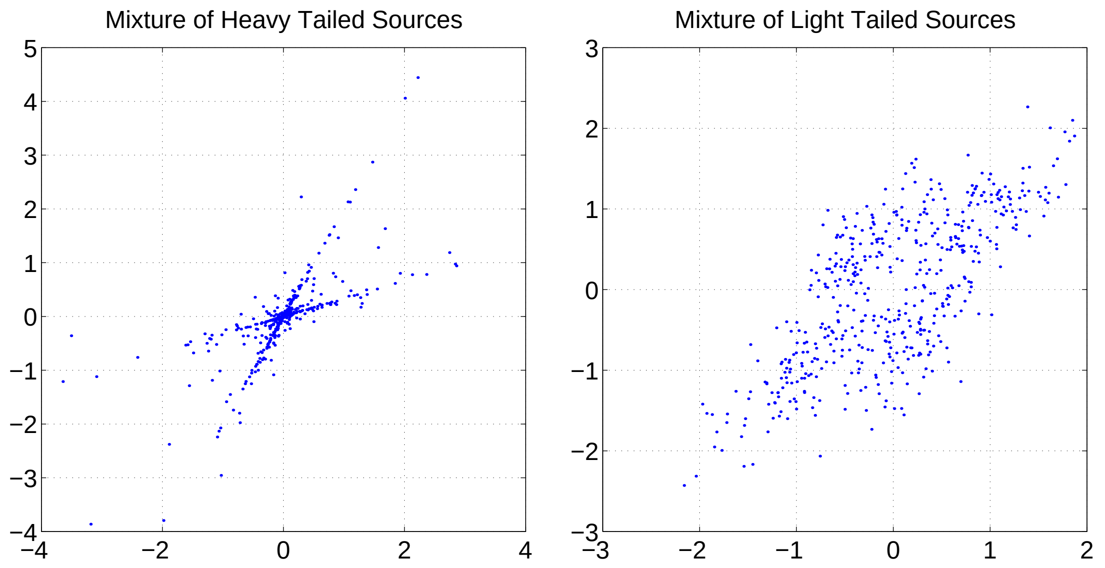
A Generative Model for Generative Models

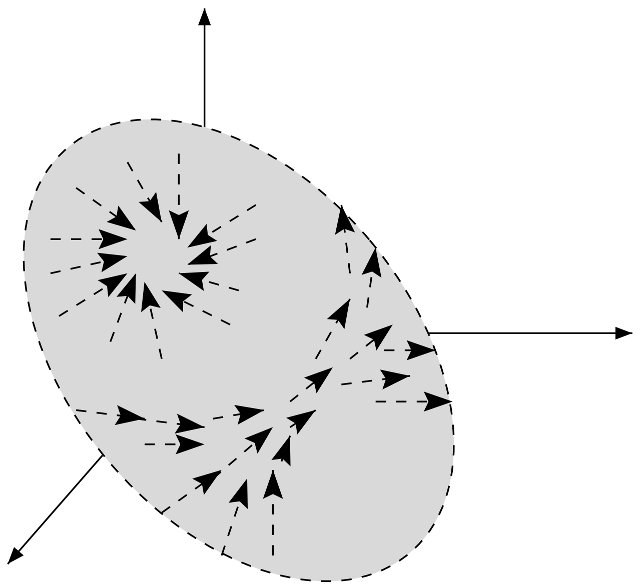


See Roweis & Ghahramani (1999). A Unifying Review of Linear Gaussian Models. *Neural Comput.* **11**(2).

Non-linear/Non-Gaussian Models

Linear-Gaussian models are limited. . .

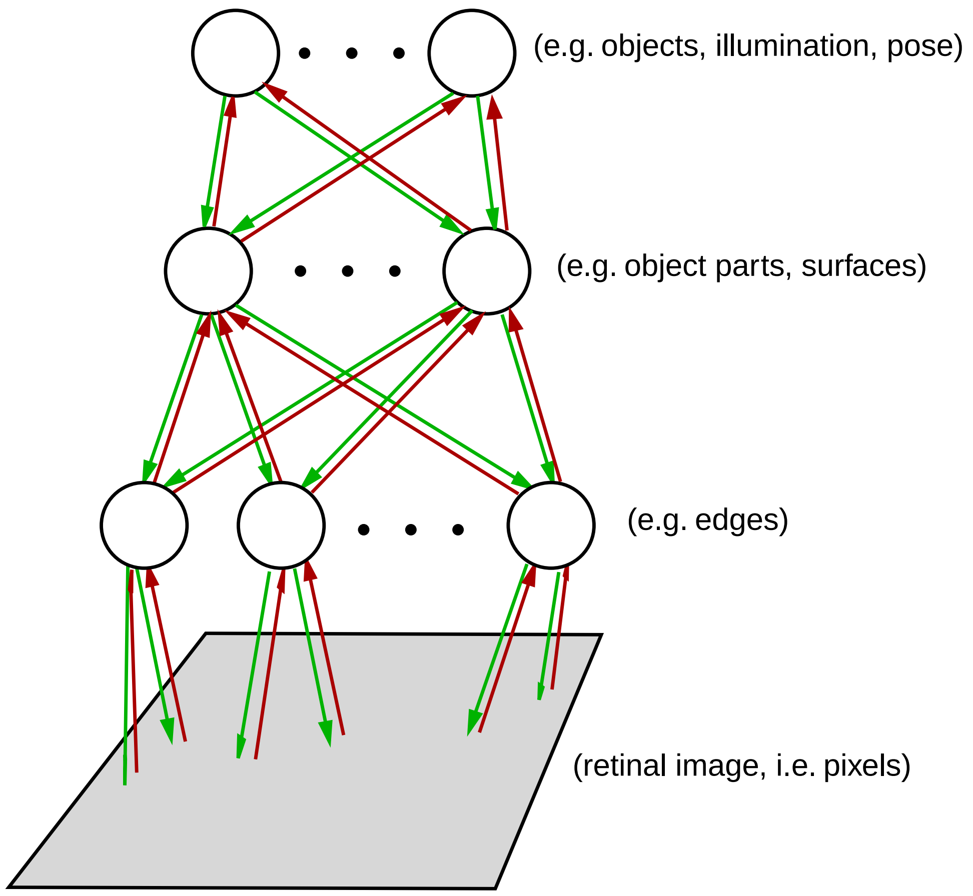




. . . and discrete approximations are not always appropriate.

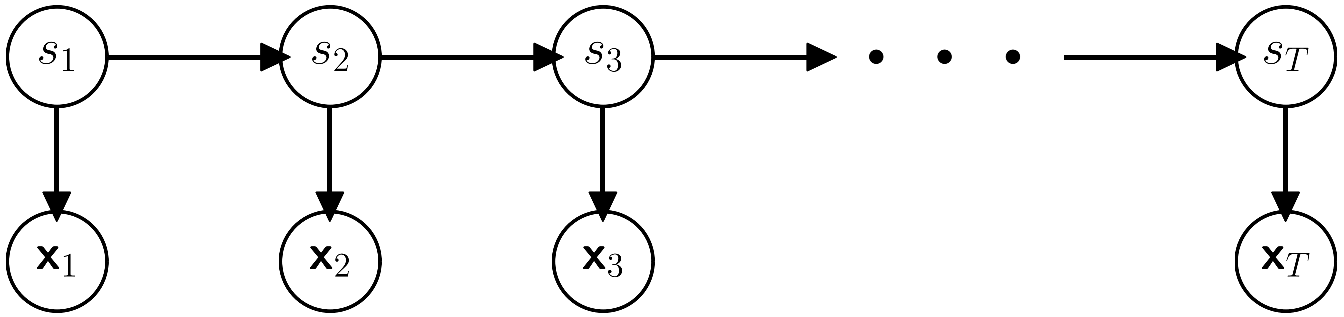
Hierarchical Models

Many generative processes can be naturally described at different levels of detail.


(e.g. objects, illumination, pose)
(e.g. object parts, surfaces)
(e.g. edges)
(retinal image, i.e. pixels)

Biology seems to have developed hierarchical representations.

Distributed Representations



Consider a hidden Markov model. To capture N bits of information about the history of the sequence, an HMM *requires* $K = 2^N$ *states!*

In a *distributed representation* each data point is represented by a vector of (discrete or continous) attibutes. Some attributes might be *latent* (i.e. hidden).

> For example, you could cluster an electorate into `Labour`, `Tory`, `Lib-Dem` and `Undecided`, but this is **not** a distributed representation since each person is described by a single 4-valued discrete variable. A distributed representation might be: (`Tory`, `Single`, `Black`, `Female`, `18-35 years old`, `City-dweller`, `Liberal`, `Procedural`). We might use such a representation to model voting preferences.

These attributes resemble factors, but may be discrete (and non-Gaussian), and may outnumber the observed dimensions (say voting preference). Any dynamics may be independent (FHMM) or correlated (DBN).

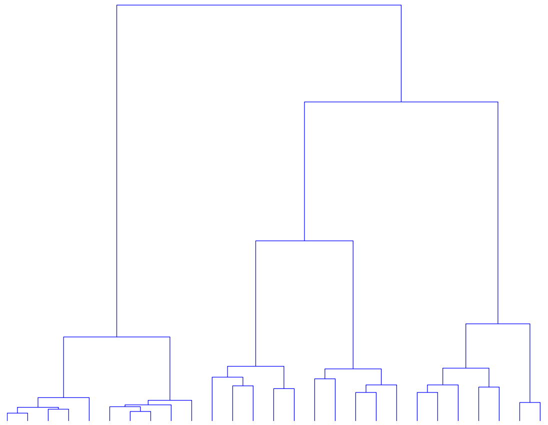
Some more complex generative unsupervised learning methods

- Hierarchical clustering: clustering algorithms in which clusters are organised hierarchically
- Nonlinear dimensionality reduction methods: nonlinear generalizations of PCA and factor analysis
- Factorial hidden Markov models and dynamic Bayesian networks: time series models with distributed representations
- Nonlinear dynamical systems
- Independent components analysis (ICA): linear factor models with non-Gaussian factors
- Boltzmann machines: undirected model for binary data with binary latent variables
- Sigmoid Belief networks: directed (neural-netork-like) model for binary data

Hierarchical Clustering

(See Duda and Hart, 1973)

```
Data $\mathcal{D} = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(n)}\}$
Initialise number of clusters $c = n$
Initialise $\mathcal{D}_i = \{\mathbf{x}^{(i)}\}$ for $i = 1, \ldots, c$
while $c > 1$ do
    Find nearest pair of clusters $\mathcal{D}_i$ and $\mathcal{D}_j$
    Merge $\mathcal{D}_i \leftarrow \mathcal{D}_i \cup \mathcal{D}_j$, Delete $\mathcal{D}_j$, $c \leftarrow c - 1$
end while
```



Distance Measures:

$$\begin{aligned} d_{\min}(\mathcal{D}_i, \mathcal{D}_j) &= \min_{\mathbf{x}\in\mathcal{D}_i, \mathbf{x}'\in\mathcal{D}_j} \|\mathbf{x} - \mathbf{x}'\| && \text{nearest neighbour} \\ d_{\max}(\mathcal{D}_i, \mathcal{D}_j) &= \max_{\mathbf{x}\in\mathcal{D}_i, \mathbf{x}'\in\mathcal{D}_j} \|\mathbf{x} - \mathbf{x}'\| && \text{furthest neighbour} \\ d_{\text{avg}}(\mathcal{D}_i, \mathcal{D}_j) &= \frac{1}{n_i n_j} \sum_{\mathbf{x}\in\mathcal{D}_i} \sum_{\mathbf{x}'\in\mathcal{D}_j} \|\mathbf{x} - \mathbf{x}'\| && \text{average distance} \\ d_{\text{mean}}(\mathcal{D}_i, \mathcal{D}_j) &= \|\mathbf{m}_i - \mathbf{m}_j\| && \text{centre of mass} \end{aligned}$$

Hierarchical clustering is very widely used, e.g. in bioinformatics, because it is often natural to think of data points at multiple level of granularity, or as having been generated by an evolutionary process

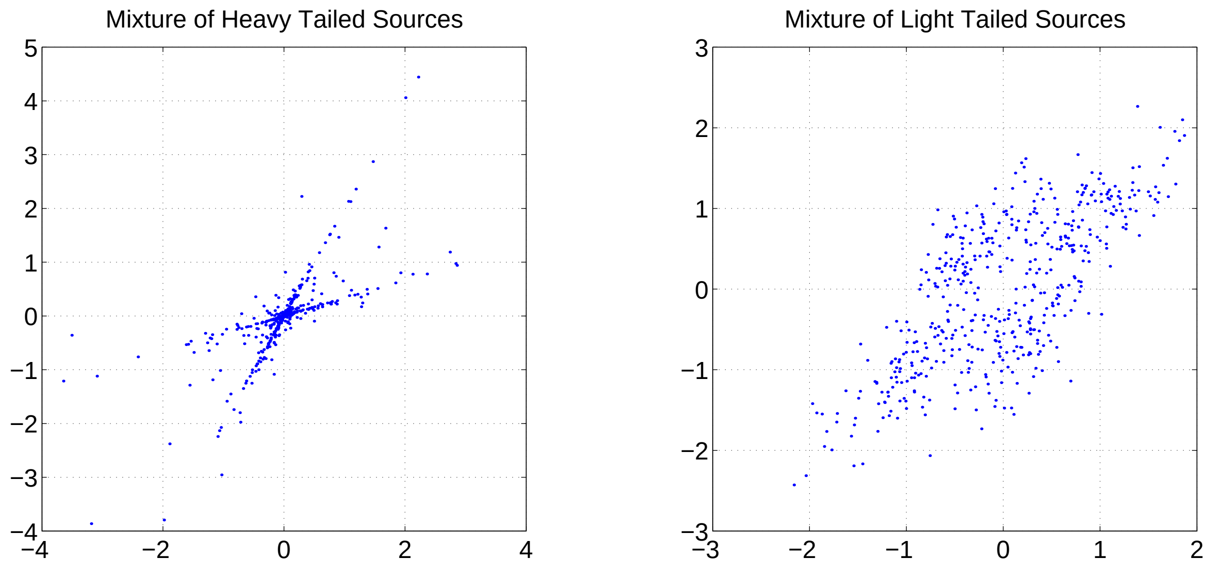
Nonlinear Dimensionality Reduction

There are many ways of generalising PCA and FA models to deal with data which lies on a nonlinear manifold:

– Principal curves
– Autoencoders
– Generative topographic mappings (GTM) and Kohonen self-organising maps (SOM)
– Density networks
– Multi-dimensional scaling (MDS)
– Isomap
– Locally linear embedding (LLE)

Unfortunately, we don't have time to cover these methods in the course...

Independent Components Analysis

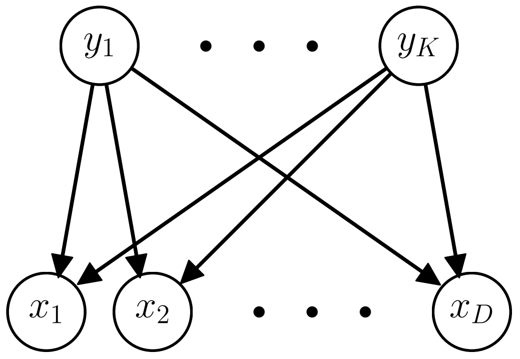


These distributions are generated by linearly combining (or mixing) two *non-Gaussian* sources.

- The ICA graphical model is identical to Factor Analysis:

$$x_d = \sum_{k=1}^{K} \Lambda_{dk}\, y_k + \epsilon_d$$

  with $y_k \sim P_y$ non-Gaussian.



$$\begin{tikzcd} y_1 \arrow[d] \arrow[dr] \arrow[drrr] & \cdots & & y_K \arrow[dlll] \arrow[dll] \arrow[d] \\ x_1 & x_2 & \cdots & x_D \end{tikzcd}$$

- Alternatively, we can consider Gaussian factors passed through a non-linearity before mixing, i.e., $y_k = g(z_k)$.

BUT:

- Well-posed even with $K \geq D$ (e.g., $K = D = 2$ above).
- With non-zero noise, MAP inference is non-linear, and the full posterior is non-Gaussian.
- This makes making exact learning difficult for most P_y.

Infomax ICA

- The special case of $K = D$, and zero observation noise has been studied extensively (standard infomax ICA, c.f. PCA):

$$\mathbf{x} = \Lambda \mathbf{y} \qquad \text{which implies} \qquad \mathbf{y} = W\mathbf{x} \qquad \text{where} \qquad W = \Lambda^{-1}$$

  where $\mathbf{y}$ are the independent components (factors), $\mathbf{x}$ are the observations, and W is the unmixing matrix.

- The likelihood can be written in terms of W:

$$P(\mathbf{x}|W) = |W| \prod_k P_y(\underbrace{[W\mathbf{x}]_k}_{y_k})$$

  where p_y is marginal probability distribution of factors.

- Equivalently, consider $z = \text{cdf}(y) = g(y)$; then P_z is uniform. Maximising $P(z)$ is then equivalent to making z as uniformly distributed as possible

$$\Rightarrow \text{max } \mathbf{H}[z] \Rightarrow \text{max } \mathbf{H}[z] - \mathbf{H}[z|x] \Rightarrow \text{max } \mathbf{I}[z;x].$$

  Thus infomax.

- Learning by gradient ascent:

$$\Delta W \propto W^{-T} - g(\mathbf{y})\mathbf{x}^{\mathsf{T}} \qquad \text{or natural gradient} \qquad \Delta W = (I - g(\mathbf{y})\mathbf{y}^{\mathsf{T}})W$$

See: `http://www.cnl.salk.edu/~tony/ica.html` (a bit out-of-date).

Transforming densities

Consider $\mathbf{y} \in Y$ and $\mathbf{x} \in X$ with $F : Y \overset{1-1}{\rightarrow} X$. If $\mathbf{y} \sim p_y$, what is the density $p_x(\mathbf{x})$?

$$\begin{aligned} \int_{\Omega} d\mathbf{x}\; p_x(\mathbf{x}) &= \int_{F^{-1}(\Omega)} d\mathbf{y}\; p_y(\mathbf{y}) && \left[\mathbf{x} \in \Omega \Leftrightarrow \mathbf{y} \in F^{-1}(\Omega)\right] \\ &= \int_{F(F^{-1}(\Omega))} d\mathbf{x}\; p_y(F^{-1}(\mathbf{x})) \left|\frac{d\mathbf{y}}{d\mathbf{x}}\right| && \left[\text{change of variables } \mathbf{y} \to \mathbf{x}\right] \end{aligned}$$

where $\left|\frac{d\mathbf{y}}{d\mathbf{x}}\right|$ is the Jacobian (determinant of matrix of partial derivatives).

Since this is true for any region Ω we must have

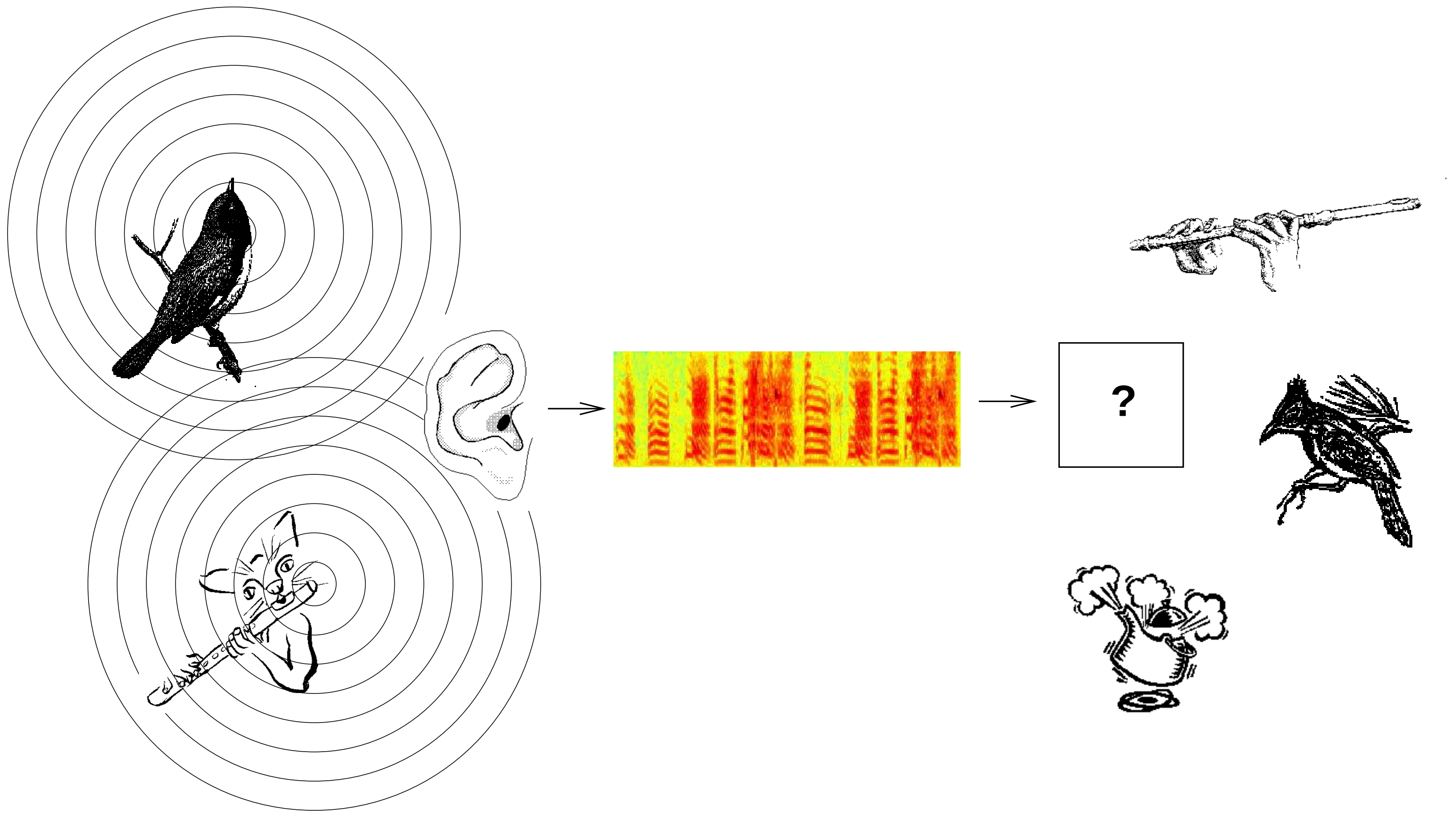
$$p_x(\mathbf{x}) = p_y(F^{-1}(\mathbf{x})) \left|\frac{d\mathbf{y}}{d\mathbf{x}}\right|$$

A special case: $X \subseteq \mathbb{R}$; $Y = [0, 1]$; $p_y(y) = 1$

$$\begin{aligned} p_x(x) = p_y(F^{-1}(x)) \cdot (F^{-1})'(x) = 1 \cdot (F^{-1})'(x) \\ \Rightarrow F^{-1} = \int_{-\infty}^{x} dx'\; p_x(x') = \text{cdf}(x) \end{aligned}$$

(Boundary conditions set by range of F^{-1}.)

ICA – Blind Source Separation



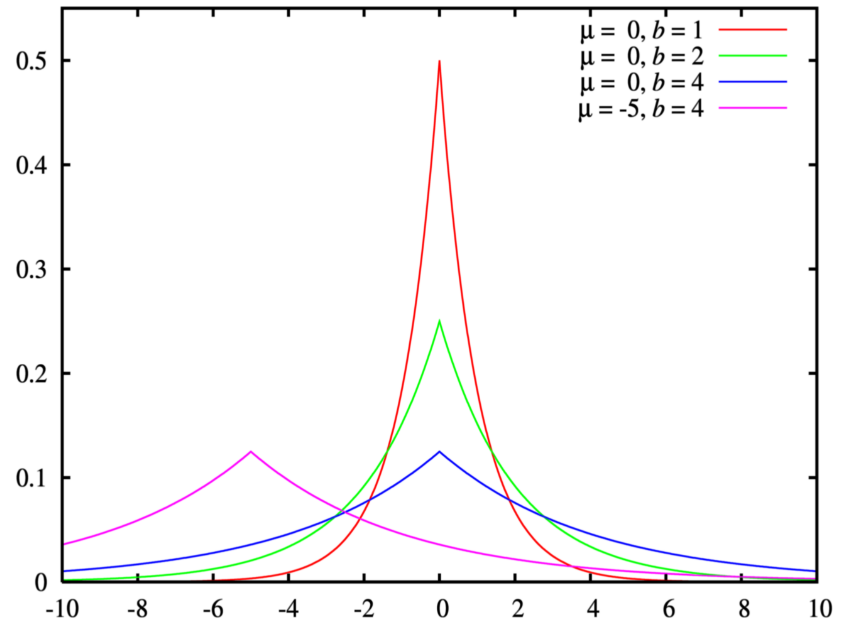
- Often used interchangeably with ICA, but the BSS problem involves time-series.
- Even if temporal dependence is not modelling, independence of non-simultaneous measurements on different channels can be exploited.
- Many algorithms: DCA, SOBI, JADE, . . .

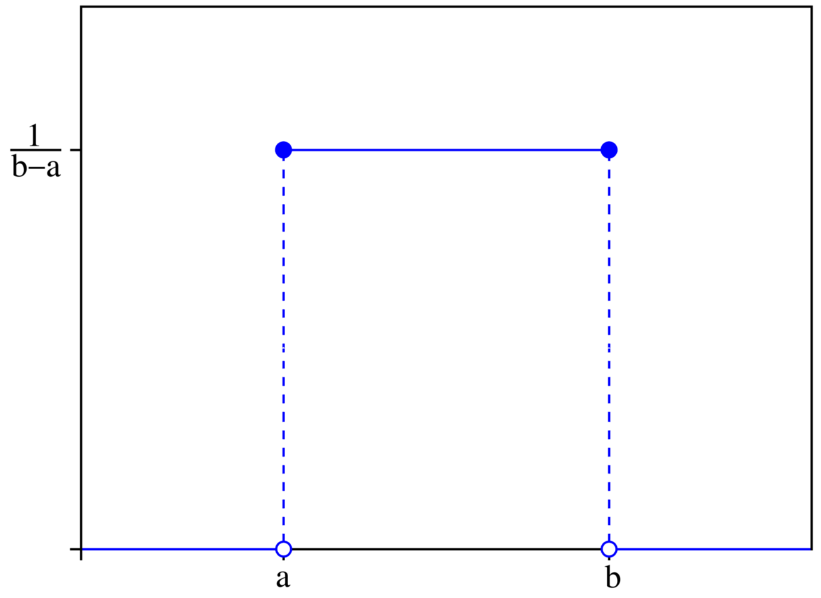
Kurtosis

The kurtosis (or excess kurtosis) measures how “peaky” or “heavy-tailed” a distribution is.

$$K = \frac{E((x-\mu)^4)}{E((x-\mu)^2)^2} - 3$$

where $\mu = E(x)$ is the mean of x.
Gaussian distributions have zero kurtosis.



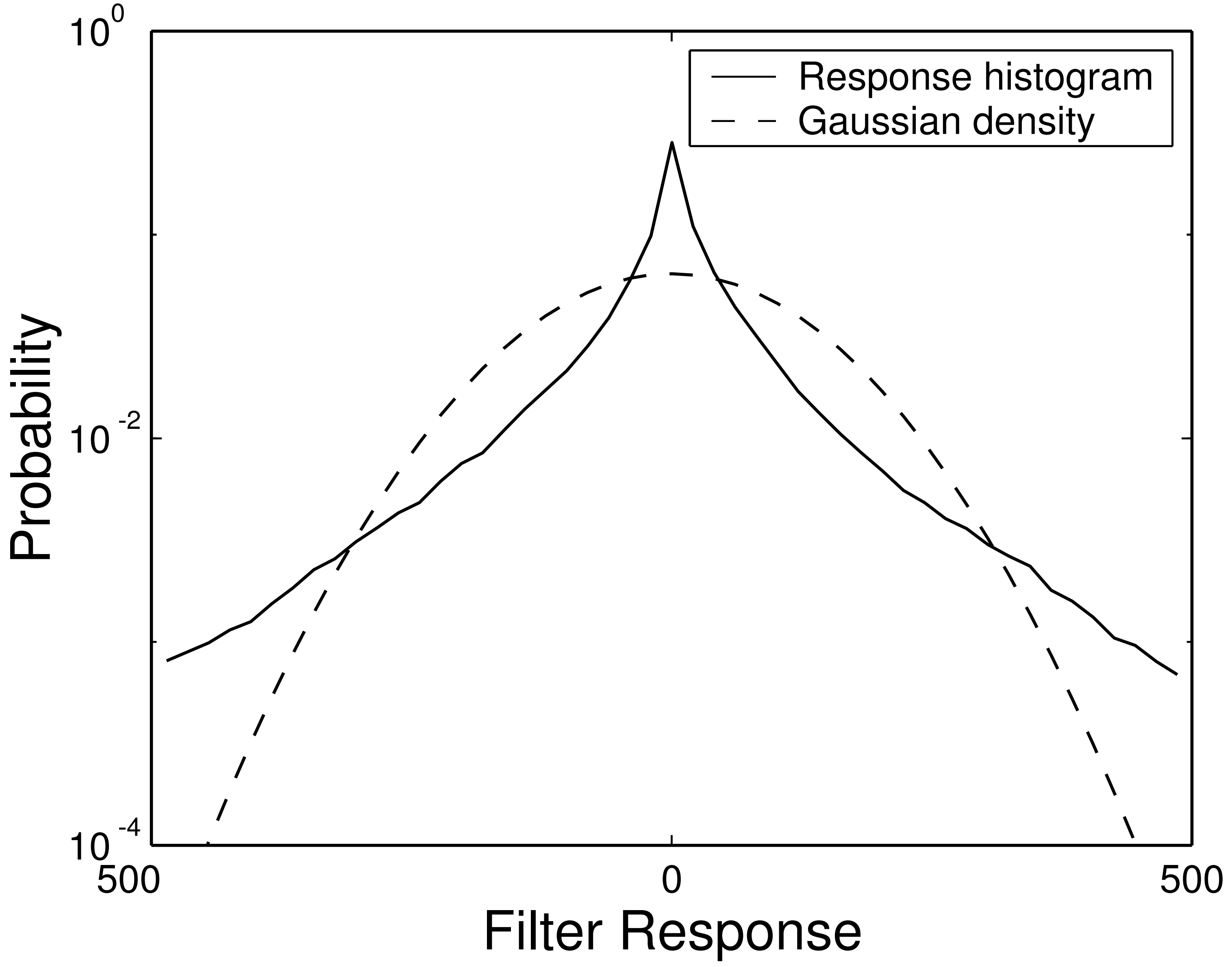


Heavy tailed distributions have positive kurtosis (leptokurtic).

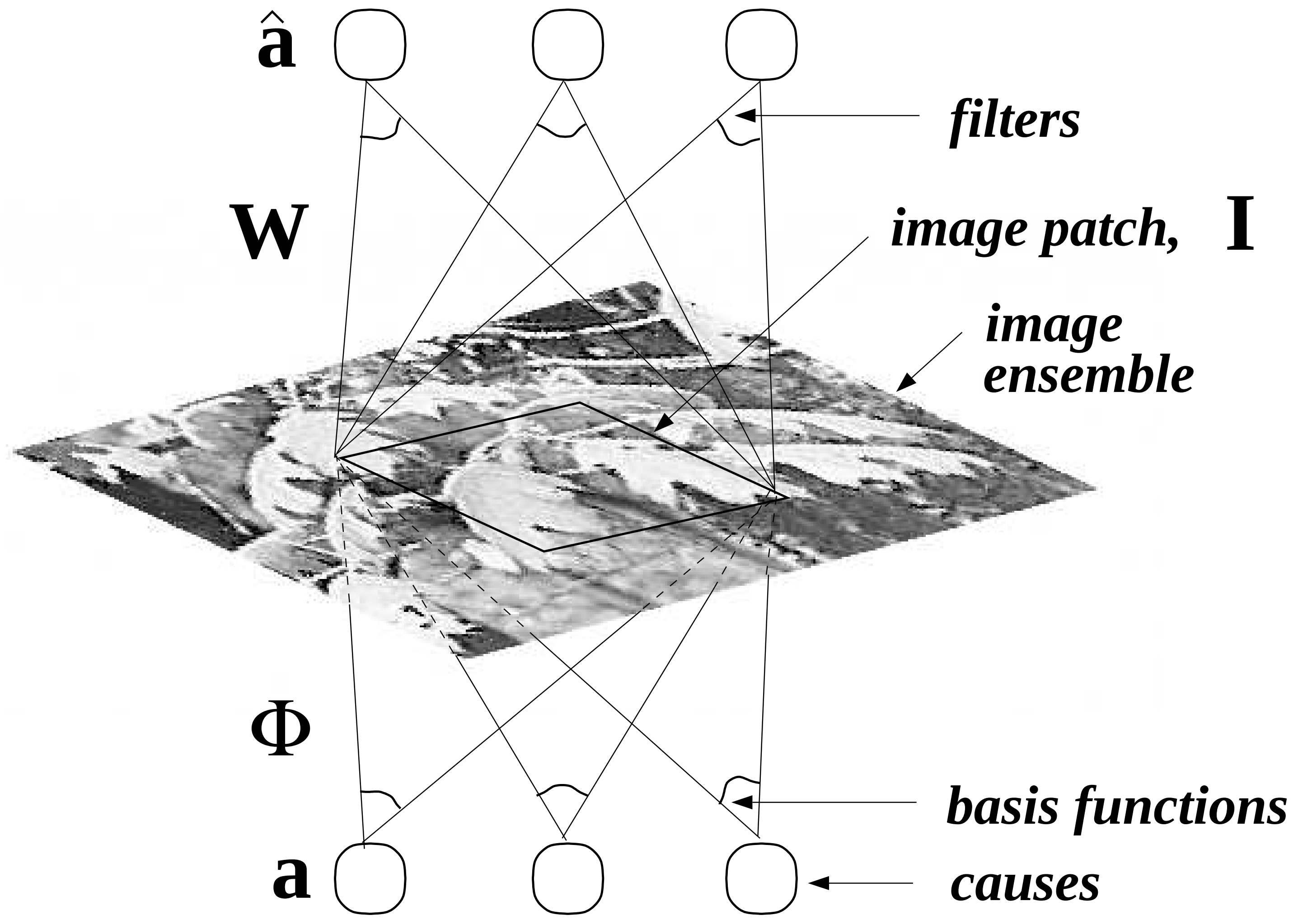
Light tailed distributions have negative kurtosis (platykurtic).

Some ICA algorithms are essentially kurtosis pursuit approaches. Possibly fewer assumptions about generating distributions.

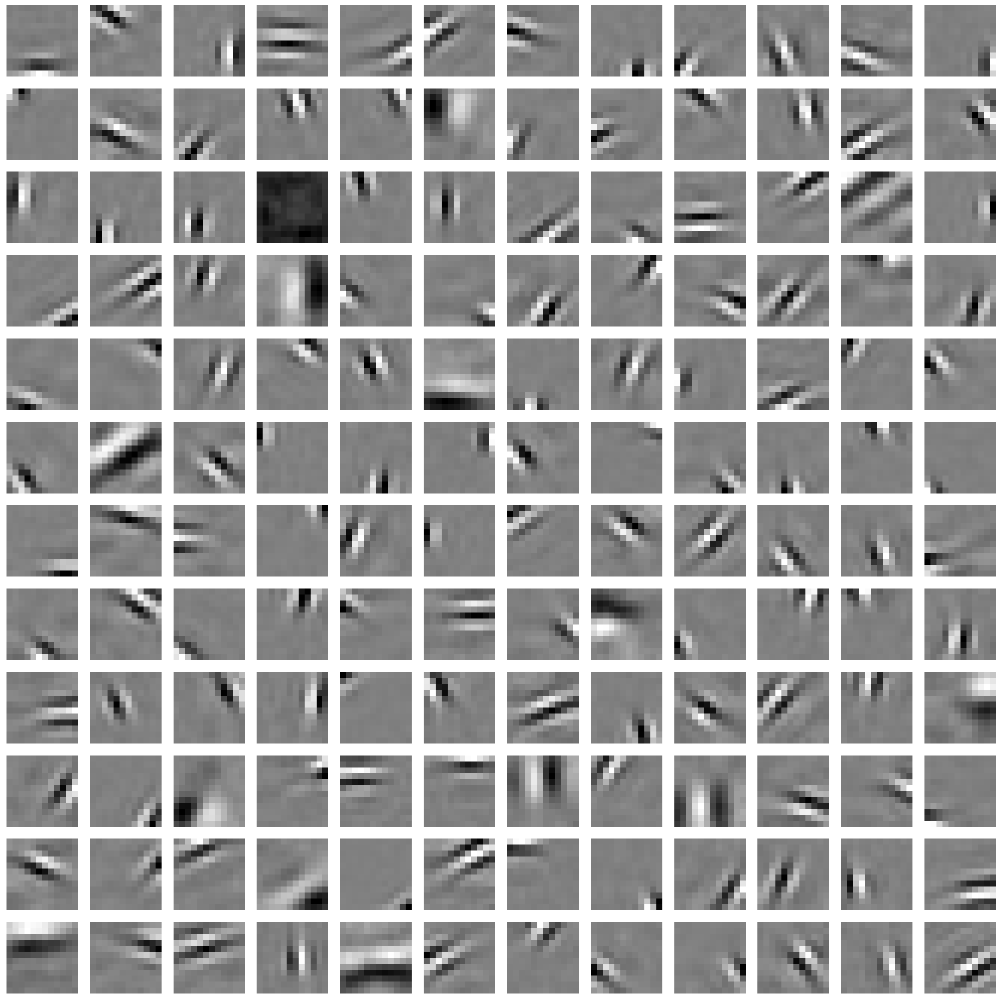
Natural Scenes and Sounds



Images


$\hat{\mathbf{a}}$
filters
$\mathbf{W}$
image patch, $\mathbf{I}$
image ensemble
Φ
basis functions
$\mathbf{a}$
causes

Natural Scenes



Olshausen & Field (1996). Emergence of Simple-Cell Receptive Field Properties by Learning a Sparse Code for Natural Images. *Nature* **381**:607-609.

ICA and BSS

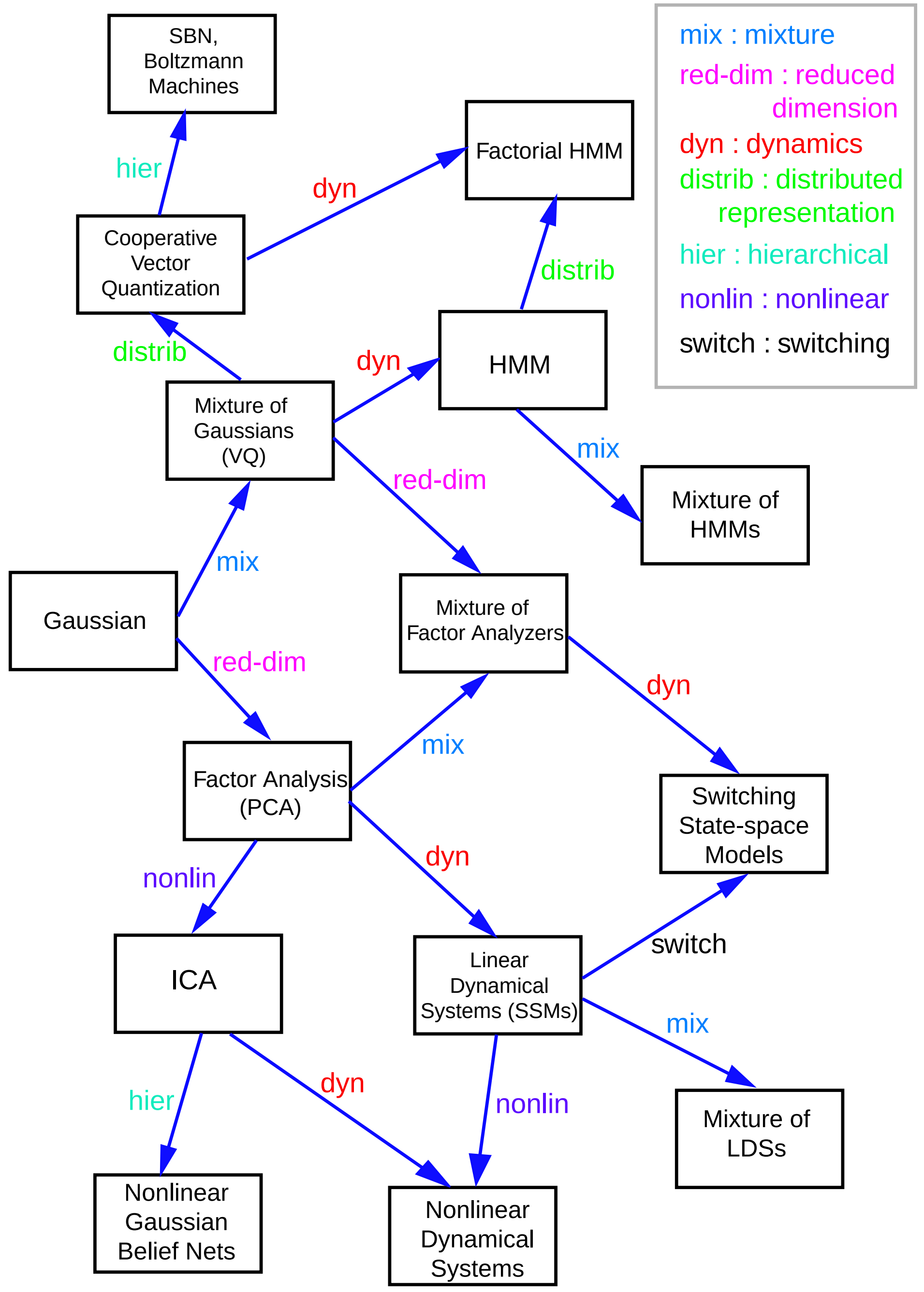
Applications:

- Separating auditory sources
- Analysis of EEG data
- Analysis of functional MRI data
- Natural scene analysis
- . . .

Extensions:

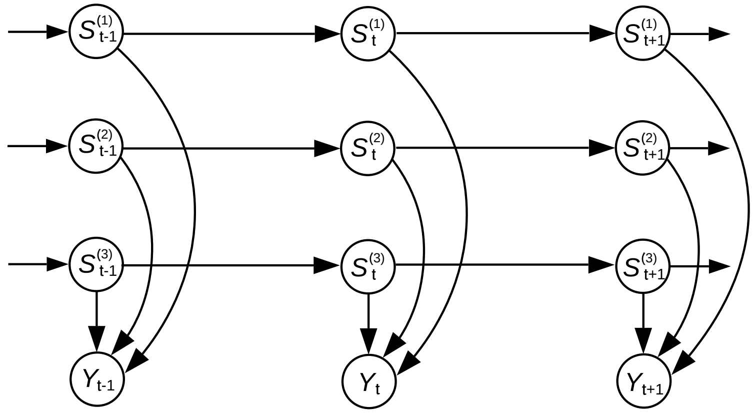
- Non-zero output noise – approximate posteriors and learning.
- Undercomplete ($K < D$) or overcomplete ($K > D$).
- Learning prior distributions (on **y**).
- Dynamical hidden models (on **y**).
- Learning number of sources.
- Time-varying mixing matrix.
- . . .

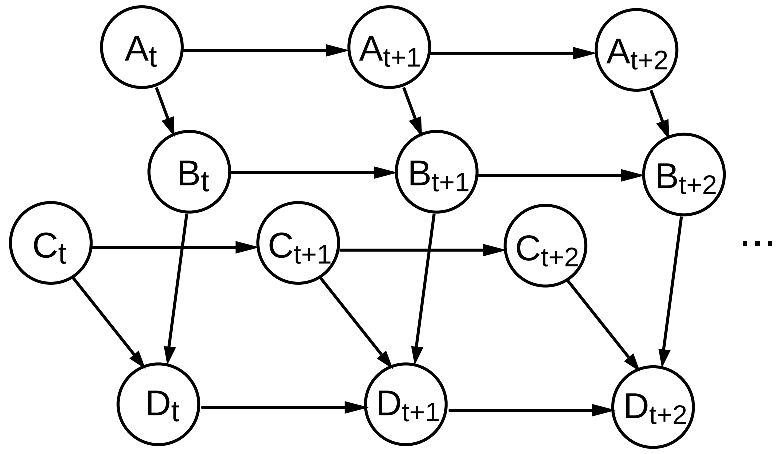
How ICA Relates to Factor Analysis and Other Models



- **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": $y_k = 1$ and $y_j = 0$ 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.

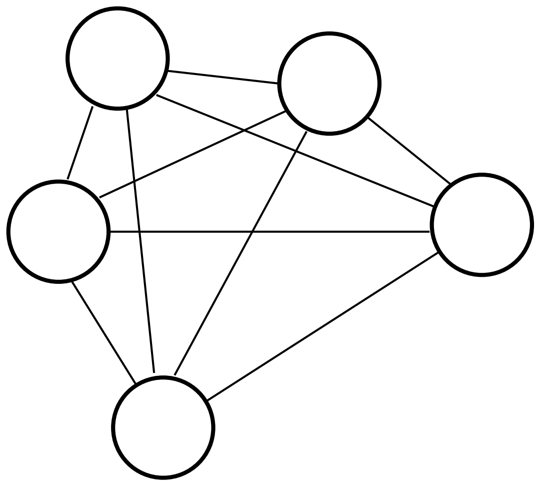
Distributed representations: FHMMs and DBNs





- These are hidden Markov models with many state variables (i.e. a distributed representation of the state).
- The state can capture many more bits of information about the sequence (linear in the number of state variables).
- E step is usally intractable (due to coupling of parents).

Distributed representations: Boltzmann Machines



Undirected graphical model (i.e. a Markov network) over a vector of binary variables $s_i \in \{0, 1\}$. Some variables may be hidden, some may be visible (observed).

$$P(\mathbf{s}|W, \mathbf{b}) = \frac{1}{Z} \exp \left\{ \sum_{ij} W_{ij} s_i s_j - \sum_i b_i s_i \right\}$$

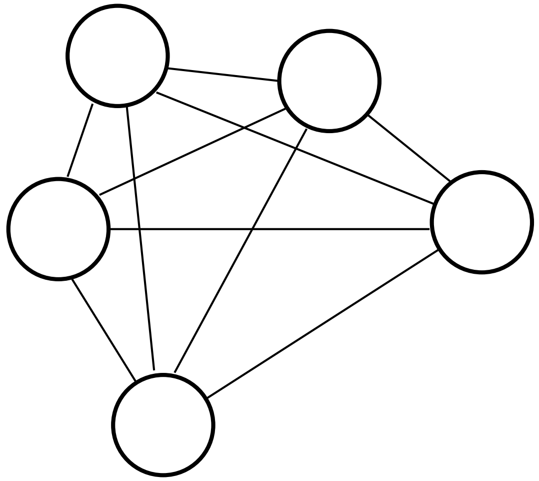
where Z is the normalization constant (partition function).

Learning algorithm: a gradient version of EM

- E step involves computing averages w.r.t. $P(\mathbf{s}^H|\mathbf{s}^V, W, \mathbf{b})$ ("clamped phase"). This could be done via a propagation algorithm or (more usually) an approximate method such as Gibbs sampling.
- The M step requires gradients w.r.t. Z, which can be computed by averages w.r.t. $P(\mathbf{s}|W, \mathbf{b})$ ("unclamped phase").

$$\Delta W_{ij} = \eta[\langle s_i s_j \rangle_c - \langle s_i s_j \rangle_u]$$

Learning in Boltzmann Machines



$$\log P(\mathbf{s}^V \mathbf{s}^H | W, \mathbf{b}) = \sum_{ij} W_{ij} s_i s_j - \sum_i b_i s_i - \log Z$$

with $Z = \sum_{\mathbf{s}} e^{\sum_{ij} W_{ij} s_i s_j - \sum_i b_i s_i}$

Generalised (gradient M-step) EM requires parameter step

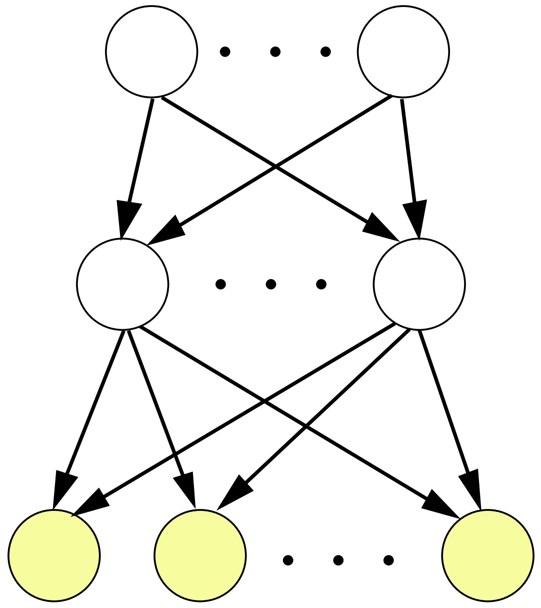
$$\Delta W_{ij} \propto \frac{\partial}{\partial W_{ij}} \Big\langle \log P(\mathbf{s}^V \mathbf{s}^H | W, \mathbf{b}) \Big\rangle_{P(\mathbf{s}^H | \mathbf{s}^V)}$$

Write $\langle\rangle_c$ (clamped) for expectations under $P(\mathbf{s}|\mathbf{s}^V)$ (with delta function $P(\mathbf{s}^V|\mathbf{s}^V)$). Then

$$\begin{aligned}
\Delta W_{ij} &\propto \frac{\partial}{\partial W_{ij}} \Big[\textstyle\sum_{ij} W_{ij} \langle s_i s_j \rangle_c - \sum_i b_i \langle s_i \rangle_c - \log Z \Big] \\
&= \langle s_i s_j \rangle_c - \frac{\partial}{\partial W_{ij}} \log Z \\
&= \langle s_i s_j \rangle_c - \frac{1}{Z} \frac{\partial}{\partial W_{ij}} \sum_{\mathbf{s}} e^{\sum_{ij} W_{ij} s_i s_j - \sum_i b_i s_i} \\
&= \langle s_i s_j \rangle_c - \sum_{\mathbf{s}} \frac{1}{Z} e^{\sum_{ij} W_{ij} s_i s_j - \sum_i b_i s_i} s_i s_j \\
&= \langle s_i s_j \rangle_c - \sum_{\mathbf{s}} P(\mathbf{s}|W, \mathbf{b}) s_i s_j \quad = \langle s_i s_j \rangle_c - \langle s_i s_j \rangle_u
\end{aligned}$$

with $\langle\rangle_u$ (unclamped) an expectation under the current joint distribution.

Sigmoid Belief Networks



Directed graphical model (i.e. a Bayesian network) over a vector of binary variables $s_i \in \{0, 1\}$.

$$P(\mathbf{s}|W, \mathbf{b}) = \prod_i P(s_i|\{s_j\}_{j<i}, W, \mathbf{b})$$

$$P(s_i = 1|\{s_j\}_{j<i}, W, \mathbf{b}) = \frac{1}{1 + \exp\{-\sum_{j<i} W_{ij} s_j - b_i\}}$$

A probabilistic version of sigmoid multilayer perceptrons (“neural networks”).

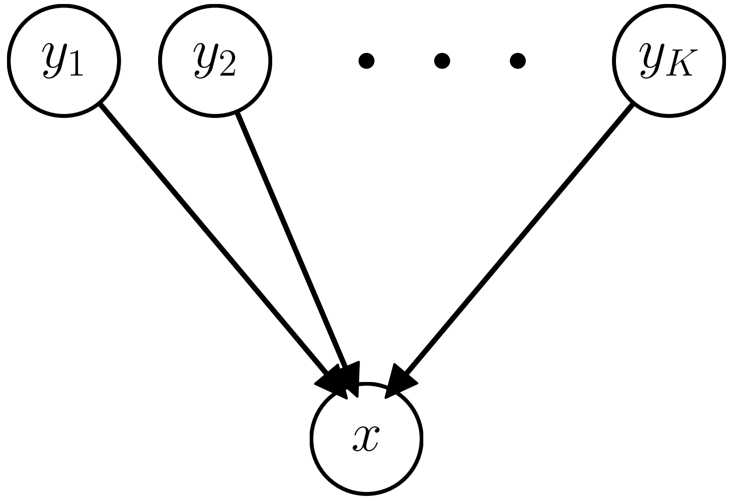
Learning algorithm: a gradient version of EM

- E step involves computing averages w.r.t. $P(\mathbf{s}_H|\mathbf{s}_V, W, \mathrm{b})$. This could be done via the Belief Propagation algorithm (if singly connected) or (more usually) an approximate method such as Gibbs sampling or mean field (see later lectures).
- Unlike Boltzmann machines, there is no partition function, so no need for an unclamped phase in the M step.

Intractability

For many probabilistic models of interest, exact inference is not computationally feasible. This occurs for two (main) reasons:

- distributions may have complicated forms (non-linearities in generative model)
- "explaining away" causes coupling from observations
  observing the value of a child induces dependencies amongst its parents (high order interactions)

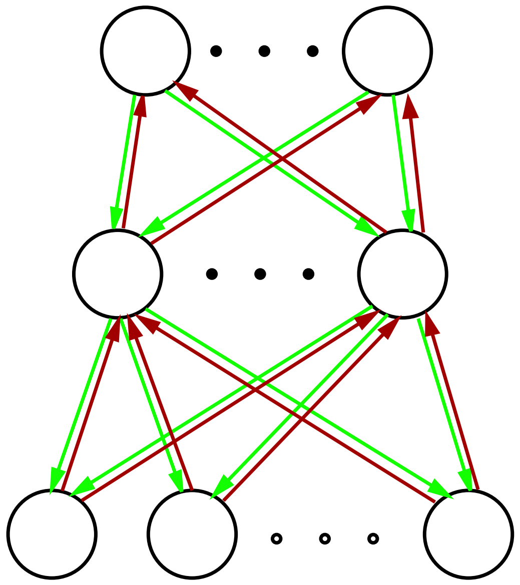


We can still work with such models by using *approximate inference* techniques to estimate the latent variables.

Approximate Inference

- **Sampling**: Approximate posterior distribution over hidden variables by a set of random samples. We often need **Markov chain Monte carlo** methods to sample from difficult distributions.
- **Linearisation**: Approximate nonlinearities by Taylor series expansion about a point (e.g. the approximate mean of the hidden variable distribution). Linear approximations are particularly useful since Gaussian distributions are closed under linear transformations e.g., EKF.
- **Recognition Models**: Approximate the hidden variable posterior distribution using an explicit *bottom-up* recognition model/network.
- **Variational Methods**: Approximate the hidden variable posterior $p(H)$ with a tractable form $q(H)$, such that **KL**$[q\|p]$ is minimised. This gives a lower bound on the likelihood that can be maximised with respect to the parameters of $q(H)$.
- **Other Deterministic Methods**: Approximate the hidden variable posterior $p(H)$ with a tractable form $q(H)$ by other means (moment-matching, Laplace $\ldots$). Still gives a lower bound (by Jensen), but we may not be increasing this lower bound.

Recognition Models



- a model is trained in a supervised way to recover the hidden causes (latent variables) from the observations
- this may take the form of explicit recognition network (e.g. Helmholtz machine) which mirrors the generative network (tractability at the cost of restricted approximating distribution)
- inference is done in a single *bottom-up* pass (no iteration required)