CIAR Second Summer School Tutorial
Lecture 1b

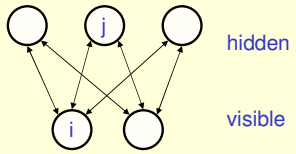
# Contrastive Divergence and Deterministic Energy-Based Models

Geoffrey Hinton

## Restricted Boltzmann Machines

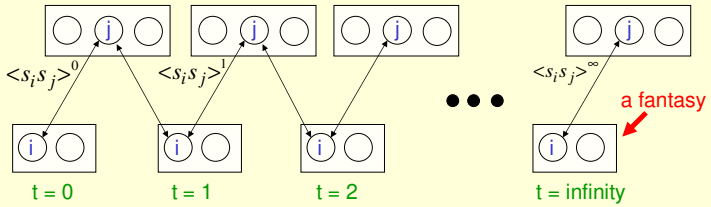
- We restrict the connectivity to make inference and learning easier.
  - Only one layer of hidden units.
  - No connections between hidden units.
- In an RBM it only takes one step to reach thermal equilibrium when the visible units are clamped.
  - So we can quickly get the exact value of :

$$< s_i s_j >_{\mathbf{v}}$$




$$p(s_j = 1) = \frac{1}{1 + e^{-\left(b_j + \sum_{i \in vis} s_i w_{ij}\right)}}$$

## A picture of the Boltzmann machine learning algorithm for an RBM

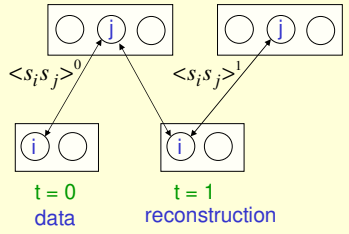



Start with a training vector on the visible units.

Then alternate between updating all the hidden units in parallel and updating all the visible units in parallel.

$$\Delta w_{ij} = \varepsilon \left( <s_i s_j>^0 - <s_i s_j>^\infty \right)$$

## The short-cut




Start with a training vector on the visible units.

Update all the hidden units in parallel

Update the all the visible units in parallel to get a "reconstruction".

Update the hidden units again.

$$\Delta w_{ij} = \varepsilon \left( <s_i s_j>^0 - <s_i s_j>^1 \right)$$

This is not following the gradient of the log likelihood. But it works very well.

## Contrastive divergence

Aim is to minimize the amount by which a step toward equilibrium improves the data distribution.

data distribution

model's distribution

distribution after one step of Markov chain

$$CD = KL(P \| Q^{\infty}) - KL(Q^{1} \| Q^{\infty})$$

Minimize Contrastive Divergence

Minimize divergence between data distribution and model's distribution

Maximize the divergence between confabulations and model's distribution

## Contrastive divergence

$$-\frac{\partial KL(Q^0 \| Q^{\infty})}{\partial \theta} = -\left\langle \frac{\partial E}{\partial \theta} \right\rangle_{Q^0} + \left\langle \frac{\partial E}{\partial \theta} \right\rangle_{Q^{\infty}}$$

$$-\frac{\partial KL(Q^1 \| Q^{\infty})}{\partial \theta} = -\left\langle \frac{\partial E}{\partial \theta} \right\rangle_{Q^1} + \left\langle \frac{\partial E}{\partial \theta} \right\rangle_{Q^{\infty}} - \frac{\partial Q^1}{\partial \theta} \frac{\partial KL(Q^1 \| Q^{\infty})}{\partial Q^1}$$

Contrastive divergence makes the awkward terms cancel

changing the parameters changes the distribution of confabulations

## How to learn a set of features that are good for reconstructing images of the digit 2

50 binary feature neurons

50 binary feature neurons

Increment weights between an active pixel and an active feature

Decrement weights between an active pixel and an active feature

16 x 16 pixel image

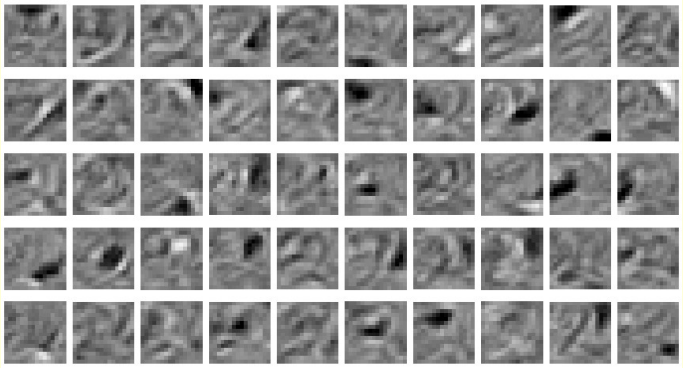
data (reality)

16 x 16 pixel image

reconstruction (lower energy than reality)

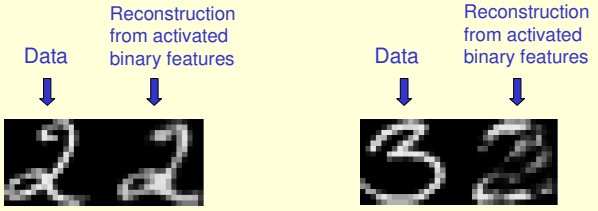
Bartlett

## The final 50 x 256 weights



Each neuron grabs a different feature.

## How well can we reconstruct the digit images from the binary feature activations?




New test images from the digit class that the model was trained on

Images from an unfamiliar digit class (the network tries to see every image as a 2)

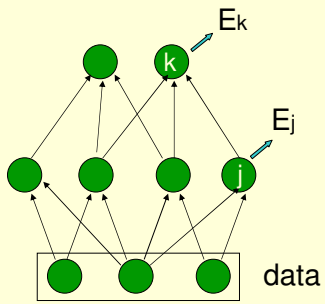
## Another use of contrastive divergence

- CD is an efficient way to learn Restricted Boltzmann Machines.
- But it can also be used for learning other types of energy-based model that have multiple hidden layers.
- Methods very similar to CD have been used for learning non-probabilistic energy-based models (LeCun, Hertzmann).

## Energy-Based Models with deterministic hidden units

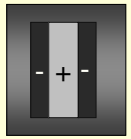
- Use multiple layers of deterministic hidden units with non-linear activation functions.
- Hidden activities contribute additively to the global energy, E.
- Familiar features help, violated constraints hurt.

$$p(d) = \frac{e^{-E(d)}}{\sum_c e^{-E(c)}}$$

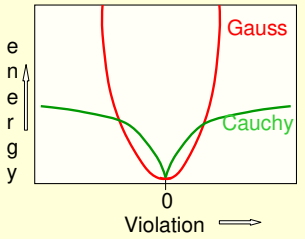



## Frequently Approximately Satisfied constraints

- The intensities in a typical image satisfy many different linear constraints very accurately, and violate a few constraints by a lot.
- The constraint violations fit a heavy-tailed distribution.
- The negative log probabilities of constraint violations can be used as energies.




On a smooth intensity patch the sides balance the middle



## Reminder:
## Maximum likelihood learning is hard

- To get high log probability for d we need low energy for d and high energy for its main rivals, c

$$\log p(d) = -E(d) - \log \sum_c e^{-E(c)}$$

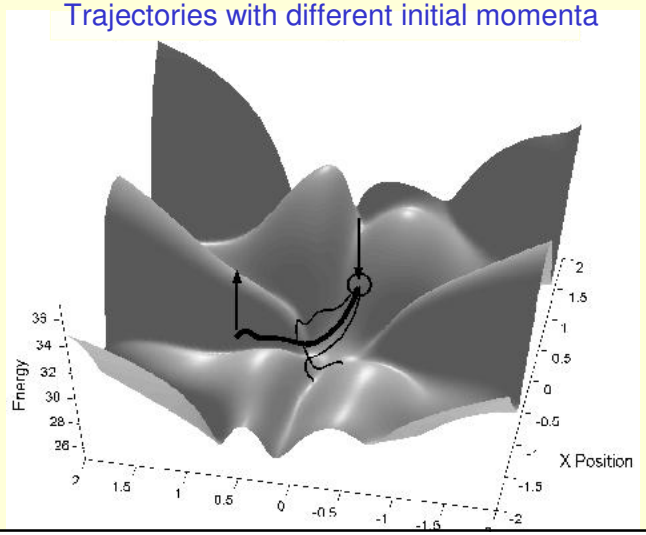
$$\frac{\partial \log p(d)}{\partial \theta} = -\frac{\partial E(d)}{\partial \theta} + \sum_c p(c) \frac{\partial E(c)}{\partial \theta}$$

To sample from the model use Markov Chain Monte Carlo. But what kind of chain can we use when the hidden units are deterministic and the visible units are real-valued.

## Hybrid Monte Carlo

- We could find good rivals by repeatedly making a random perturbation to the data and accepting the perturbation with a probability that depends on the energy change.
  - Diffuses very slowly over flat regions
  - Cannot cross energy barriers easily
- In high-dimensional spaces, it is much better to use the gradient to choose good directions.
- HMC adds a random momentum and then simulates a particle moving on an energy surface.
  - Beats diffusion. Scales well.
  - Can cross energy barriers.
  - Back-propagation can give us the gradient of the energy surface.

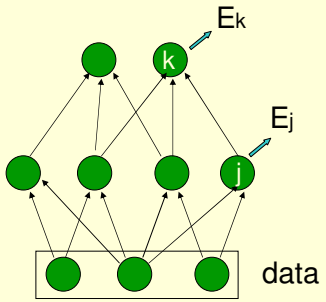
## Trajectories with different initial momenta




## Backpropagation can compute the gradient that Hybrid Monte Carlo needs

1. Do a forward pass computing hidden activities.
2. Do a backward pass all the way to the data to compute the derivative of the global energy w.r.t each component of the data vector.

works with any smooth non-linearity



## The online HMC learning procedure

1. Start at a datavector, d, and use backprop to compute for every parameter $\partial E(d)/\partial\theta$
2. Run HMC for many steps with frequent renewal of the momentum to get equilibrium sample, c. Each step involves a forward and backward pass to get the gradient of the energy in dataspace.
3. Use backprop to compute $\partial E(c)/\partial\theta$
4. Update the parameters by :

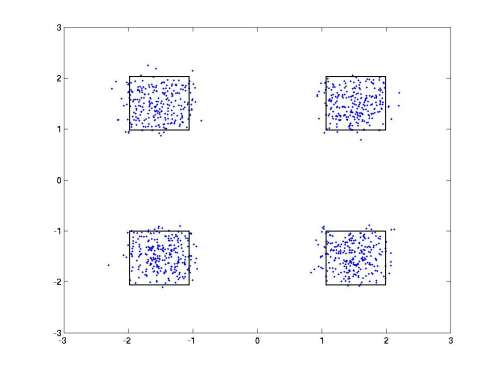
$$\Delta\theta = \varepsilon\left(-\partial E(d)/\partial\theta + \partial E(c)/\partial\theta\right)$$

## The shortcut

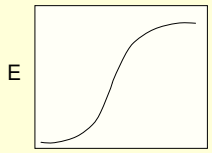
- Instead of taking the negative samples from the equilibrium distribution, use slight corruptions of the datavectors. Only add random momentum once, and only follow the dynamics for a few steps.
  - Much less variance because a datavector and its confabulation form a matched pair.
  - Gives a very biased estimate of the gradient of the log likelihood.
  - Gives a good estimate of the gradient of the contrastive divergence (i.e. the amount by which F falls during the brief HMC.)
- Its very hard to say anything about what this method does to the log likelihood because it only looks at rivals in the vicinity of the data.
- Its hard to say exactly what this method does to the contrastive divergence because the Markov chain defines what we mean by "vicinity", and the chain keeps changing as the parameters change.
  - But its works well empirically, and it can be proved to work well in some very simple cases.

## A simple 2-D dataset

The true data is uniformly distributed within the 4 squares. The blue dots are samples from the model.




## The network for the 4 squares task

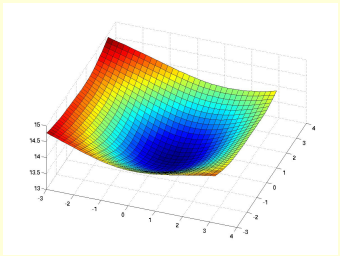


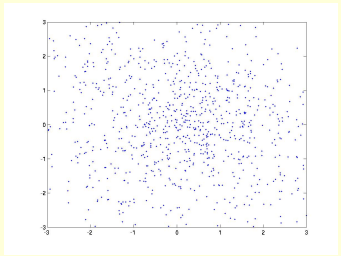

3 logistic units

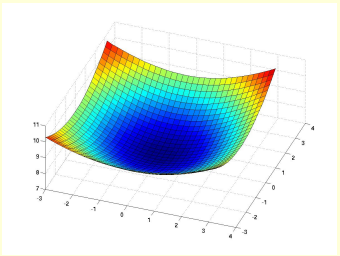
Each hidden unit contributes an energy equal to its activity times a learned scale.

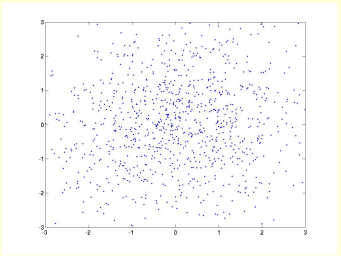
20 logistic units

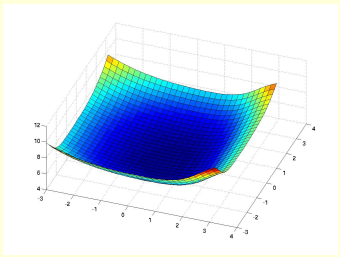
2 input units

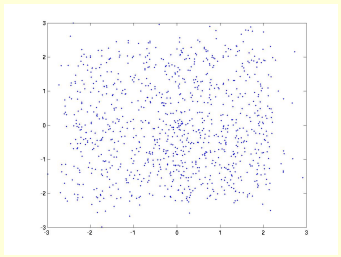


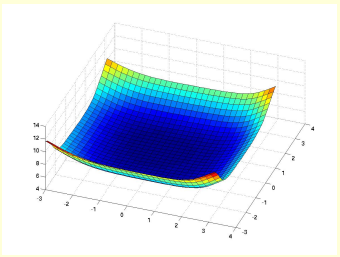


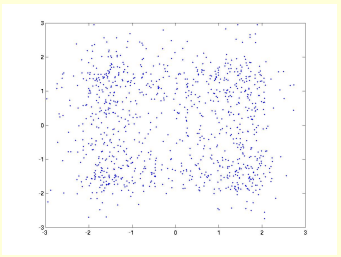


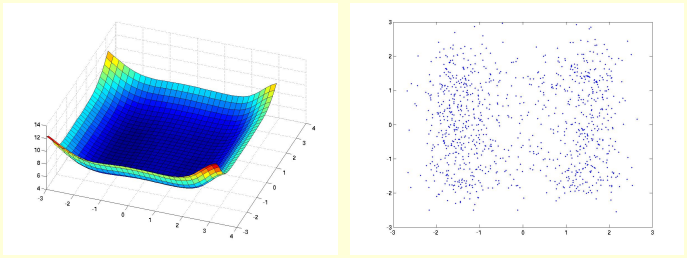


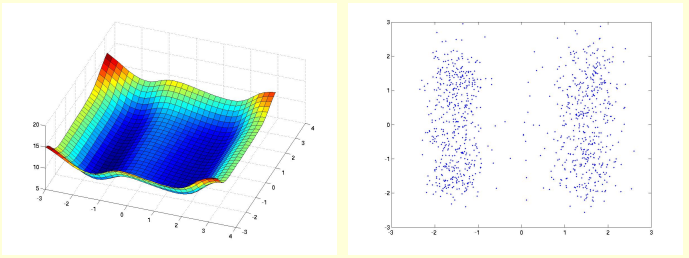


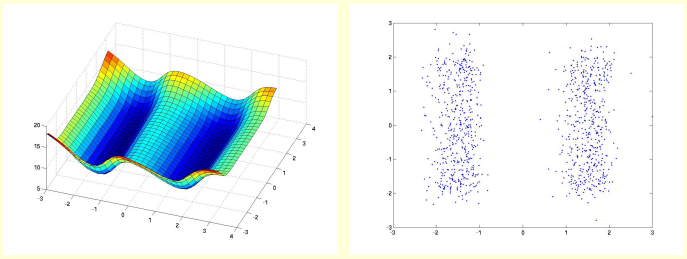


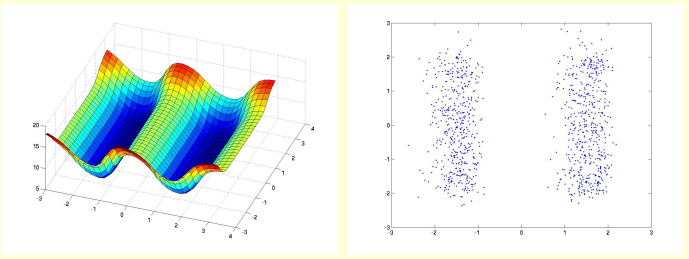


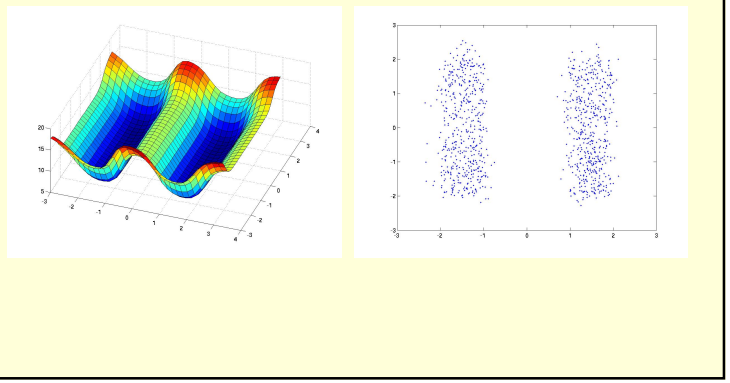


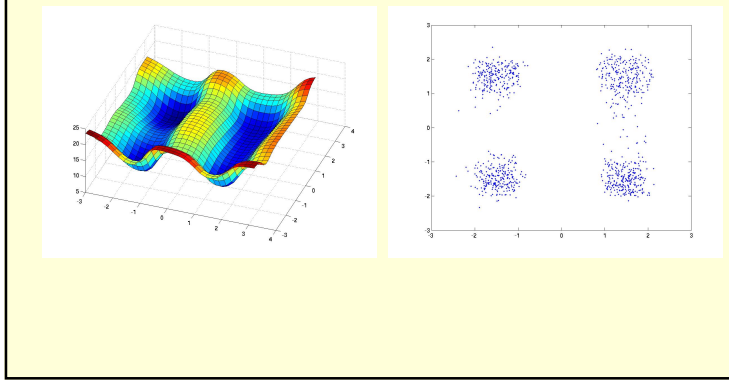


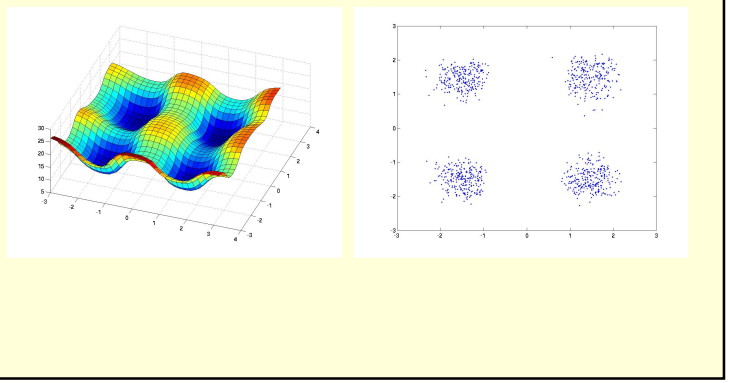


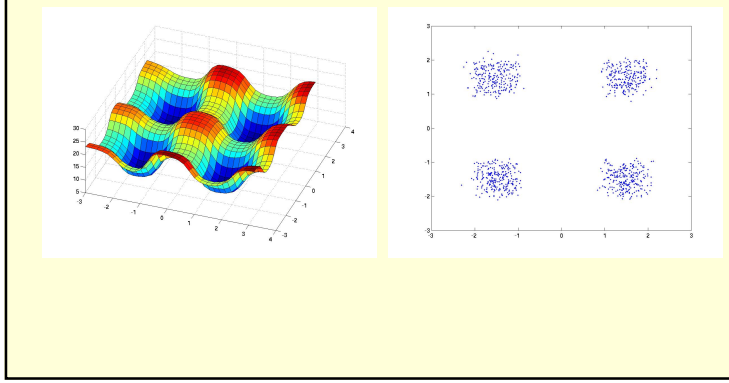






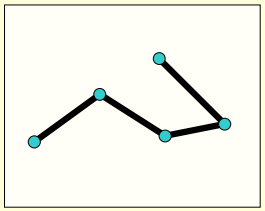


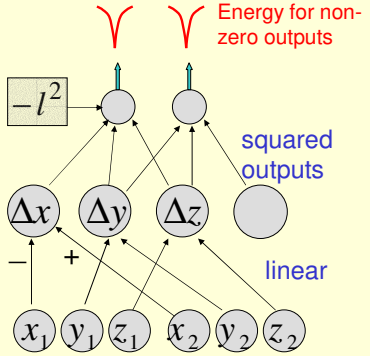




## Learning the constraints on an arm

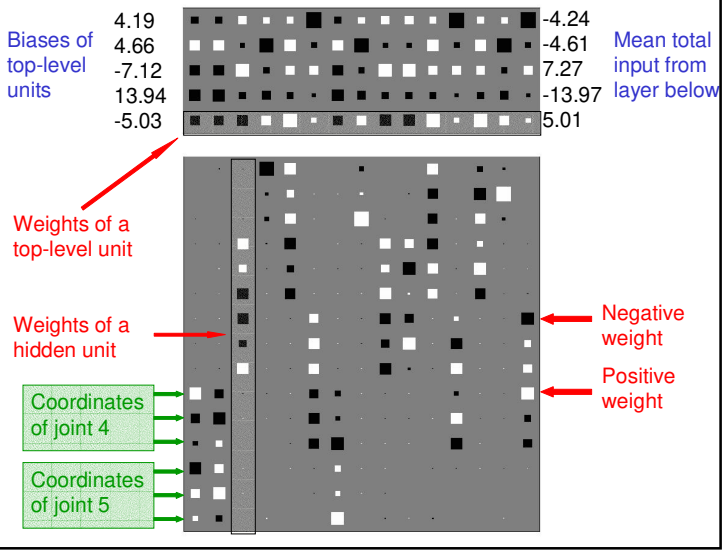
3-D arm with 4 links and 5 joints






For each link:

$$\Delta x^2 + \Delta y^2 + \Delta z^2 - l^2 = 0$$




## Superimposing constraints

- A unit in the second layer could represent a single constraint.
- But it can model the data just as well by representing a linear combination of constraints.

$$a\Delta x_{34}^2 + a\Delta y_{34}^2 + a\Delta z_{34}^2 - al_{34}^2 = 0$$

$$b\Delta x_{45}^2 + b\Delta y_{45}^2 + b\Delta z_{45}^2 - bl_{45}^2 = 0$$

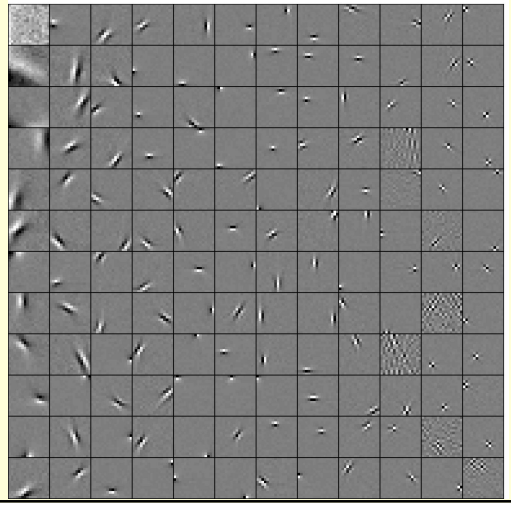
## Dealing with missing inputs

- The network learns the constraints even if 10% of the inputs are missing.
  - First fill in the missing inputs randomly
  - Then use the back-propagated energy derivatives to slowly change the filled-in values until they fit in with the learned constraints.
- Why don't the corrupted inputs interfere with the learning of the constraints?
  - The energy function has a small slope when the constraint is violated by a lot.
  - So when a constraint is violated by a lot it does not adapt.
    - Don't learn when things don't make sense.

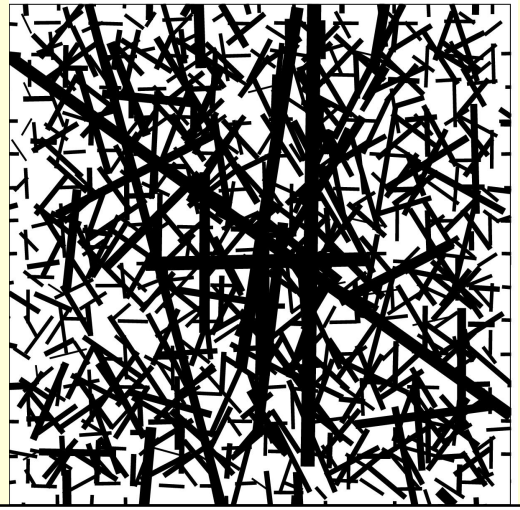
## Learning constraints from natural images (Yee-Whye Teh)

- We used 16x16 image patches and a single layer of 768 hidden units (3 x over-complete).
- Confabulations are produced from data by adding random momentum once and simulating dynamics for 30 steps.
- Weights are updated every 100 examples.
- A small amount of weight decay helps.

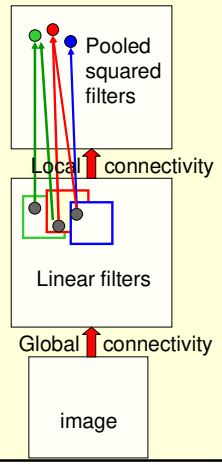
## A random subset of 768 basis functions



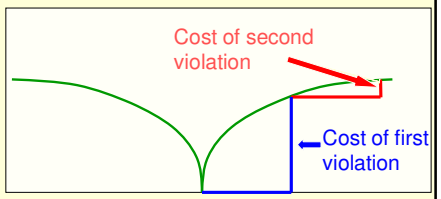
## The distribution of all 768 learned basis functions

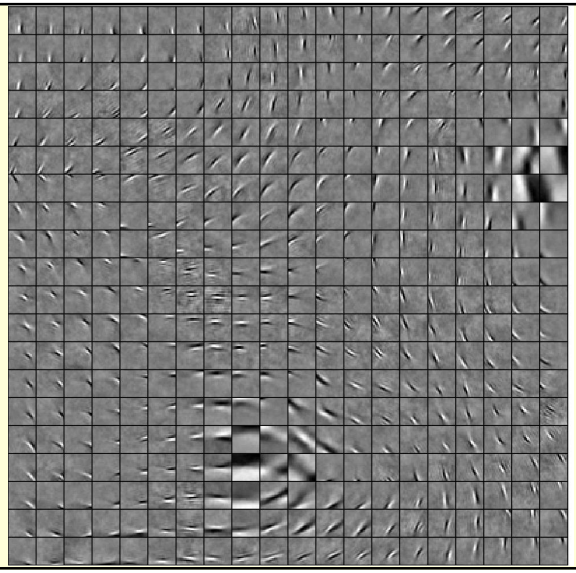


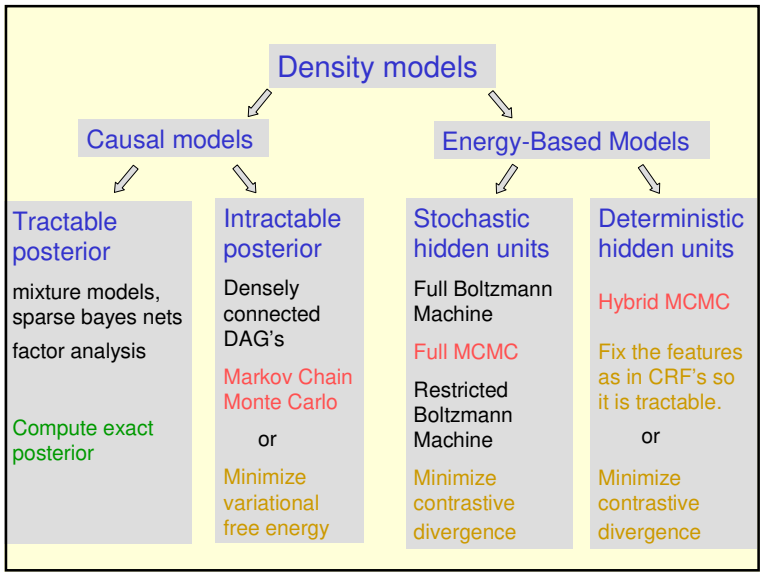
## How to learn a topographic map




The outputs of the linear filters are squared and locally pooled. This makes it cheaper to put filters that are violated at the same time next to each other.








# THE END

## Independence relationships of hidden variables in three types of model that have one hidden layer

| | Causal model | Product of experts (RBM) | Square ICA |
|---|---|---|---|
| Hidden states unconditional on data | independent (generation is easy) | dependent (rejecting away) | independent (by definition) |
| Hidden states conditional on data | dependent (explaining away) | independent (inference is easy) | independent (the posterior collapses to a single point) |

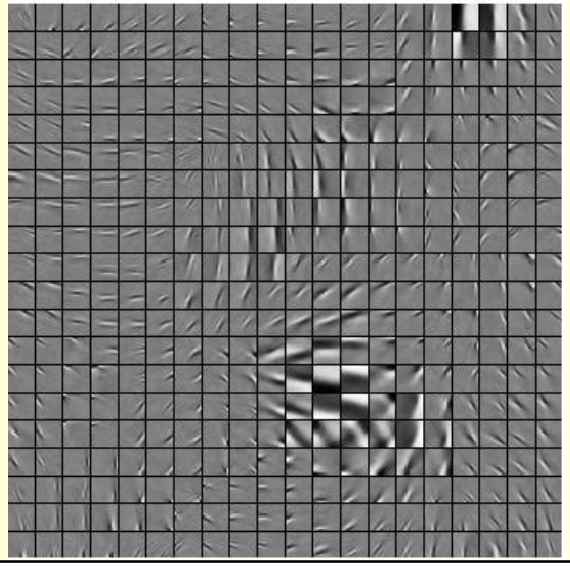
We can use an almost complementary prior to reduce this dependency so that variational inference works

## Faster mixing chains

- Hybrid Monte Carlo can only take small steps because the energy surface is curved.
- With a single layer of hidden units, it is possible to use alternating parallel Gibbs sampling.
  - Step 1: each student-t hidden unit picks a variance from the posterior distribution over variances given the violation produced by the current datavector. If the violation is big, it picks a big variance
    - This is equivalent to picking a Gaussian from an infinite mixture of Gaussians (because that's what a student-t is).
  - With the variances fixed, each hidden unit defines a one-dimensional Gaussians in the dataspace.
  - Step 2: pick a visible vector from the product of all the one-dimensional Gaussians.

## Pro's and Con's of Gibbs sampling

- Advantages of Gibbs sampling
  - Much faster mixing
  - Can be extended to use pooled second layer (Max Welling)
- Disadvantages of Gibbs sampling
  - Can only be used in deep networks by learning hidden layers (or pairs of layers) greedily.
  - But maybe this is OK. Its scales better than contrastive backpropagation.



## Over-complete ICA using a causal model

- What if we have more independent sources than data components? (independent \= orthogonal)
  - The data no longer specifies a unique vector of source activities. It specifies a distribution.
    - This also happens if we have sensor noise in square case.
  - The posterior over sources is non-Gaussian because the prior is non-Gaussian.
- So we need to approximate the posterior:
  - MCMC samples
  - MAP (plus Gaussian around MAP?)
  - Variational

## Over-complete ICA using an energy-based model

- Causal over-complete models preserve the unconditional independence of the sources and abandon the conditional independence.
- Energy-based overcomplete models preserve the conditional independence (which makes perception fast) and abandon the unconditional independence.
  - Over-complete EBM's are easy if we use contrastive divergence to deal with the intractable partition function.