

**10707**

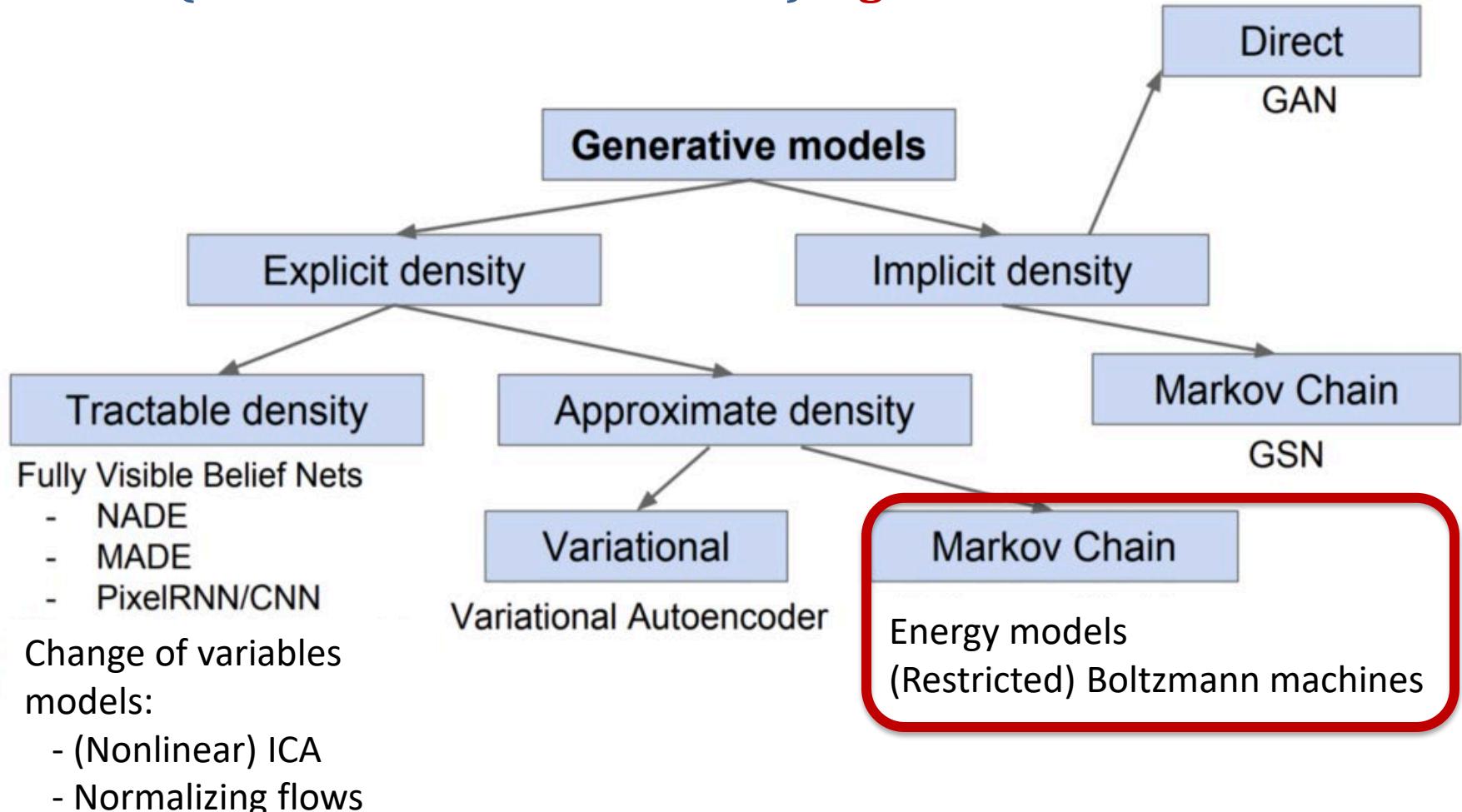
# **Deep Learning: Spring 2021**

Andrej Risteski

Machine Learning Department

**Lecture 9+10:**  
Algorithmic techniques for  
intractable explicit density models:  
MCMC, energy models, RBMs

The distribution  $p_\theta(x)$  does not have a tractable form, but we can approximate it by running a MCMC (Markov Chain Monte Carlo) algorithm.



# Training explicit density models

**Likelihood-based:** maximize the likelihood of the data under the model (possibly with variational or MCMC approximations)

$$\max_{\theta} \sum_{\text{samples } x_i} \log p_{\theta}(x_i)$$

In the limit of infinite number of samples:

$$\begin{aligned} \max_{\theta} \mathbb{E}_{x \sim p} \log p_{\theta}(x) &= -(\mathbb{E}_{x \sim p} \log \frac{1}{p_{\theta}(x)} + \mathbb{E}_{x \sim p} \log p(x) - \mathbb{E}_{x \sim p} \log p(x)) \\ &= -(KL(p||p_{\theta}) + H(p)) \end{aligned}$$

Hence, we are minimizing KL divergence ( $H(p)$  is a constant).

# Training explicit density models

**Likelihood-based:** maximize the likelihood of the data under the model (possibly with variational or MCMC approximations)

In this example:

Need to be able to evaluate  
 $\nabla \log p_\theta(x)$

$$\begin{aligned} \max_{\theta} \mathbb{E}_{x \sim p} \log p_\theta(x) &= -(\mathbb{E}_{x \sim p} \log \frac{1}{p_\theta(x)} + \mathbb{E}_{x \sim p} \log p(x) - \mathbb{E}_{x \sim p} \log p(x)) \\ &= -(KL(p||p_\theta) + H(p)) \end{aligned}$$

Hence, we are minimizing KL divergence ( $H(p)$  is a constant).

# Unnormalized density models

A frequent paradigm is for probabilistic models to have the form:

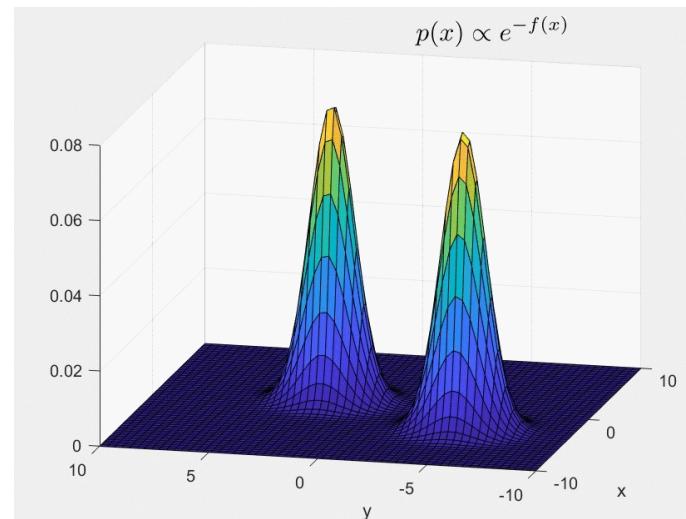
$$p_\theta(x) \propto \exp(-E_\theta(x))$$

where  $E_\theta(x)$  has some easy to evaluate form.

It's an easy way to convert an **energy**  $E_\theta$  to a **probability distribution**.

(The above distribution puts more mass on points with lower energy.)

Furthermore, by scaling  $E_\theta(x)$  you can regulate the “sharpness” of the distribution.)



# Unnormalized density models

A frequent paradigm is for probabilistic models to have the form:

$$p_\theta(x) \propto \exp(-E_\theta(x))$$

where  $E_\theta(x)$  has some easy to evaluate form.

To evaluate  $p_\theta(x)$ , we need to calculate  $\int_x \exp(-E_\theta(x))$  or  $\sum_x \exp(-E_\theta(x))$

How about gradient? Denoting above quantity  $Z_\theta$ , we have:

$$\nabla \log p_\theta(x) = -\nabla E_\theta(x) + \nabla \log Z_\theta$$

$$\begin{aligned} \nabla_\theta \log Z_\theta &= \frac{1}{Z_\theta} \nabla_\theta Z_\theta &= \frac{1}{Z_\theta} \int_x \exp(-E_\theta(x)) \nabla_\theta (-E_\theta(x)) \\ &= \mathbb{E}_{p_\theta}[-\nabla_\theta E_\theta(x)] \end{aligned}$$

We'd like to draw samples from  $p_\theta$

# Interpretation of gradient descent for unnormalized density models

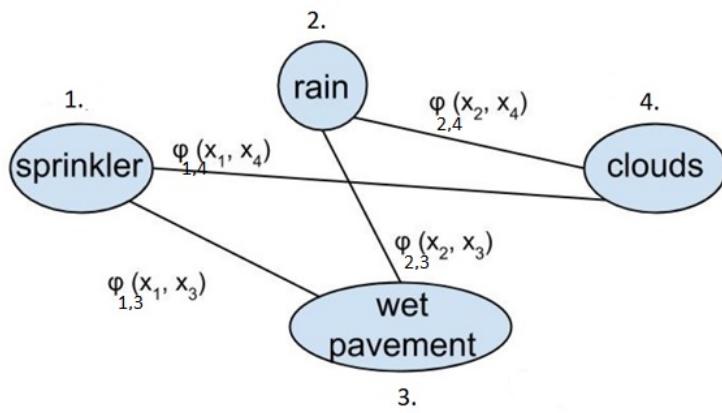
$$\nabla_{\theta} \left( \frac{1}{n} \sum_{i=1}^n \log p_{\theta}(x_i) \right) = \frac{1}{n} \left( \sum_i -\nabla_{\theta} E_{\theta}(x_i) \right) - \mathbb{E}_{p_{\theta}}[-\nabla_{\theta} E_{\theta}(x)]$$

$$\approx \mathbb{E}_{p_{data}}[-\nabla_{\theta} E_{\theta}(x)] - \mathbb{E}_{p_{\theta}}[-\nabla_{\theta} E_{\theta}(x)]$$

***Goal of the algorithm:*** Try to make the expectation of the energy match

# **Examples of unnormalized density models**

# Undirected graphical models or Markov Random Fields (MRFs)



A *pairwise undirected graphical model (MRF)* expresses a distribution as product of local **potentials**  $\phi_{ij}$  (**interactions**), for example

$$p(x) = \frac{1}{Z} \prod_{(i,j) \in E(G)} \phi_{ij}(x_i, x_j)$$

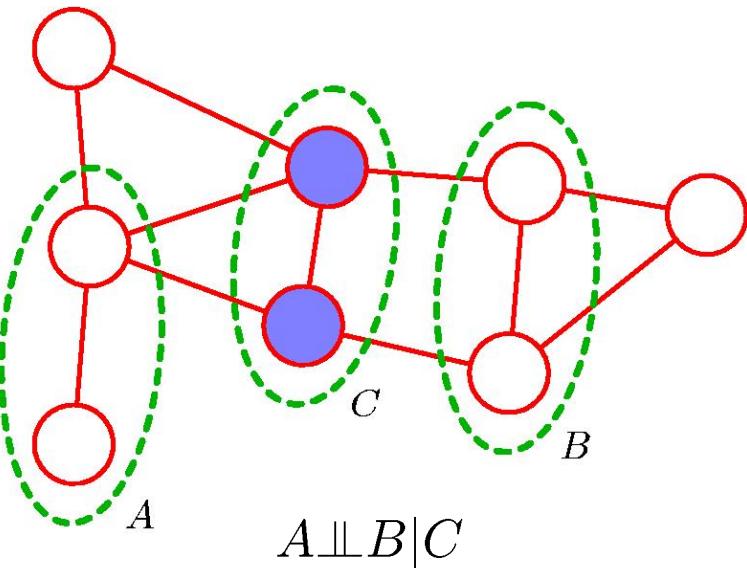
Encodes “*soft constraints*”: the distribution tries to find a *good balance* in satisfying the (possibly conflicting) influences of the potentials.

**Sampling from an MRF is hard:** *partition function*

$$Z = \sum_x \prod_{(i,j) \in E(G)} \phi_{ij}(x_i, x_j) \text{ is intractable.}$$

(In fact, there are simple cases where classical results in TCS show it is #P-hard to calculate.)

# Conditional independence in MRFs



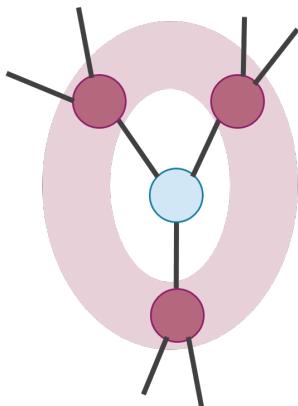
*The independence structure of variables is “captured” by the graph.*

Nodes in A, B are independent, given a set of nodes C separating A, B

$$p(x_A \mid x_C, x_B) = p(x_A \mid x_C)$$

Equivalently:

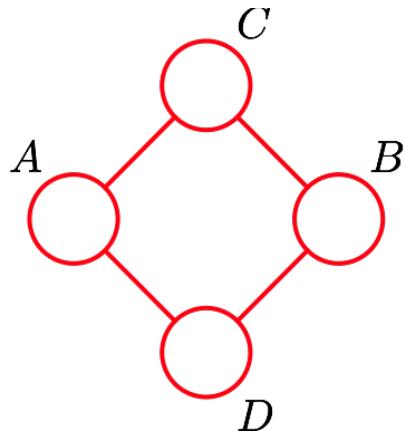
$$p(x_A, x_B \mid x_C) = p(x_A \mid x_C) p(x_B \mid x_C)$$



**Special case:** node is independent of the rest of the graph, given values of the neighbors

$$p(x_v \mid x_{N(v)}, x_{V \setminus \{N(v), v\}}) = p(x_v \mid x_{N(v)})$$

# Instance of a more general principle



More generally, we can describe the family of distributions whose conditional independence structure “tracks” a given (undirected) graph.

The way to formalize this is in terms of **maximal cliques C** (clique = fully connected subset of nodes) of the graph:

$$p(x) \propto \prod_C \phi_C(x_C)$$

For example, the joint distribution above factorizes as:

$$p(A, B, C, D) \propto \phi_{AC}(A, C)\phi_{BC}(B, C)\phi_{AC}(B, D)\phi_{AD}(A, D)$$

# Maximal cliques

The subsets that are used to define the potential functions are represented by maximal cliques in the undirected graph.

**Clique:** a subset of nodes such that there exists an edge between all pairs of nodes in a subset.

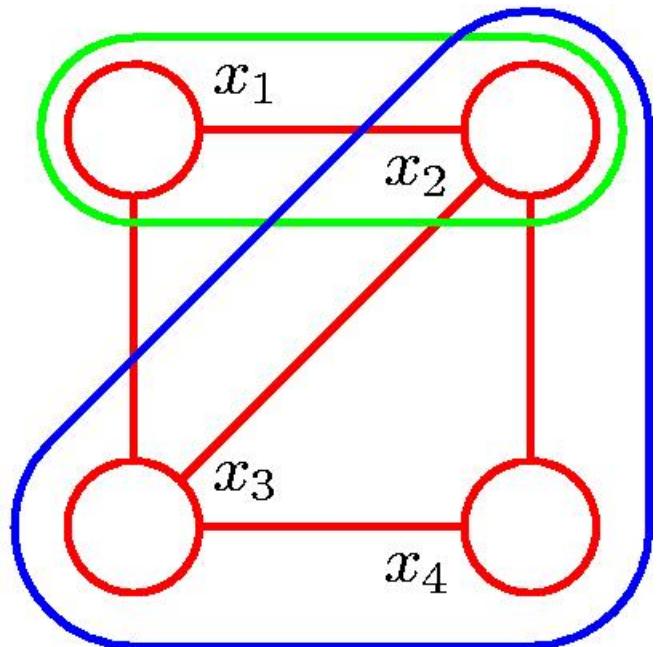
**Maximal Clique:** a clique such that it is not possible to include any other nodes in the set without it ceasing to be a clique.

This graph has 5 cliques:

$$\{x_1, x_2\}, \{x_2, x_3\}, \{x_3, x_4\}, \\ \{x_4, x_2\}, \{x_1, x_3\}.$$

Two maximal cliques:

$$\{x_1, x_2, x_3\}, \{x_2, x_3, x_4\}.$$



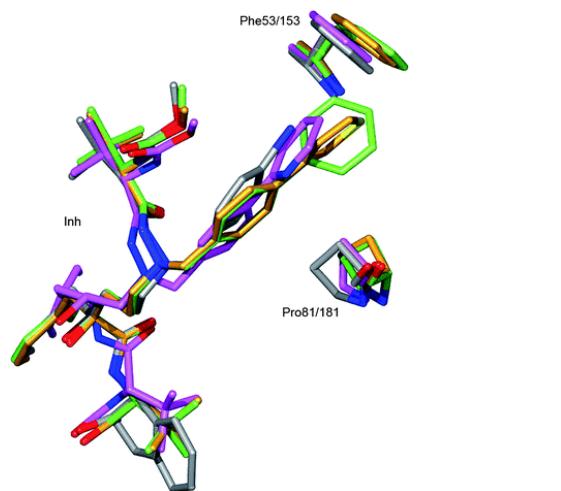
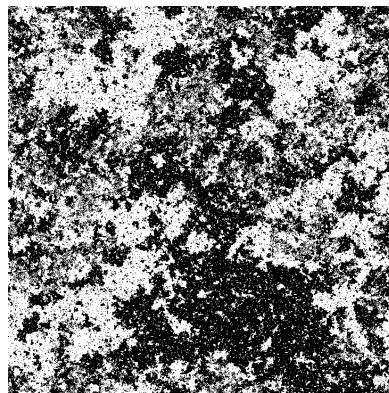
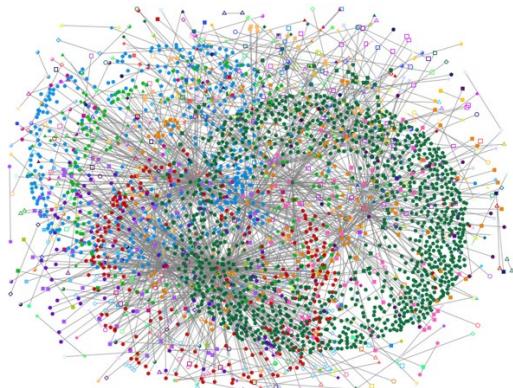
# Hammersley-Clifford theorem

Consider the following sets of distributions:

- The set of distributions consistent with the **conditional independence relationships** defined by the undirected graph.
- The set of distributions consistent with the **factorization** defined by potential functions on **maximal cliques** of the graph.

**Hammersley-Clifford theorem:** these two sets of distributions are the same.

# Some applications



# Example: multivariate Gaussian

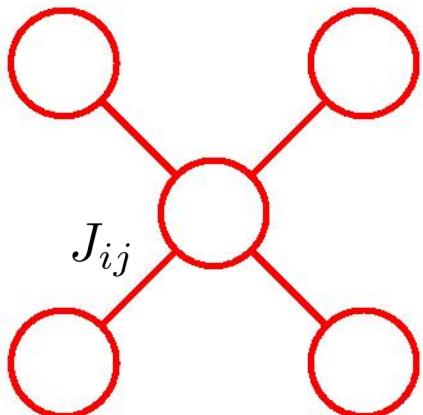
Recall the **multivariate Gaussian**

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}$$

The term inside the exponential is **quadratic**: namely, we can write

$$P(\mathbf{x}) = \frac{1}{Z} \exp \left( -\frac{1}{2} \mathbf{x}^T J \mathbf{x} + \mathbf{g}^T \mathbf{x} \right),$$

$$\text{where } J = \boldsymbol{\Sigma}^{-1}, \quad \boldsymbol{\mu} = J^{-1} \mathbf{g}.$$



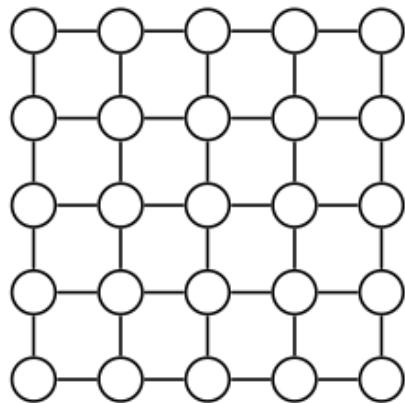
$$\mathbf{x}^T J \mathbf{x} = \sum_i J_{ii} x_i^2 + 2 \sum_{ij \in E} J_{ij} x_i x_j,$$

Thus, the interactions are given by the precision matrix  $J$ .

(Note: precision mx being sparse **does not** imply the covariance mx is sparse.)

# Example: Ising models

MRFs with binary variables are sometimes called **Ising models** in statistical mechanics, and **Boltzmann machines** in machine learning literature.



Denoting the binary valued variable at node  $j$  by  $x_j \in \{\pm 1\}$ , the **Ising model** for the joint probabilities is given by:

$$P_\theta(\mathbf{x}) = \frac{1}{Z(\theta)} \exp \left( \sum_{ij \in E} x_i x_j \theta_{ij} + \sum_{i \in V} x_i \theta_i \right)$$

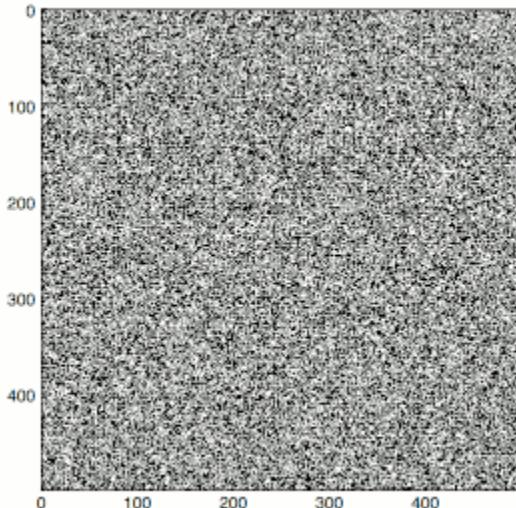
The conditional distribution is given by logistic (*only depends on nbrhood!*):

$$P_\theta(x_i = 1 | \mathbf{x}_{-i}) = \frac{1}{1 + \exp(-\theta_i - \sum_{j \in N(i)} x_j \theta_{ij})}, \text{ where } \mathbf{x}_{-i} \text{ denotes all nodes except for } i.$$

If  $\theta_{ij} \geq 0$ : the nodes  $i, j$ , prefer to be the same. If  $\theta_{ij} \leq 0$ : they prefer to be different.

# Example: Ferromagnetic Ising models

$$P_{\theta}(\mathbf{x}) = \frac{1}{\mathcal{Z}(\theta)} \exp \left( \sum_{ij \in E} x_i x_j \theta_{ij} + \sum_{i \in V} x_i \theta_i \right)$$

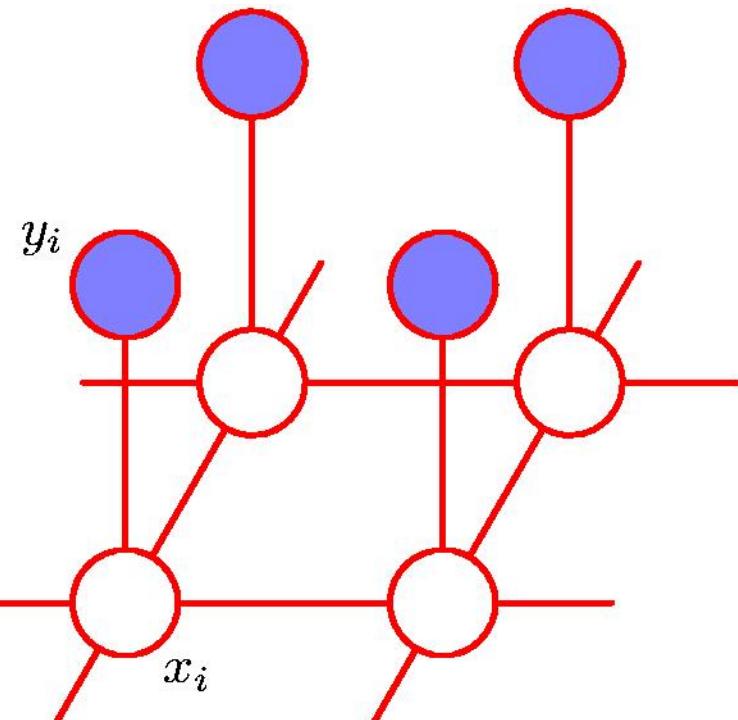


If  $\theta_{ij} \geq 0$ : the model is called ferromagnetic, and is used in physics to model the atomic structure (spins) of iron.

# Example: Image Denoising

*Noise removal from a binary image:*

Let the observed noisy image be described by an array of binary pixel values:  $y_j \in \{-1, +1\}$ ,  $i=1, \dots, D$ .



We take a noise-free image  $x_j \in \{-1, +1\}$ , randomly flip the sign of pixels with some small probability.

Bias term

$$E(\mathbf{x}, \mathbf{y}) = h \sum_i x_i - \beta \sum_{\{i,j\}} x_i x_j$$

Neighboring pixels are likely to have the same sign

$$-\eta \sum_i x_i y_i$$

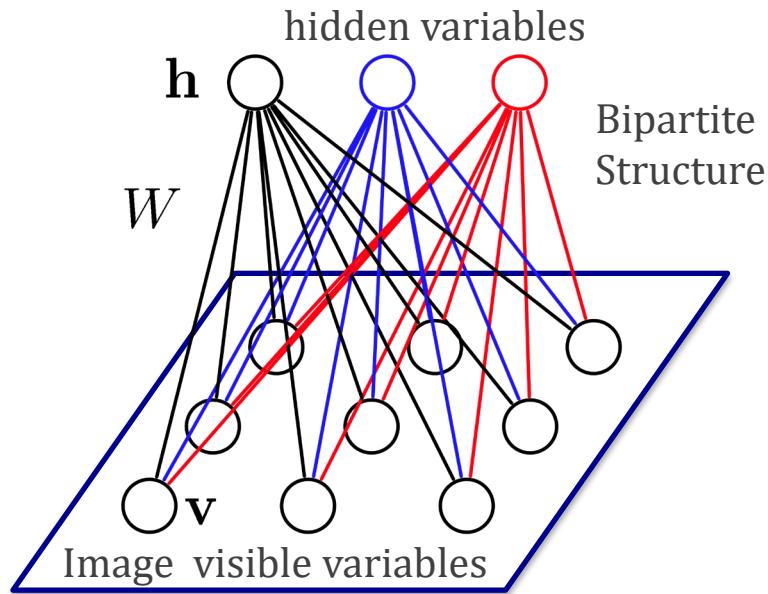
$$p(\mathbf{x}, \mathbf{y}) = \frac{1}{Z} \exp\{-E(\mathbf{x}, \mathbf{y})\}$$

Noisy and clean pixels are likely to have the same sign

# Restricted Boltzmann Machines

A **latent-variable model**: some of the variables the distribution models are not observed (hidden).

We denote visible and hidden variables with vectors  $\mathbf{v}, \mathbf{h}$  respectively:



Visible variables  $\mathbf{v} \in \{0, 1\}^D$   
are connected to hidden variables  $\mathbf{h} \in \{0, 1\}^F$

The energy of the joint configuration:

$$E(\mathbf{v}, \mathbf{h}; \theta) = - \sum_{ij} W_{ij} v_i h_j - \sum_i b_i v_i - \sum_j a_j h_j$$

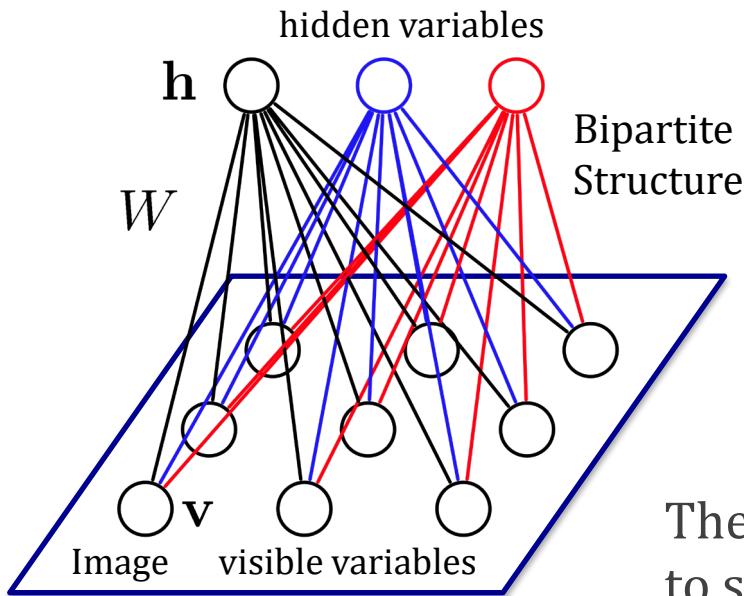
$\theta = \{W, a, b\}$  model parameters.

Probability of the joint configuration is given by the Boltzmann distribution:

$$P_\theta(\mathbf{v}, \mathbf{h}) = \frac{1}{Z(\theta)} \exp(-E(\mathbf{v}, \mathbf{h}; \theta)) = \frac{1}{Z(\theta)} \prod_{ij} e^{W_{ij} v_i h_j} \prod_i e^{b_i v_i} \prod_j e^{a_j h_j}$$

$$Z(\theta) = \sum_{\mathbf{h}, \mathbf{v}} \exp(-E(\mathbf{v}, \mathbf{h}; \theta))$$

# Restricted Boltzmann Machines



*Restricted:* No interaction between hidden variables



The **posterior** over the hidden variables is easy to sample from! (Conditional independence!)

$$P(\mathbf{h}|\mathbf{v}) = \prod_j P(h_j|\mathbf{v}) \quad P(h_j = 1|\mathbf{v}) = \frac{1}{1 + \exp(-\sum_i W_{ij} v_i - a_j)}$$

Similarly: Factorizes

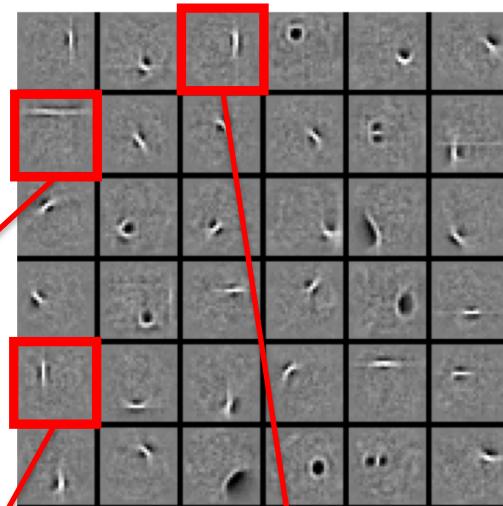
$$P(\mathbf{v}|\mathbf{h}) = \prod_i P(v_i|\mathbf{h}) \quad P(v_i = 1|\mathbf{h}) = \frac{1}{1 + \exp(-\sum_j W_{ij} h_j - b_i)}$$

# Example: learning alphabets using RBMs

Observed Data  
Subset of 25,000 characters



Learned W: “edges”  
Subset of 1000 features



New Image:  $p(h_7 = 1|v)$



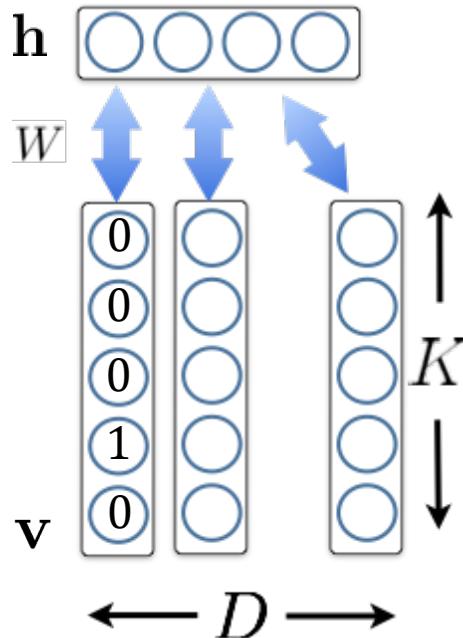
$$= \sigma\left(0.99 \times \begin{array}{c} \text{Small image} \\ \downarrow \end{array} + 0.97 \times \begin{array}{c} \text{Small image} \\ \downarrow \end{array} + 0.82 \times \begin{array}{c} \text{Small image} \\ \downarrow \end{array} \dots\right)$$

$$\sigma(x) = \frac{1}{1+\exp(-x)}$$

Logistic Function: Suitable for  
modeling binary images

Represent:  as  $P(\mathbf{h}|\mathbf{v}) = [0, 0, 0.82, 0, 0, 0.99, 0, 0 \dots]$

# Example: inferring topic structure using RBMs



$$P_{\theta}(\mathbf{v}, \mathbf{h}) = \frac{1}{Z(\theta)} \exp \left( \sum_{i=1}^D \sum_{k=1}^K \sum_{j=1}^F W_{ij}^k v_i^k h_j + \sum_{i=1}^D \sum_{k=1}^K v_i^k b_i^k + \sum_{j=1}^F h_j a_j \right)$$

**Pair-wise**      **Unary**

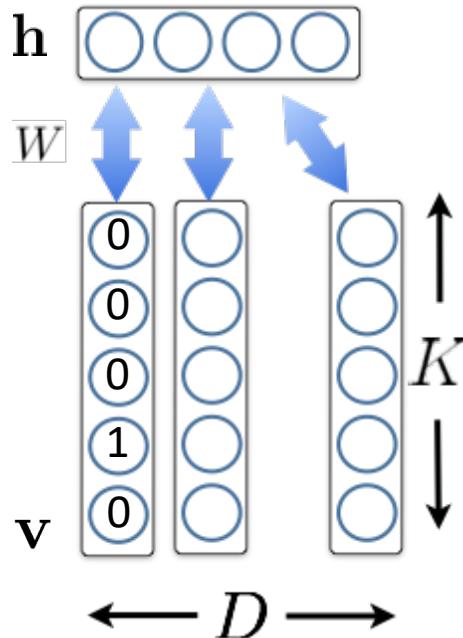
$$\theta = \{W, a, b\}$$

$$P_{\theta}(v_i^k = 1 | \mathbf{h}) = \frac{\exp(b_i^k + \sum_{j=1}^F h_j W_{ij}^k)}{\sum_{q=1}^K \exp(b_i^q + \sum_{j=1}^F h_j W_{ij}^q)}$$

Replicated Softmax Model: *undirected* topic model:

- Stochastic 1-of-K visible variables.
- Stochastic binary hidden variables  $\mathbf{h}$
- Bipartite connections.

# Example: inferring topic structure using RBMs



$$P_{\theta}(v, h) = \frac{1}{Z(\theta)} \exp \left( \sum_{i=1}^D \sum_{k=1}^K \sum_{j=1}^F W_{ij}^k v_i^k h_j + \sum_{i=1}^D \sum_{k=1}^K v_i^k b_i^k + \sum_{j=1}^F h_j a_j \right)$$

$$\theta = \{W, a, b\}$$

$$P_{\theta}(v_i^k = 1 | h) = \frac{\exp \left( b_i^k + \sum_{j=1}^F h_j W_{ij}^k \right)}{\sum_{q=1}^K \exp \left( b_i^q + \sum_{j=1}^F h_j W_{ij}^q \right)}$$



Reuters dataset:  
804,414 **unlabeled**  
newswire stories  
Bag-of-Words



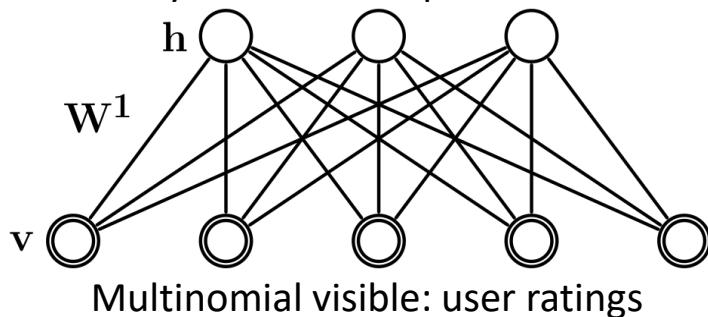
Learned features: "topics"

russian	clinton	computer	trade	stock
russia	house	system	country	wall
moscow	president	product	import	street
yeltsin	bill	software	world	point
soviet	congress	develop	economy	dow

# Example: movie predictions using RBMs

$$P_{\theta}(\mathbf{v}, \mathbf{h}) = \frac{1}{\mathcal{Z}(\theta)} \exp \left( \sum_{ijk} W_{ij}^k v_i^k h_j + \sum_{ik} b_i^k v_i^k + \sum_j a_j h_j \right)$$

Binary hidden: user preferences



Netflix dataset:

480,189 users

17,770 movies

Over 100 million ratings



Learned features: ``genre''

Fahrenheit 9/11  
Bowling for Columbine  
The People vs. Larry Flynt  
Canadian Bacon  
La Dolce Vita

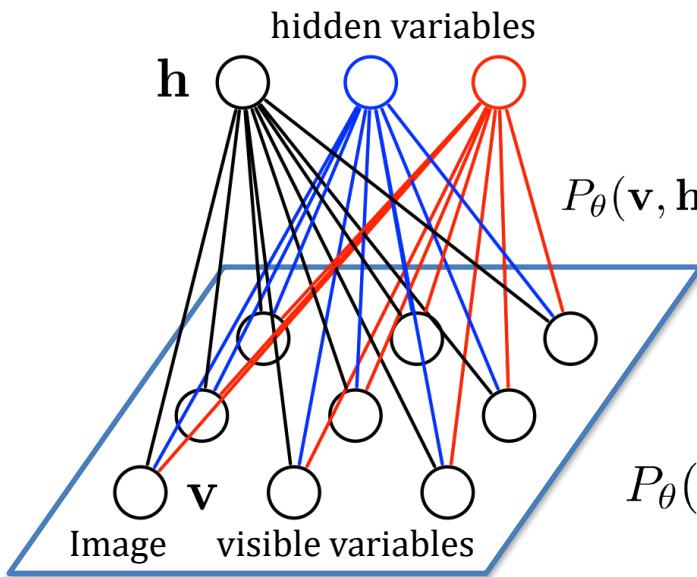
Independence Day  
The Day After Tomorrow  
Con Air  
Men in Black II  
Men in Black

Friday the 13th  
The Texas Chainsaw Massacre  
Children of the Corn  
Child's Play  
The Return of Michael Myers

Scary Movie  
Naked Gun  
Hot Shots!  
American Pie  
Police Academy

**State-of-the-art** performance  
on the Netflix dataset.

# Example: RBMs for image data



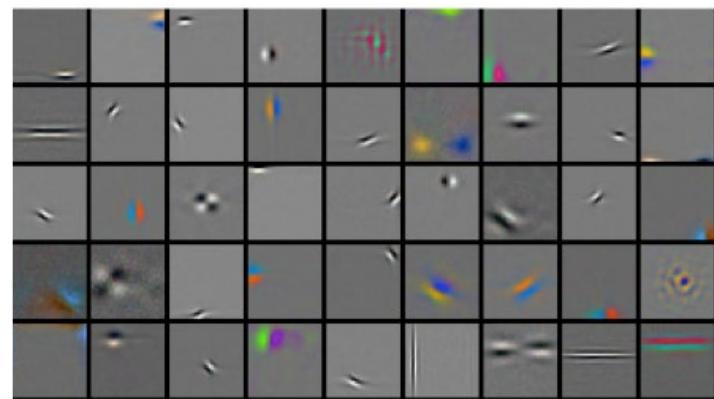
$$P_{\theta}(\mathbf{v}, \mathbf{h}) = \frac{1}{Z(\theta)} \exp \left( \sum_{i=1}^D \sum_{j=1}^F W_{ij} h_j \frac{v_i}{\sigma_i} + \sum_{i=1}^D \frac{(v_i - b_i)^2}{2\sigma_i^2} + \sum_{j=1}^F a_j h_j \right)$$

**Pair-wise**      **Unary**

$$\theta = \{W, a, b\}$$

$$P_{\theta}(\mathbf{v}|\mathbf{h}) = \prod_{i=1}^D P_{\theta}(v_i|\mathbf{h}) = \prod_{i=1}^D \mathcal{N} \left( b_i + \sum_{j=1}^F W_{ij} h_j, \sigma_i^2 \right)$$

Learned features (out of 10,000)



4 million **unlabelled** images

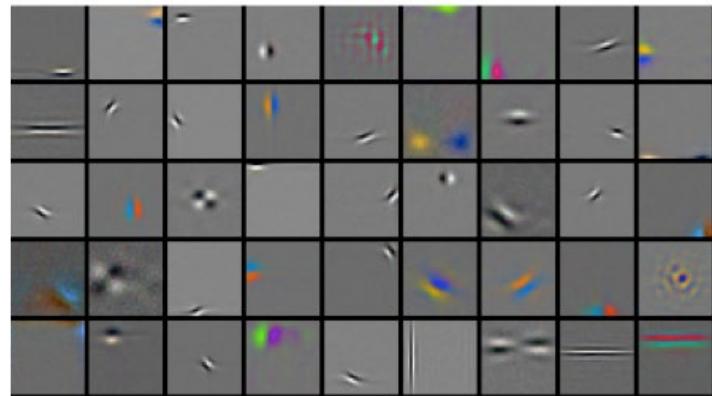


# Example: RBMs for image data

4 million **unlabelled** images



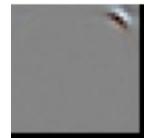
Learned features (out of 10,000)



New Image

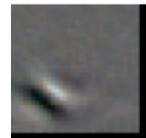
$$p(h_7 = 1|v)$$

$$= 0.9 * \quad \quad \quad$$



$$p(h_{29} = 1|v)$$

$$+ \quad \quad \quad 0.8 * \quad \quad \quad$$



$$+ 0.6 * \quad \quad \quad \dots$$



# Continuous-space energy-based models (EBMs)

If the distribution  $p_\theta(x) \propto \exp(-E_\theta(x))$  has as domain  $\mathbb{R}^d$ , an easy choice is  $E_\theta$  is a neural net of some kind.

These have scaled up *only very recently* to real-life data, e.g. images.

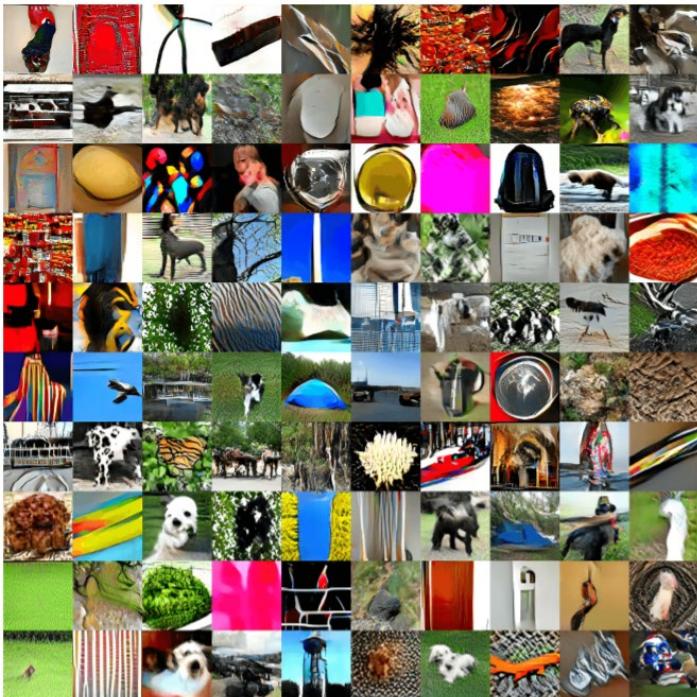
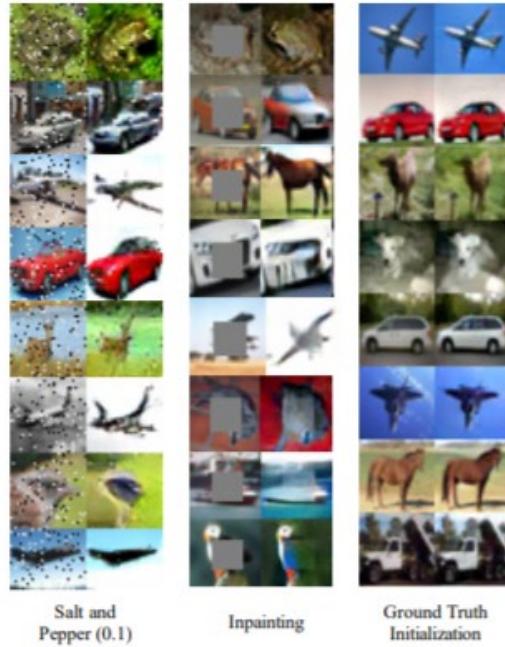


Figure from (Du, Mordatch '20)

Figure 18: MCMC samples from conditional ImageNet128x128 models

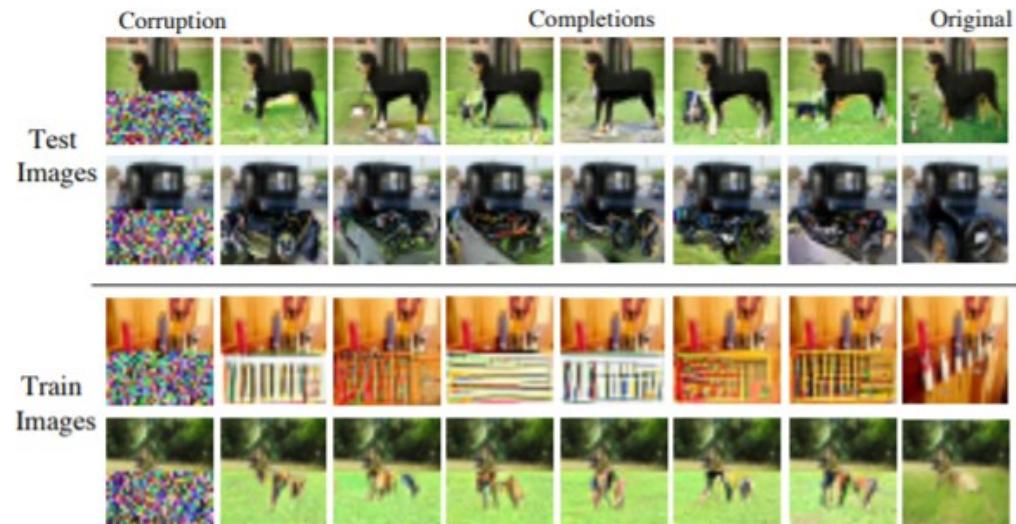
# Example: denoising using EBMs



Starting from a noised image, find the “closest” image via MCMC. (Stay tuned!) Indicates good coverage.

Figure 5: EBM image restoration on images in the **test** set via MCMC. The right column shows failure (approx. 10% objects change with ground truth initialization and 30% of objects change in salt/pepper corruption or inpainting. Bottom two rows shows worst case of change.)

# Example: image completion using EBMs



Testing overfitting via image completion.

Figure 7: Illustration of image completions on conditional ImageNet model. Our models exhibit diversity in inpainting.

# Example: “logical” operators using energy functions

(Du, Li, Mordatch '20)

**Concept conjunction:**

$$p(x|c_1 \text{ and } c_2 \dots \text{ and } c_n) := \prod_i p(x|c_i) = e^{-\sum_i E_i(x)}$$

**Concept disjunction:**

$$\begin{aligned} & p(x|c_1 \text{ or } c_2 \dots \text{ or } c_n) \\ & := \sum P(x|c_i) \propto^* \sum_i e^{-E_i(x)} = e^{\log \sum \exp(-E_1(x), \dots, -E_n(x))} \end{aligned}$$

# Example: “logical” operators using energy functions



Figure 3: Combinations of different attributes on CelebA via concept conjunction. Each row adds an additional energy function. Images on the first row are conditioned on young, while images on the last row are conditioned on young, female, smiling, and wavy hair.



Figure 5: Examples of recursive compositions of disjunction, conjunction, and negation on the CelebA dataset.

Figure from (Du, Li, Mordatch '20)

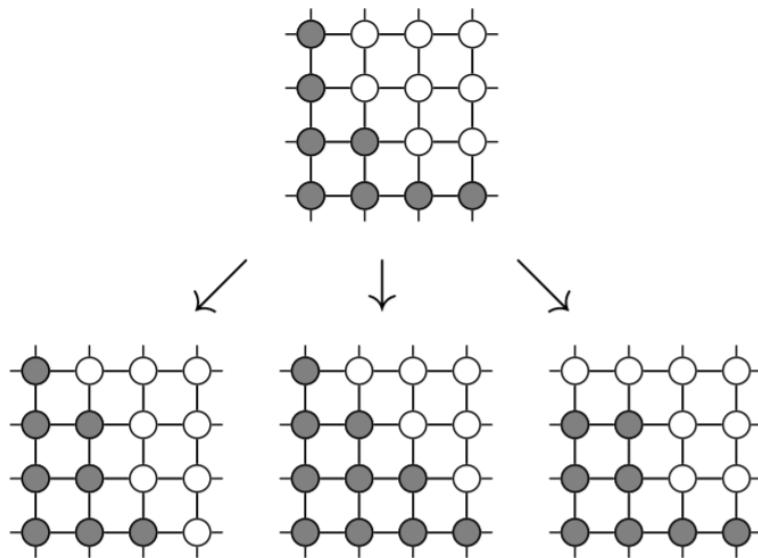
# **Basics of Markov Chains: tools for sampling from unnormalized densities**

# Random walks over discrete domains

# Sampling via random walks

**Goal:** Sample from distribution given up to constant of proportionality.

*Idea:* explore domain via *random, local* moves



*Hope:* enough moves  $\Rightarrow$  the random process “forgets” starting point,  
follows the distr. we are  
trying to sample.

# Preamble: some simple distributions to sample

*Most univariate distributions:*

**Uniform[0,1]:** `np.random()` (*Mersenne Twister based*)

**Bernoulli(1/2):** (*if `np.random() < ½`, output 1; else output 0*)

**Bernoulli(p):** (*if `np.random() < p`, output 1; else output 0*)

**Standard univariate Gaussian:** (*Box-Muller transform of uniform*)

*.... (bruteforce transforms work in 1d)*

# Preamble: some simple distributions to sample

*Extremely few multivariate distributions:*

**Product distributions:**  $P(x_1, x_2, \dots, x_n) = \prod_i P(x_i)$  (*sample coordinates independently*)

“Tractably factorized” distributions:  $P(x_1, x_2, \dots, x_n) = \prod_i P(x_i | x_{<i})$  for which factors are some easy to sample distribution (e.g. Bernoulli, Gaussian, etc.)

**Standard Gaussian** (*product of standard univariate Gaussians*)

**Any Gaussian:** a sample from  $\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$  can be generated as follows

Sample  $\mathbf{x} \sim \mathcal{N}(0, \mathbf{I})$ . Output  $\mathbf{y} = \boldsymbol{\mu} + \boldsymbol{\Sigma}^{1/2} \mathbf{x}$ .

# Sampling via random walks

**Goal:** Sample from distribution given up to constant of proportionality.

*Definition:* A set of random variables  $(X_1, X_2, \dots, X_T)$  is **Markov** if  
 $\forall t: P(X_t | X_{<t}) = P(X_t | X_{t-1})$

It is homogeneous if  $P(X_t | X_{t-1})$  doesn't depend on t.

We can describe a homogeneous Markov process on a discrete domain  $\mathcal{X}$  by a **transition matrix**  $T \in \mathbb{R}_+^{|\mathcal{X}| \times |\mathcal{X}|}: T_{ij} = P(X_{t+1} = j | X_t = i)$

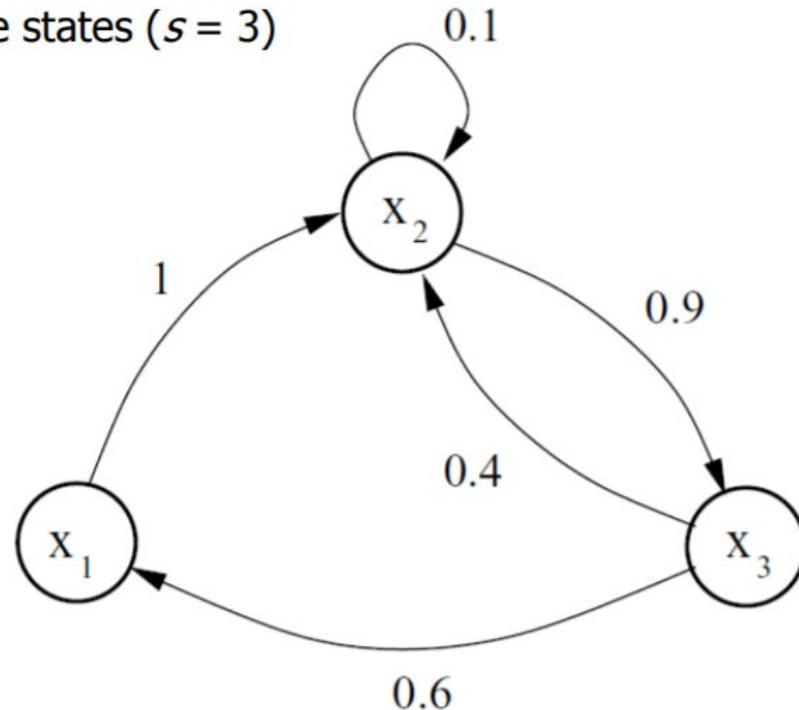
Clearly,  $\forall i, \sum_j T_{ij} = 1$ . We will also call such process a Markov Chain/  
Markov random walk.

# Example

**Markov chain** with three states ( $s = 3$ )

$$T = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0.1 & 0.9 \\ 0.6 & 0.4 & 0 \end{bmatrix}$$

**Transition matrix**



**Transition graph**

# Stationary distribution

**Stationary distribution:** a distribution  $\pi = (\pi_1, \dots \pi_{|\mathcal{X}|})$  is stationary for a Markov walk if  $\pi T = \pi$ .

In other words: if we start with a sample of  $\pi$  and transition according to  $T$ , we end with a sample following  $\pi$  as well.

$$(0.22, 0.41, 0.37) \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0.1 & 0.9 \\ 0.6 & 0.4 & 0 \end{bmatrix} = (0.22, 0.41, 0.37)$$

Stationary distribution need not be unique: e.g.  $T$  is the identity matrix.

Many Markov Chains have unique stationary distributions: after taking many steps, starting with any distribution, we get to the same distribution

$\forall p_0, \lim_{t \rightarrow \infty} p_0 T^t = \pi$    In other words, eventually, the chain “forgets” the starting point.

# Stationary distribution

*Stationary distribution:* a distribution  $\pi = (\pi_1, \dots \pi_{|\chi|})$  is stationary for a Markov walk if  $\pi T = \pi$ .

Many Markov Chains have unique stationary distributions: after taking many steps, starting with any distribution, we get to the same distribution

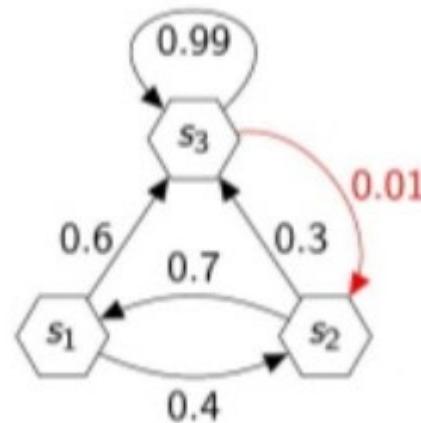
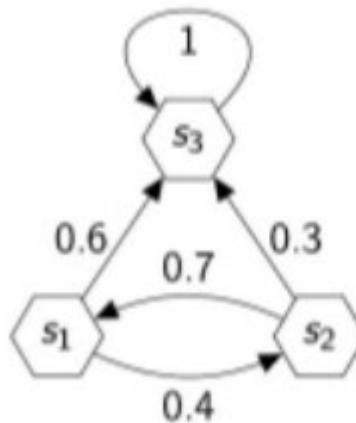
$$\forall p_0, \lim_{t \rightarrow \infty} p_0 T^t = \pi$$

**Name of the game:** if we wish to sample from some  $\pi$ , design a Markov Chain which has  $\pi$  as stationary distribution.

If we run chain long enough (??), we can draw samples from something close to  $\pi$

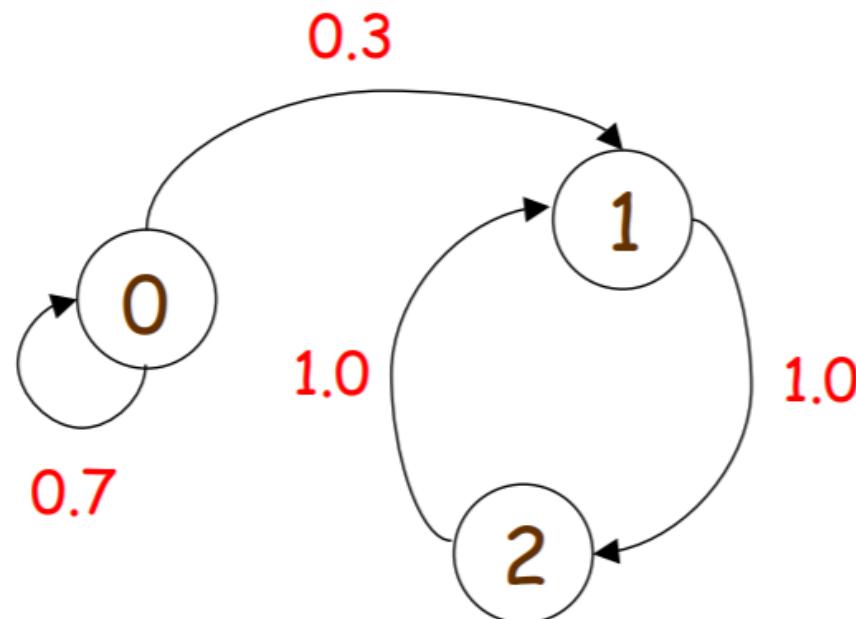
# Conditions for having a unique stationary distribution

*Potential problem:* transition graph is not connected.



# Conditions for having a unique stationary distribution

*Potential problem:* there are cycles in graph



# Conditions for having a unique stationary distribution

These are all the possible problems!

**Irreducibility:** there is a path that transitions from any state to any other.

For each pairs of states (i,j), there is a positive probability, starting in state i, that the process will ever enter state j.

= Transition graph is connected;

**Aperiodicity:** random walk doesn't get trapped in cycles.

A state i is aperiodic if there exists n s.t.,  $\forall n' \geq n, P(X_{n'} = i | X_0 = i) > 0$ .

If all states are aperiodic, chain is called aperiodic.

**Thm:** for any *irreducible+aperiodic* Markov chain there is a unique  $\pi$ , s.t.

$$\forall p_0, \lim_{t \rightarrow \infty} p_0 T^t = \pi$$

# Detailed balance

**Useful sufficient condition** for  $\pi$  to be a stationary distribution:  
detailed balance.

$$\pi_i T_{ij} = \pi_j T_{ji}, \forall (i, j)$$

*Proof:*

$$\begin{aligned} (\pi T)_i &= \sum_j \pi_j T_{ji} &= \sum_j \pi_i T_{ij} \\ &= \pi_i \sum_j T_{ij} \\ &= \pi_i \end{aligned}$$

# Metropolis-Hastings

Suppose we are trying to sample from  $\pi$  defined over a domain of size  $m$  (think  $m$  is very large, like in Ising models), up to a constant of proportionality:

$$\pi_i = \frac{b(i)}{Z}, Z = \sum_{i=1}^m b(i)$$

Metropolis-Hastings: random walk assuming an “easy-to-sample from” transition kernel  $q(i,j)$ , along with “corrections”.

# Metropolis-Hastings

Suppose we have an easy to sample from “transition kernel”  $q(i,j)$ .

Consider the following random walk, for some  $\alpha(i,j)$  we will pick:

$$\Pr(X_n = j | X_{n-1} = i) =$$

- 1., from state  $i$  go to state  $j$  with prob.  $q(i,j)$
- 2.,  $\begin{cases} \text{with prob } 1 - \alpha(i,j) \text{ go back to state } i, \\ \text{with prob } \alpha(i,j) \text{ stay in state } j. \end{cases}$

Then, we have:

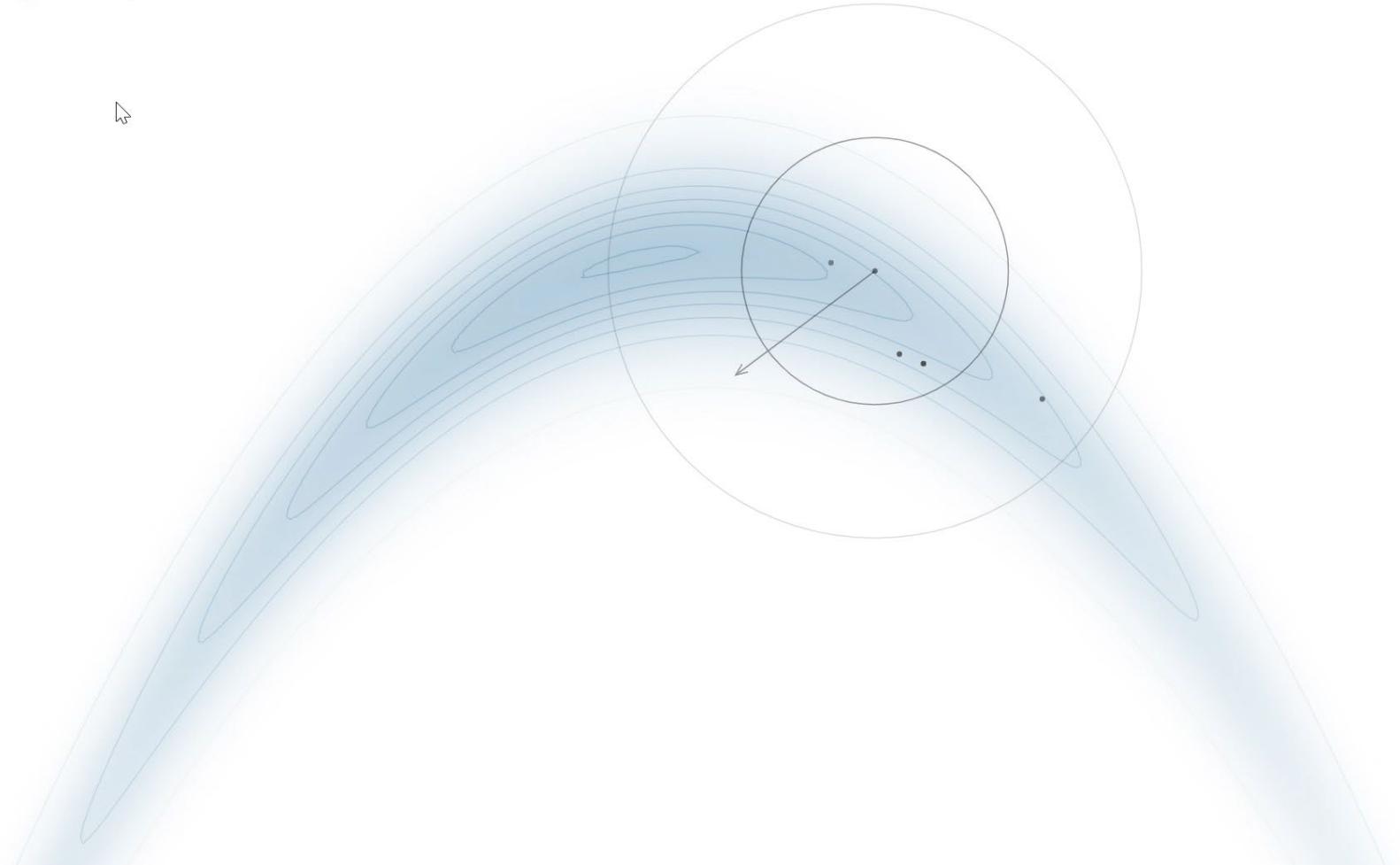
$$P(X_{n+1} = j | X_n = i) = q(i,j)\alpha(i,j) \quad \forall j \neq i$$

$$P(X_{n+1} = i | X_n = i) = q(i,i) + \sum_{k \neq i} q(i,k)(1 - \alpha(i,k))$$

# Metropolis-Hastings

Random walk Metropolis-Hastings

Open Controls



<https://chi-feng.github.io/mcmc-demo/app.html>

# Metropolis-Hastings

## Observation

$$\pi_i P_{ij} = \pi_j P_{ji} \quad \forall j \neq i \Leftrightarrow \pi_i q(i, j) \alpha(i, j) = \pi_j q(j, i) \alpha(j, i) \quad \forall j \neq i \quad (*)$$

**Proof:**  $P_{ij} = P(X_{n+1} = j | X_n = i) = q(i, j) \alpha(i, j) \quad \forall j \neq i$

## Theorem

$$\text{If } \alpha(i, j) = \min\left(\frac{\pi_j q(j, i)}{\pi_i q(i, j)}, 1\right) = \min\left(\frac{b(j)q(j, i)}{b(i)q(i, j)}, 1\right)$$

$\Rightarrow (\pi_1, \dots, \pi_m)$  stationary distribution

**Proof:**

*Note, this only depends on unnormalized distribution*

*( $b(i)$  values)*

$$\text{If } \alpha(i, j) = \frac{\pi_j q(j, i)}{\pi_i q(i, j)} \Leftrightarrow \alpha(j, i) = 1$$

=> Detailed balance (\*) holds

# Gibbs sampling

Consider sampling a distribution over  $n$  variables  $\mathbf{x} = (x_1, x_2, \dots, x_n)$ , s.t. each of the conditional distributions  $P(x_i | \mathbf{x}_{-i})$  is easy to sample. :

e.g. recall Ising models:  $P_\theta(x_i = 1 | \mathbf{x}_{-i}) = \frac{1}{1 + \exp(-\theta_i - \sum_{ij \in E} x_j \theta_{ij})}$ ,

A common way to do this is using **Gibbs sampling**:

Repeat:

Let current state be  $\mathbf{x} = (x_1, x_2, \dots, x_n)$

Pick  $i \in [n]$  uniformly at random.

Sample  $x \sim P(X_i = x | \mathbf{x}_{-i})$

Update state to  $\mathbf{y} = (x_1, x_2, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n)$

# Gibbs sampling

Repeat:

Let current state be  $\mathbf{x} = (x_1, x_2, \dots, x_n)$

Pick  $i \in [n]$  uniformly at random.

Sample  $x \sim P(X_i = x | \mathbf{x}_{-i})$

Update state to  $\mathbf{y} = (x_1, x_2, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n)$

Why does it work? Metropolis-Hastings with appropriate kernel!

Let

$$\begin{aligned} q(\mathbf{x}, \mathbf{y}) &= q(\overbrace{(x_1, \dots, x_n)}^{\mathbf{x}}, \overbrace{(x_1, \dots, x_{i-1}, x, x_{i+1}, x_n)}^{\mathbf{y}}) \\ &\doteq \frac{1}{n} P(X_i = x | X_j = x_j, \forall j \neq i) \\ &= \frac{1}{n} \frac{P(\mathbf{y})}{P(X_j = x_j, \forall j \neq i)} \end{aligned}$$

# Gibbs sampling

Why does it work? Metropolis-Hastings with appropriate kernel!

Let

$$\begin{aligned} q(\mathbf{x}, \mathbf{y}) &= q(\overbrace{(x_1, \dots, x_n)}^{\mathbf{x}}, \overbrace{(x_1, \dots, x_{i-1}, x, x_{i+1}, x_n)}^{\mathbf{y}}) \\ &\doteq \frac{1}{n} P(X_i = x | X_j = x_j, \forall j \neq i) \\ &= \frac{1}{n} \frac{P(\mathbf{y})}{P(X_j = x_j, \forall j \neq i)} \end{aligned}$$

Shouldn't we reject occasionally? **No:**

## Theorem

If  $\alpha(i, j) = \min\left(\frac{\pi_j q(j, i)}{\pi_i q(i, j)}, 1\right) = \min\left(\frac{b(j)q(j, i)}{b(i)q(i, j)}, 1\right)$   
 $\Rightarrow (\pi_1, \dots, \pi_m)$  stationary distribution

$$\frac{p(\mathbf{y})q(\mathbf{y}, \mathbf{x})}{p(\mathbf{x})q(\mathbf{x}, \mathbf{y})} = \frac{p(\mathbf{y})\frac{1}{n}\frac{P(\mathbf{x})}{P(Y_j=y_j, j \neq i)}}{p(\mathbf{x})\frac{1}{n}\frac{P(\mathbf{y})}{P(X_j=x_j, j \neq i)}}$$

# Gibbs sampling

Why does it work? Metropolis-Hastings with appropriate kernel!

Let

$$\begin{aligned} q(\mathbf{x}, \mathbf{y}) &= q(\overbrace{(x_1, \dots, x_n)}^{\mathbf{x}}, \overbrace{(x_1, \dots, x_{i-1}, x, x_{i+1}, x_n)}^{\mathbf{y}}) \\ &\doteq \frac{1}{n} P(X_i = x | X_j = x_j, \forall j \neq i) \\ &= \frac{1}{n} \frac{P(\mathbf{y})}{P(X_j = x_j, \forall j \neq i)} \end{aligned}$$

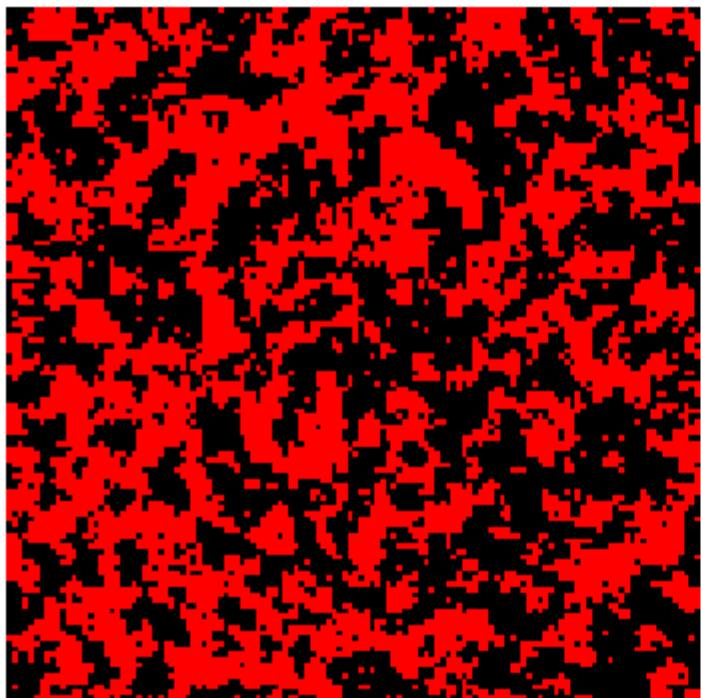
Shouldn't we reject occasionally? **No**:

$$\frac{p(\mathbf{y})q(\mathbf{y}, \mathbf{x})}{p(\mathbf{x})q(\mathbf{x}, \mathbf{y})} = \frac{p(\mathbf{y}) \frac{1}{n} \frac{P(\mathbf{x})}{P(Y_j = y_j, j \neq i)}}{p(\mathbf{x}) \frac{1}{n} \frac{P(\mathbf{y})}{P(X_j = x_j, j \neq i)}} = \frac{p(\mathbf{y})p(\mathbf{x})}{p(\mathbf{x})p(\mathbf{y})} = 1$$

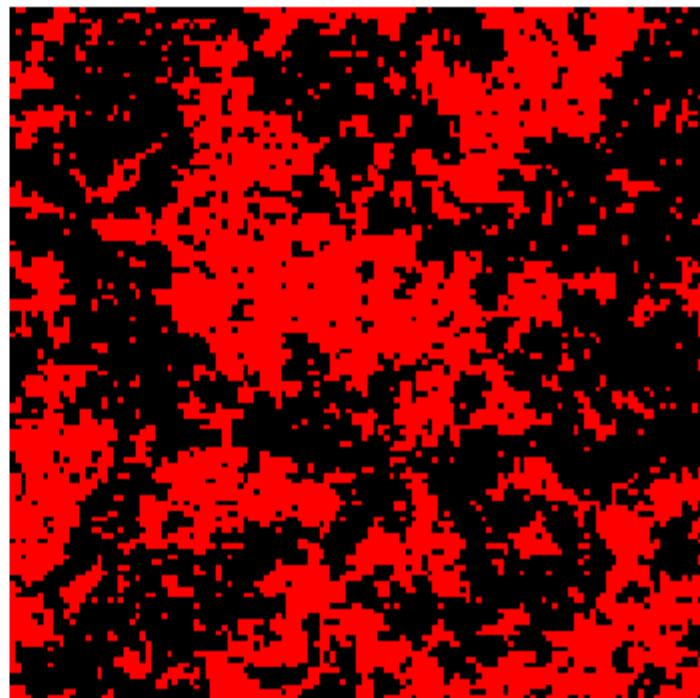
since  $P(X_j = x_j, j \neq i) = P(Y_j = y_j, j \neq i)$

# Gibbs sampling

Gibbs sampler: Step 5



Swendsen-Wang: Step 5



<https://www.math.mcgill.ca/dstephens/MySite/index.html>

# What governs “mixing time”

So far, we've only worried about designing chains s.t.  $\forall p_0, \lim_{t \rightarrow \infty} p_0 T^t = \pi$

But, we're running this in practice, so want for sensible  $t$ ,  $\forall p_0, p_0 T^t \approx \pi$

(Appropriately formalized, this is called *mixing time*.)

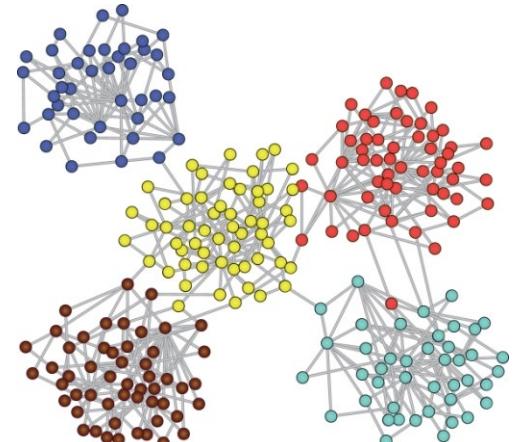
There is no silver bullet for analyzing general transition  $T$ , but one common tool is *conductance*: which essentially says the transition graph doesn't have “bottlenecks”.

The conductance of a subset  $S$  is defined as:

$$\phi(S) = \frac{\sum_{i \in S, j \notin S} T_{ij}}{\sum_{i \in S} \pi_i}$$

(e.g. how easy it is to leave  $S$ , given that we started in  $S$ )

(e.g. the colored sets have poor conductance)



# What governs “mixing time”

So far, we've only worried about designing chains s.t.  $\forall p_0, \lim_{t \rightarrow \infty} p_0 T^t = \pi$

But, we're running this in practice, so want for sensible  $t$ ,  $\forall p_0, p_0 T^t \approx \pi$

(Appropriately formalized, this is called *mixing time*.)

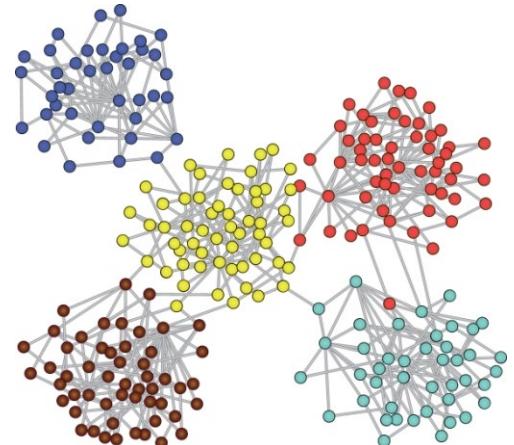
There is no silver bullet for analyzing general transition  $T$ , but one common tool is *conductance*: which essentially says the transition graph doesn't have “bottlenecks”.

The conductance of a subset  $S$  is defined as:

$$\phi(S) = \frac{\sum_{i \in S, j \notin S} T_{ij}}{\sum_{i \in S} \pi_i}$$

It's clear that sets of poor  $\phi(S)$  impede mixing time:

If we start at  $S$ , even with the correct  $\pi$ , it'll take us long to leave  $S$ .



# What governs “mixing time”

So far, we've only worried about designing chains s.t.  $\forall p_0, \lim_{t \rightarrow \infty} p_0 T^t = \pi$

But, we're running this in practice, so want for sensible  $t$ ,  $\forall p_0, p_0 T^t \approx \pi$

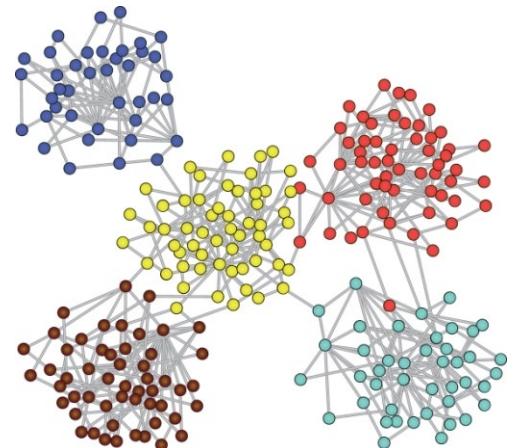
(Appropriately formalized, this is called *mixing time*.)

There is no silver bullet for analyzing general transition  $T$ , but one common tool is *conductance*: which essentially says the transition graph doesn't have “bottlenecks”.

The conductance of a subset  $S$  is defined as:

$$\phi(S) = \frac{\sum_{i \in S, j \notin S} T_{ij}}{\sum_{i \in S} \pi_i}$$

It's clear that sets of poor  $\phi(S)$  impede mixing time:



**The distribution is “multimodal”:** has  $S$ 's that have large probability, but are difficult to transition between.

# What governs “mixing time”

So far, we've only worried about designing chains s.t.  $\forall p_0, \lim_{t \rightarrow \infty} p_0 T^t = \pi$

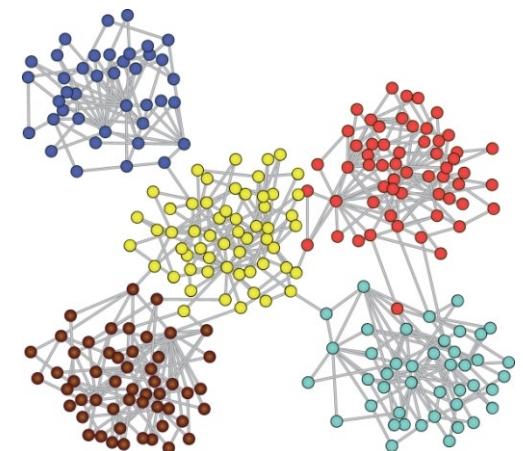
But, we're running this in practice, so want for sensible  $t$ ,  $\forall p_0, p_0 T^t \approx \pi$

(Appropriately formalized, this is called *mixing time*.)

There is no silver bullet for analyzing general transition  $T$ , but one common tool is *conductance*: which essentially says the transition graph doesn't have “bottlenecks”.

The conductance of a subset  $S$  is defined as:

$$\phi(S) = \frac{\sum_{i \in S, j \notin S} T_{ij}}{\sum_{i \in S} \pi_i}$$



Conversely, if  $\phi(S)$  is large for all  $S \Rightarrow$  mixing time is good!

# What governs “mixing time”

So far, we've only worried about designing chains s.t.  $\forall p_0, \lim_{t \rightarrow \infty} p_0 T^t = \pi$

But, we're running this in practice, so want for sensible  $t$ ,  $\forall p_0, p_0 T^t \approx \pi$

(Appropriately formalized, this is called *mixing time*.)

Note common misconception: random walk must visit each state in domain to mix.

This is of course not true! (There does however need to be a reasonable **probability** that some set of moves gets us anywhere in the domain.)

(Otherwise, what would be the point of running a Markov Chain as opposed to brute force calculation of the partition function...)

# Random walks over continuous domains *(Langevin dynamics)*

# Langevin dynamics

Consider sampling from  $p(x) = \frac{1}{Z} \exp(-f(x))$  with support  $\mathbb{R}^d$ ,  $f(x)$  is differentiable and we can efficiently take gradients. (e.g.  $f(x)$  is parametrized by a neural network).

A natural random walk:

Gradient descent   Gaussian noise

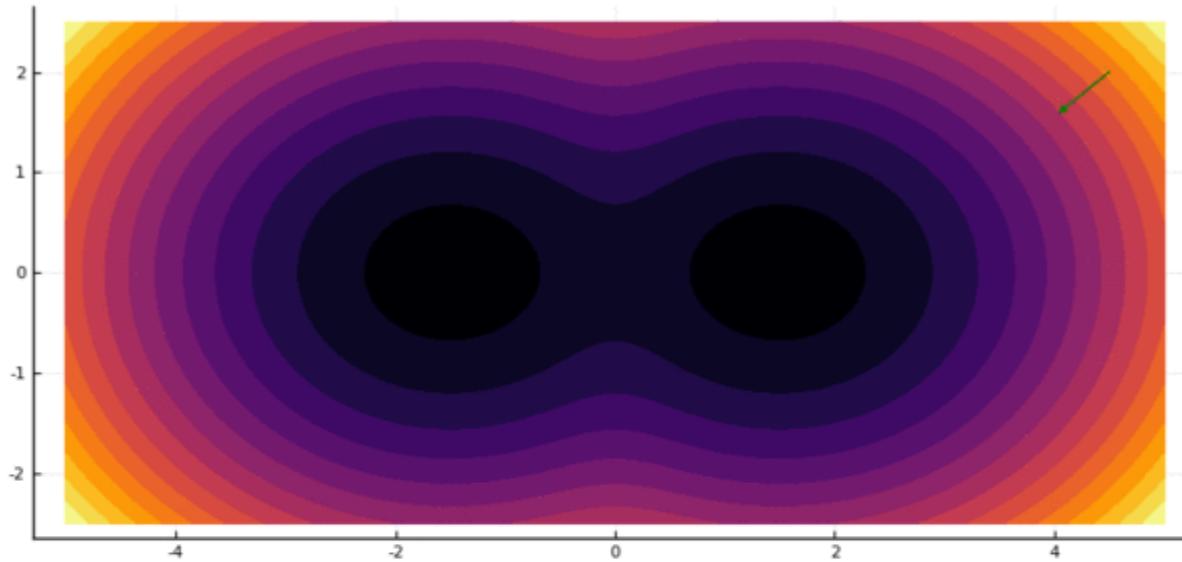
Limit (as  $\eta \rightarrow 0$ ) of: 
$$x_{t+1} = x_t - \eta \nabla f(x_t) + \sqrt{2\eta} \xi_k$$
$$\xi_k \sim N(0, I)$$

Stationary (equilibrium) distr.

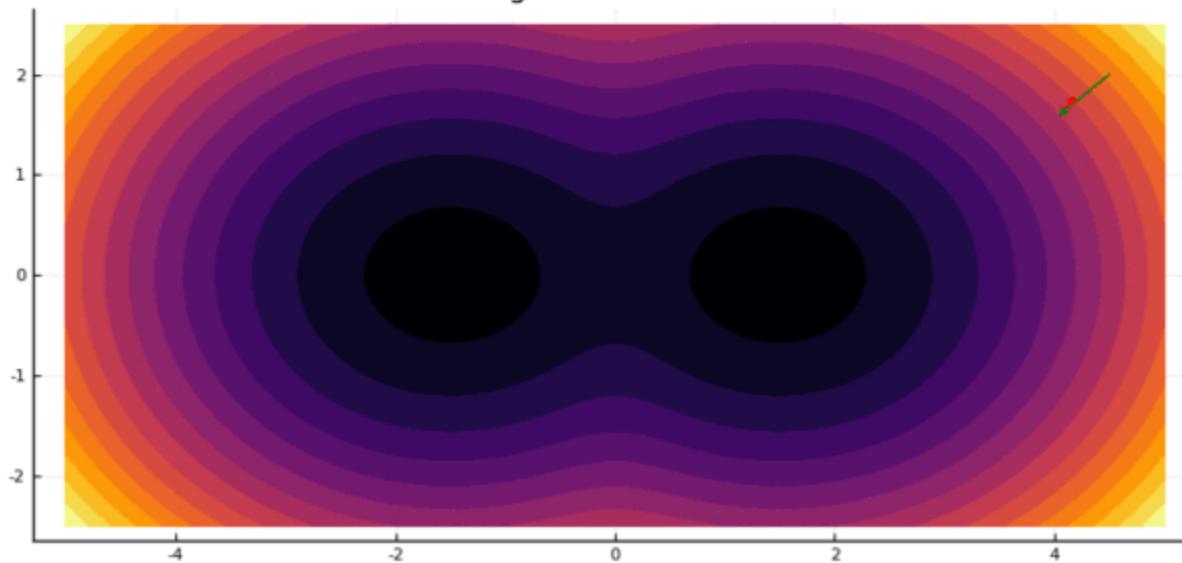
$$p(x) = \frac{1}{Z} \exp(-f(x))$$

# Langevin dynamics

Gradient descent



Langevin Monte Carlo



$$p(x) \propto e^{-f(x)}$$

# The dichotomy

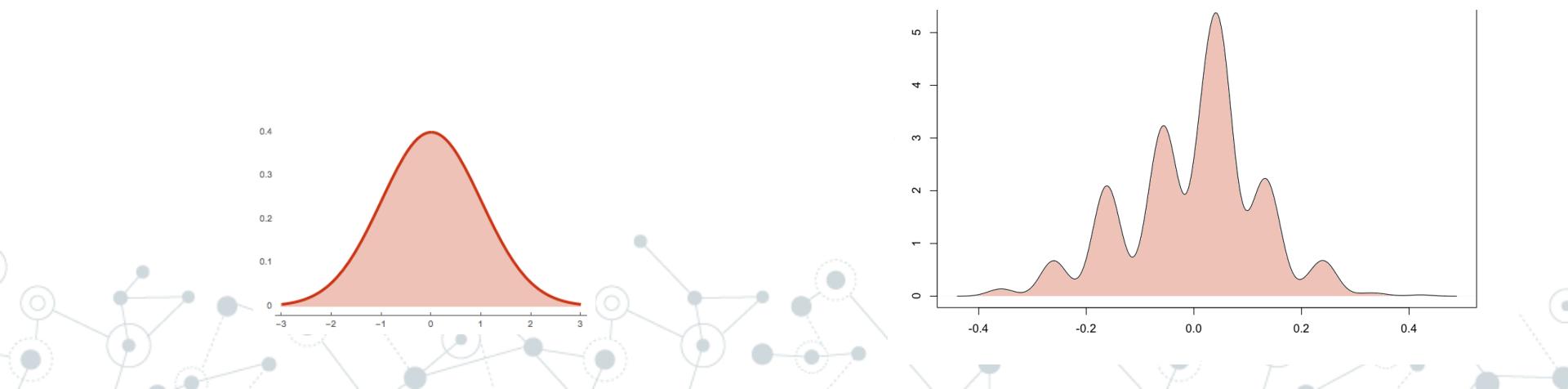
## Log-concave distribution ( $f$ convex)

- Provable algorithms, but practically restrictive (unimodal)

## Non-log-concave distributions

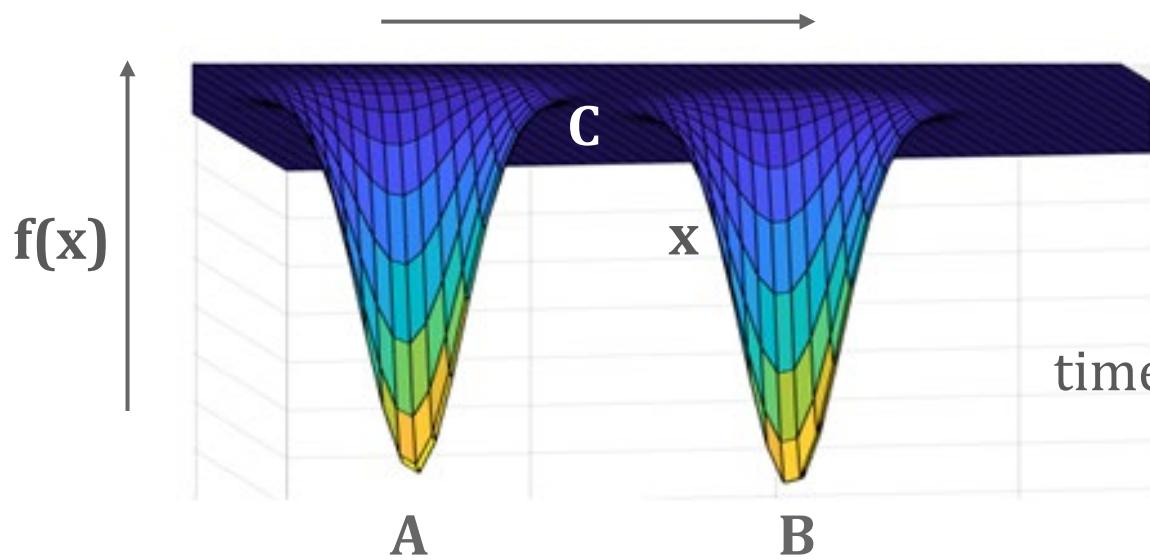
- #P-hard in the worst-case
- Common source of hardness: multimodality

Parallel to optimization: if  $f$  is convex, minimizing  $f$  is easy.  
(All local minima are global  $\Rightarrow$  gradient descent works.)



# Why multimodality is trouble

Sharp hills are hard to climb!



Recall, peaks of  $p =$   
valleys of  $f$



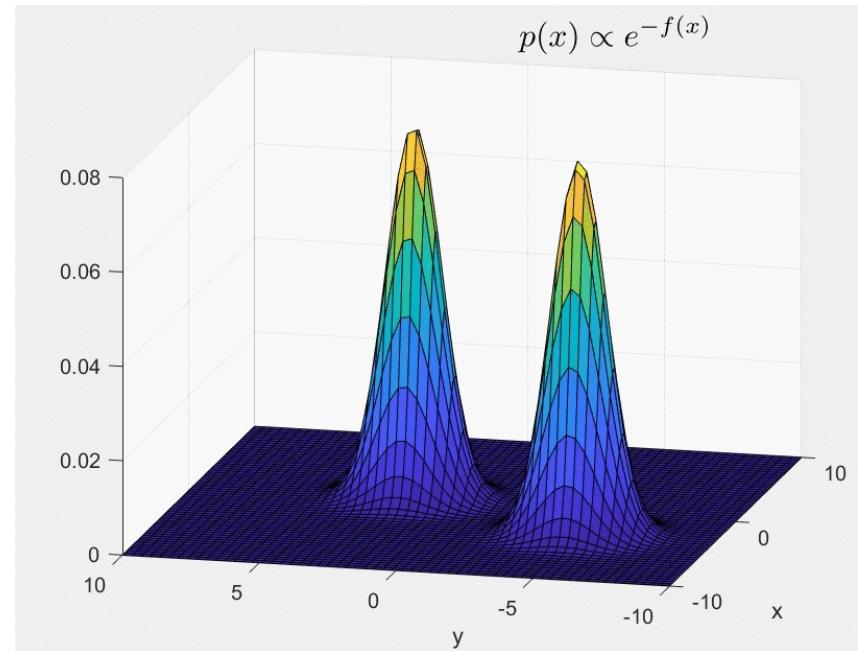
(Bovier '02,'04)  $\Rightarrow$   
time to get from A to B (through  
C) can be exponential!

# Potential solutions for multimodality

Unlike optimization,  
scale (temperature) matters!

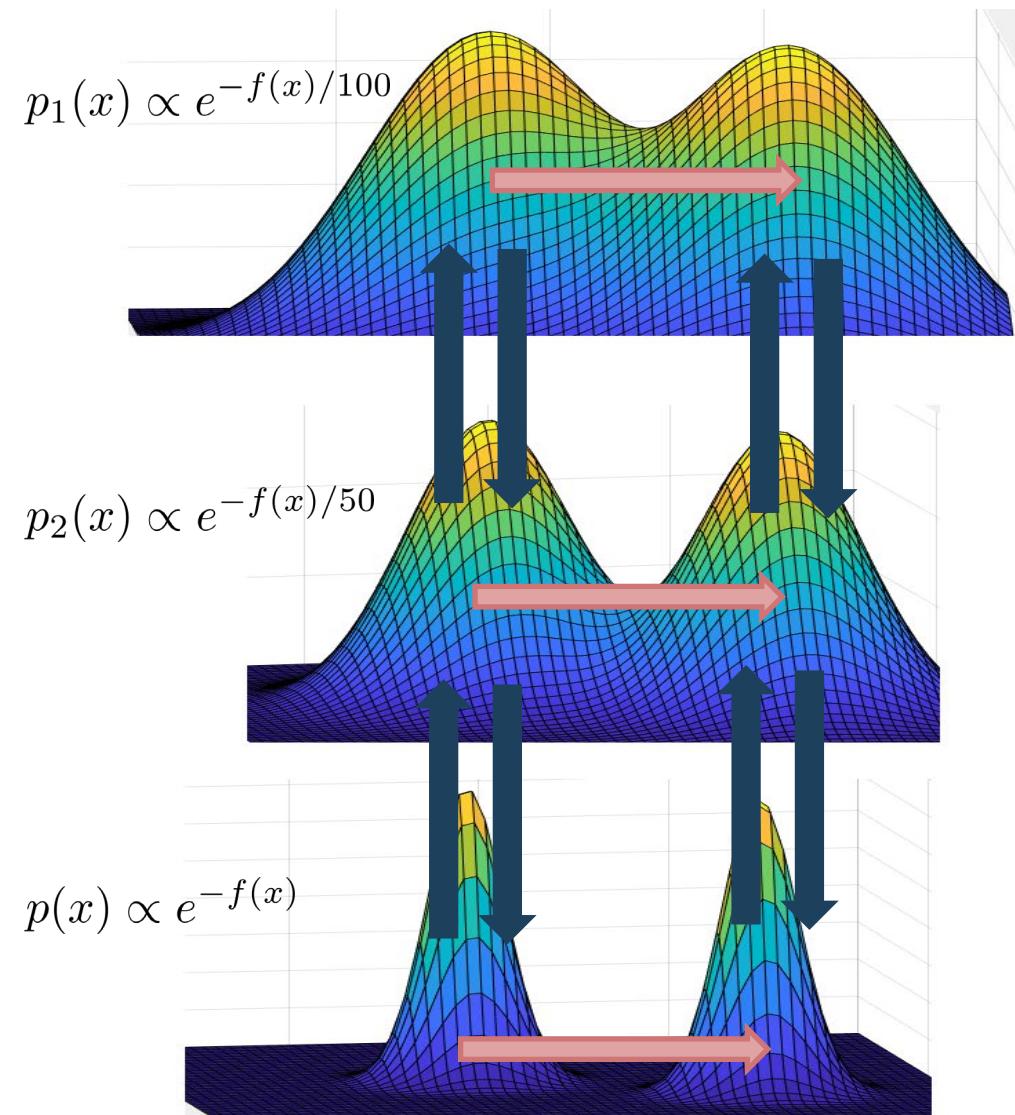
Sampling flatter distributions is easier!

Can we leverage this?



**Tempering/annealing:** run multiple chains at different temperatures, and use the fact that chains at higher temperatures move faster through landscape.

# Tempering: flatten the hills



**Algorithm:** run multiple walks in parallel for **different temperatures**.

**Swap** locations **occasionally** so lower-temp. chains explore space faster. (Occasionally = equilibrium distr. at each temperature is correct.)

Popular in practice, among other “annealing tricks” ((reverse) annealed importance sampling, tunneling...).

**Little theory...**

# The tempering chain

- Markov Chain on state space:  $\mathbb{R}^d \times [L]$ . ( $L$  is # of temperatures)
- Let  $M_k$  be the Markov Chain corresponding to temperature  $k$  (i.e. with stationary distr.  $p_k$ )

Tempering chain.

Run k-th chain

Let current point be  $(x, k)$ .

- With probability  $1/2$ :  
Swap points, perform  
Metropolis-Hastings to  
preserve stationary distr.  
Set next point to  $(x, k')$ .
- With probability  $1/2$ : pick  $k'$  uniformly, set next point to  $(x, k')$  with  
probability  $\min\left(\frac{p_{k'}(x)}{p_k(x)}, 1\right)$

The stationary distribution is  $P(x, t) = \frac{1}{L} p_k(x)$

# When does this work?

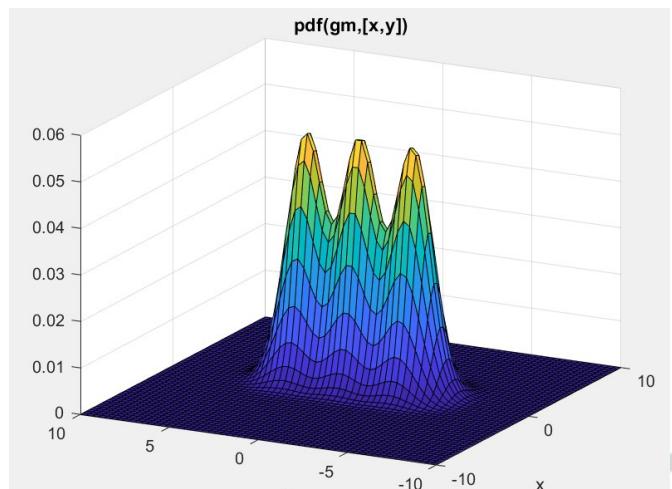
Each component a “mode”  
“Modes” have same shape

Thm [Ge-Lee-Risteski ‘18]:

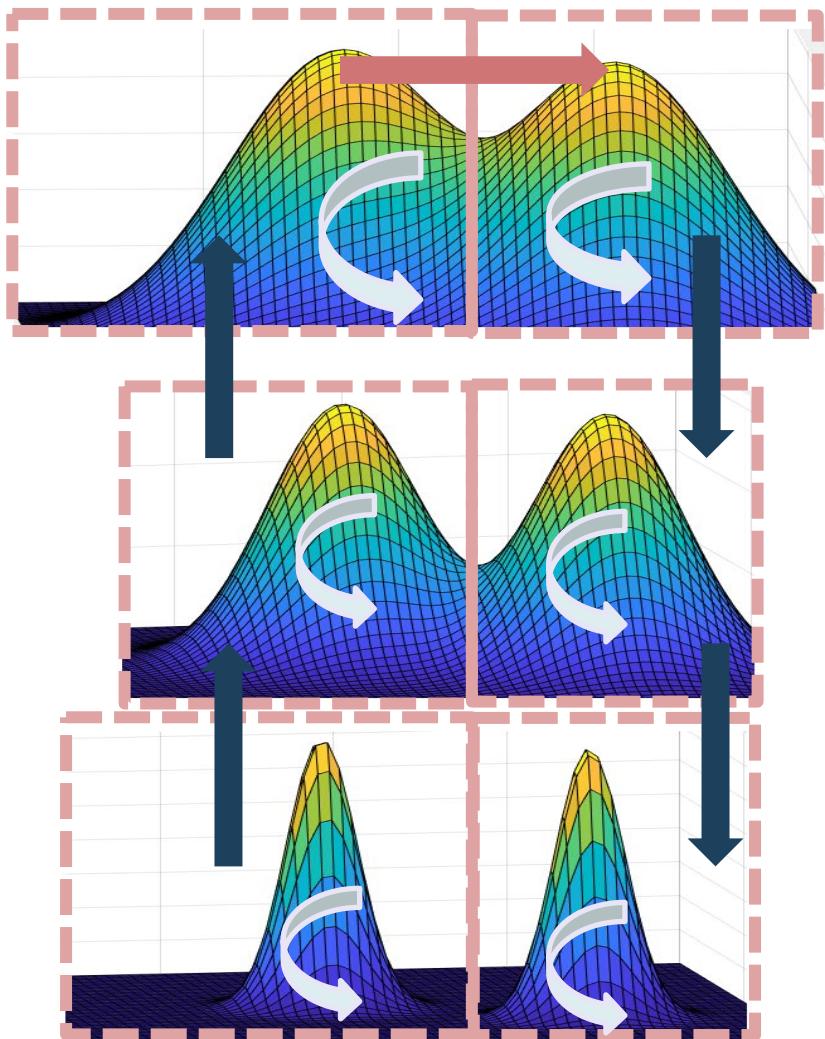
Let  $p(x) \propto e^{-f(x)}$  be a **mixture** of **n** shifts of a **d-dim. log-concave distribution**.

Then, **Langevin + simulated tempering** run for time **poly(n, d)** samples from distribution close to p

Result is **robust**: works if p is **close** to a mixture (w/ degradation in runtime)



# Why it works: take the road less hilly



Choose **highest temp.** s.t. walk converges fast. (Hills are flat)

Can partition space in **blocks** ( $\approx$  modes), s.t.

(1) Walk converges fast **inside each block**

(2) Blocks **aren't too small**

$\Rightarrow$  Fast convergence for tempering.

**Intuition:** Fast inside each mode.

Can get to highest temp. “parallel” mode.

Can get to any other mode at highest temp.

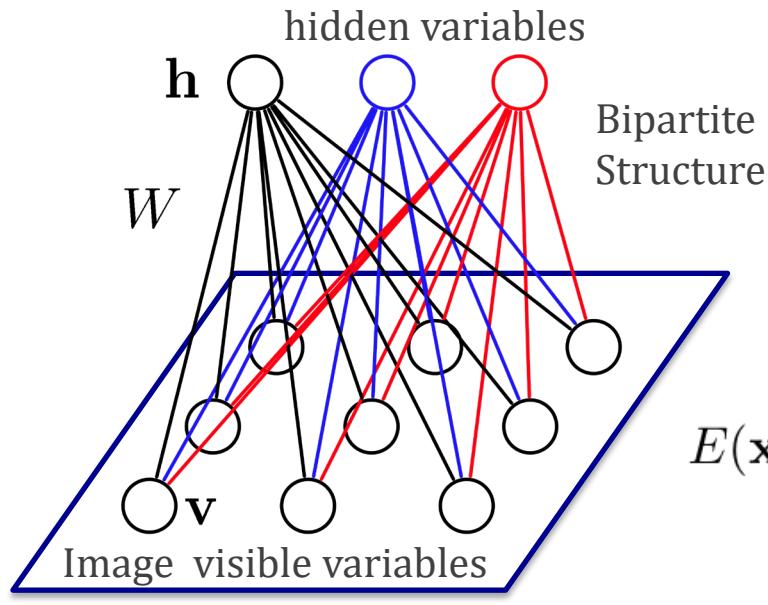
Can get to lowest temp. “parallel” mode.

# Learning unnormalized density models using MCMC

# Restricted Boltzmann Machines

An **undirected** latent-variable model

We denote visible and hidden variables with vectors  $\mathbf{v}, \mathbf{h}$  respectively:



Visible variables  $\mathbf{x} \in \{0, 1\}^D$   
are connected to hidden variables  $\mathbf{h} \in \{0, 1\}^F$

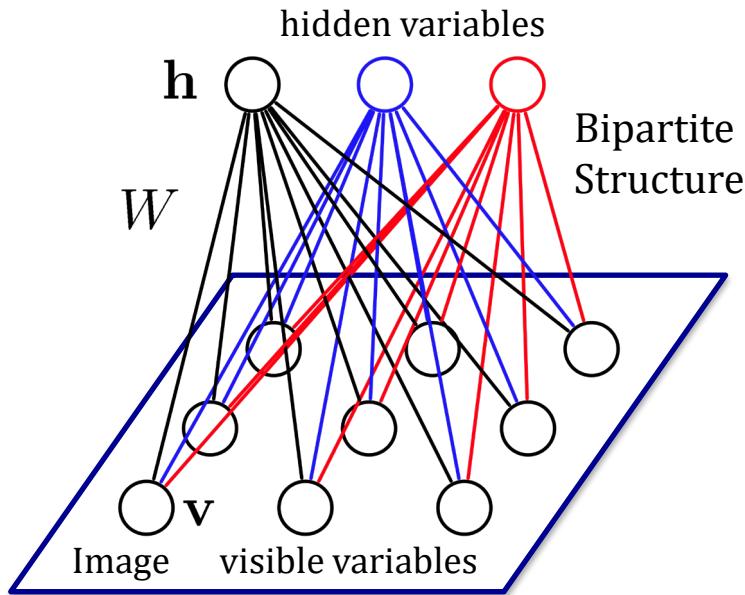
The energy of the joint configuration:

$$\begin{aligned} E(\mathbf{x}, \mathbf{h}) &= -\mathbf{h}^\top \mathbf{W} \mathbf{x} - \mathbf{c}^\top \mathbf{x} - \mathbf{b}^\top \mathbf{h} \\ &= -\sum_j \sum_k W_{j,k} h_j x_k - \sum_k c_k x_k - \sum_j b_j h_j \end{aligned}$$

Probability of the joint configuration:

$$p(\mathbf{x}, \mathbf{h}) = \exp(-E(\mathbf{x}, \mathbf{h}))/Z$$

# Restricted Boltzmann Machines



The **posterior** over the hidden variables is  
easy to sample from!  
(Conditional independence!)

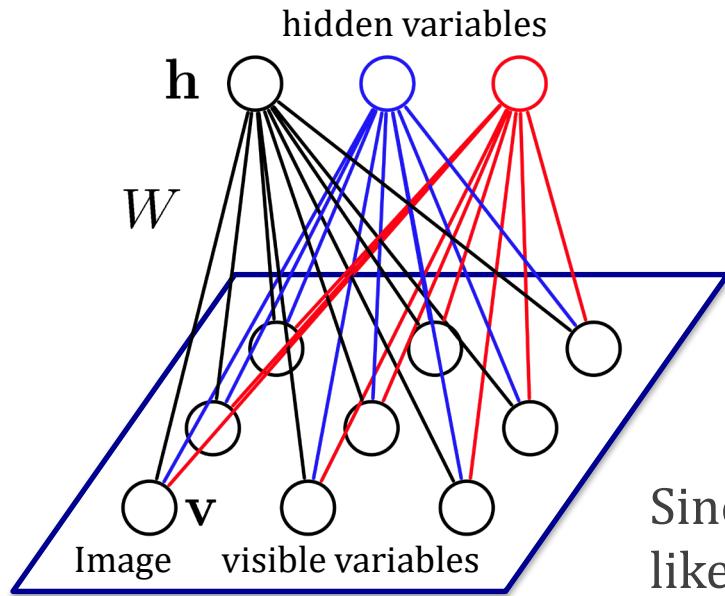
$$p(\mathbf{h}|\mathbf{x}) = \prod_j p(h_j|\mathbf{x}) \quad p(h_j = 1|\mathbf{x}) = \frac{1}{1 + \exp(-(b_j + \mathbf{W}_j \cdot \mathbf{x}))}$$

Factorizes

Similarly:

$$p(\mathbf{x}|\mathbf{h}) = \prod_k p(x_k|\mathbf{h}) \quad p(x_k = 1|\mathbf{h}) = \frac{1}{1 + \exp(-(c_k + \mathbf{h}^\top \mathbf{W}_{\cdot k}))}$$

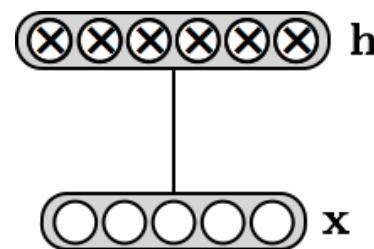
# How to learn RBM's



Given data  $x_1, x_2, \dots, x_n$ , solve

$$\max_{\theta \in \Theta} \sum_{i=1}^n \log p_\theta(x_i)$$

Since we have latent variables, we need to express the likelihood when we marginalize out the latents:



# How to learn RBM's

$$p(\mathbf{x}) = \sum_{\mathbf{h} \in \{0,1\}^H} \exp(\mathbf{h}^\top \mathbf{W} \mathbf{x} + \mathbf{c}^\top \mathbf{x} + \mathbf{b}^\top \mathbf{h}) / Z$$

# How to learn RBM's

$$\begin{aligned} p(\mathbf{x}) &= \sum_{\mathbf{h} \in \{0,1\}^H} \exp(\mathbf{h}^\top \mathbf{W} \mathbf{x} + \mathbf{c}^\top \mathbf{x} + \mathbf{b}^\top \mathbf{h}) / Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) \sum_{h_1 \in \{0,1\}} \cdots \sum_{h_H \in \{0,1\}} \exp \left( \sum_j h_j \mathbf{W}_{j \cdot} \mathbf{x} + b_j h_j \right) / Z \end{aligned}$$

# How to learn RBM's

$$\begin{aligned} p(\mathbf{x}) &= \sum_{\mathbf{h} \in \{0,1\}^H} \exp(\mathbf{h}^\top \mathbf{W}\mathbf{x} + \mathbf{c}^\top \mathbf{x} + \mathbf{b}^\top \mathbf{h}) / Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) \sum_{h_1 \in \{0,1\}} \dots \sum_{h_H \in \{0,1\}} \exp\left(\sum_j h_j \mathbf{W}_{j\cdot} \mathbf{x} + b_j h_j\right) / Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) \left( \sum_{h_1 \in \{0,1\}} \exp(h_1 \mathbf{W}_{1\cdot} \mathbf{x} + b_1 h_1) \right) \dots \left( \sum_{h_H \in \{0,1\}} \exp(h_H \mathbf{W}_{H\cdot} \mathbf{x} + b_H h_H) \right) / Z \end{aligned}$$

# How to learn RBM's

$$\begin{aligned} p(\mathbf{x}) &= \sum_{\mathbf{h} \in \{0,1\}^H} \exp(\mathbf{h}^\top \mathbf{W}\mathbf{x} + \mathbf{c}^\top \mathbf{x} + \mathbf{b}^\top \mathbf{h}) / Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) \sum_{h_1 \in \{0,1\}} \dots \sum_{h_H \in \{0,1\}} \exp\left(\sum_j h_j \mathbf{W}_{j\cdot} \mathbf{x} + b_j h_j\right) / Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) \left( \sum_{h_1 \in \{0,1\}} \exp(h_1 \mathbf{W}_{1\cdot} \mathbf{x} + b_1 h_1) \right) \dots \left( \sum_{h_H \in \{0,1\}} \exp(h_H \mathbf{W}_{H\cdot} \mathbf{x} + b_H h_H) \right) / Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) (1 + \exp(b_1 + \mathbf{W}_{1\cdot} \mathbf{x})) \dots (1 + \exp(b_H + \mathbf{W}_{H\cdot} \mathbf{x})) / Z \end{aligned}$$

# How to learn RBM's

$$\begin{aligned} p(\mathbf{x}) &= \sum_{\mathbf{h} \in \{0,1\}^H} \exp(\mathbf{h}^\top \mathbf{W}\mathbf{x} + \mathbf{c}^\top \mathbf{x} + \mathbf{b}^\top \mathbf{h})/Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) \sum_{h_1 \in \{0,1\}} \dots \sum_{h_H \in \{0,1\}} \exp\left(\sum_j h_j \mathbf{W}_{j\cdot} \mathbf{x} + b_j h_j\right)/Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) \left( \sum_{h_1 \in \{0,1\}} \exp(h_1 \mathbf{W}_{1\cdot} \mathbf{x} + b_1 h_1) \right) \dots \left( \sum_{h_H \in \{0,1\}} \exp(h_H \mathbf{W}_{H\cdot} \mathbf{x} + b_H h_H) \right)/Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) (1 + \exp(b_1 + \mathbf{W}_{1\cdot} \mathbf{x})) \dots (1 + \exp(b_H + \mathbf{W}_{H\cdot} \mathbf{x}))/Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) \exp(\log(1 + \exp(b_1 + \mathbf{W}_{1\cdot} \mathbf{x}))) \dots \exp(\log(1 + \exp(b_H + \mathbf{W}_{H\cdot} \mathbf{x}))) / Z \end{aligned}$$

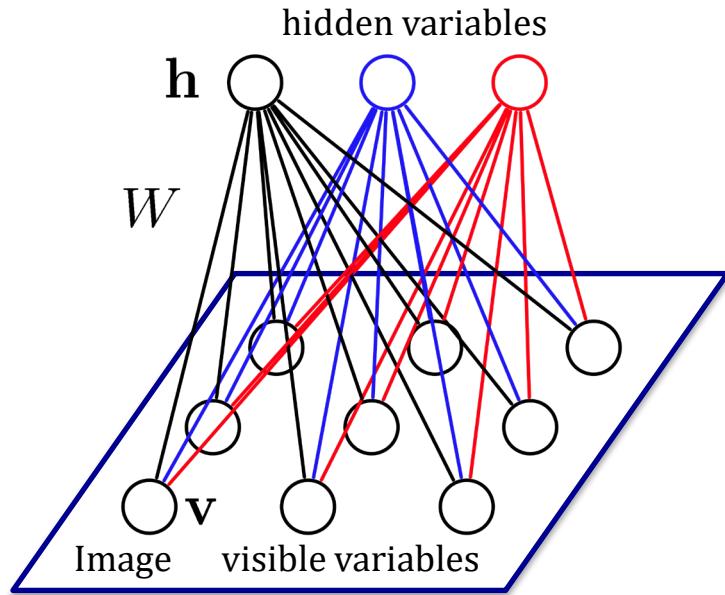
# How to learn RBM's

$$\begin{aligned} p(\mathbf{x}) &= \sum_{\mathbf{h} \in \{0,1\}^H} \exp(\mathbf{h}^\top \mathbf{W}\mathbf{x} + \mathbf{c}^\top \mathbf{x} + \mathbf{b}^\top \mathbf{h}) / Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) \sum_{h_1 \in \{0,1\}} \dots \sum_{h_H \in \{0,1\}} \exp\left(\sum_j h_j \mathbf{W}_{j\cdot} \mathbf{x} + b_j h_j\right) / Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) \left( \sum_{h_1 \in \{0,1\}} \exp(h_1 \mathbf{W}_{1\cdot} \mathbf{x} + b_1 h_1) \right) \dots \left( \sum_{h_H \in \{0,1\}} \exp(h_H \mathbf{W}_{H\cdot} \mathbf{x} + b_H h_H) \right) / Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) (1 + \exp(b_1 + \mathbf{W}_{1\cdot} \mathbf{x})) \dots (1 + \exp(b_H + \mathbf{W}_{H\cdot} \mathbf{x})) / Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) \exp(\log(1 + \exp(b_1 + \mathbf{W}_{1\cdot} \mathbf{x}))) \dots \exp(\log(1 + \exp(b_H + \mathbf{W}_{H\cdot} \mathbf{x}))) / Z \\ &= \exp\left(\mathbf{c}^\top \mathbf{x} + \underbrace{\sum_{j=1}^H \log(1 + \exp(b_j + \mathbf{W}_{j\cdot} \mathbf{x}))}_{= F(\mathbf{x})}\right) / Z \end{aligned}$$

# How to learn RBM's

$$\begin{aligned} p(\mathbf{x}) &= \sum_{\mathbf{h} \in \{0,1\}^H} \exp(\mathbf{h}^\top \mathbf{W}\mathbf{x} + \mathbf{c}^\top \mathbf{x} + \mathbf{b}^\top \mathbf{h}) / Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) \sum_{h_1 \in \{0,1\}} \dots \sum_{h_H \in \{0,1\}} \exp\left(\sum_j h_j \mathbf{W}_{j\cdot} \mathbf{x} + b_j h_j\right) / Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) \left( \sum_{h_1 \in \{0,1\}} \exp(h_1 \mathbf{W}_{1\cdot} \mathbf{x} + b_1 h_1) \right) \dots \left( \sum_{h_H \in \{0,1\}} \exp(h_H \mathbf{W}_{H\cdot} \mathbf{x} + b_H h_H) \right) / Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) (1 + \exp(b_1 + \mathbf{W}_{1\cdot} \mathbf{x})) \dots (1 + \exp(b_H + \mathbf{W}_{H\cdot} \mathbf{x})) / Z \\ &= \exp(\mathbf{c}^\top \mathbf{x}) \exp(\log(1 + \exp(b_1 + \mathbf{W}_{1\cdot} \mathbf{x}))) \dots \exp(\log(1 + \exp(b_H + \mathbf{W}_{H\cdot} \mathbf{x}))) / Z \\ &= \exp\left(\mathbf{c}^\top \mathbf{x} + \underbrace{\sum_{j=1}^H \log(1 + \exp(b_j + \mathbf{W}_{j\cdot} \mathbf{x}))}_{= F(\mathbf{x})}\right) / Z \\ &= \exp(F(\mathbf{x})) / Z \end{aligned}$$

# How to learn RBM's



Given data  $x_1, x_2, \dots, x_n$ , solve

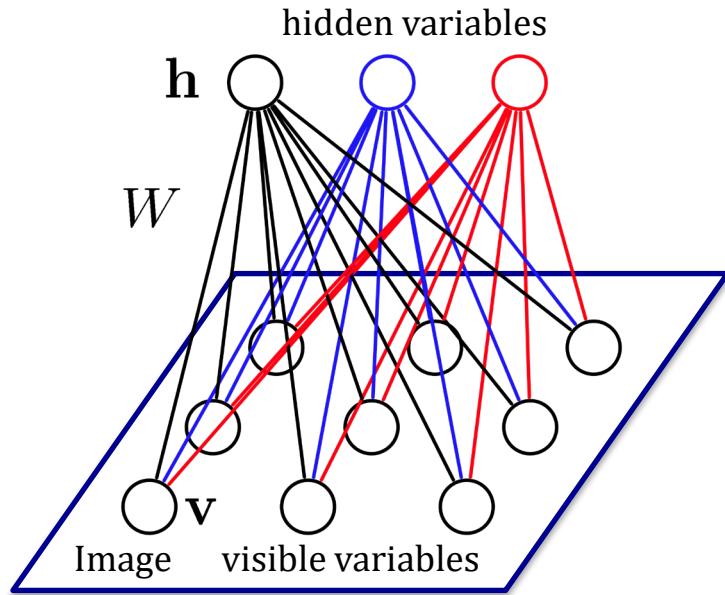
$$\max_{\theta \in \Theta} \sum_{i=1}^n \log p_\theta(x_i)$$

With this reduction, the undirected model calculations imply:

$$\nabla_\theta \left( \frac{1}{n} \sum_{i=1}^n \log p_\theta(x_i) \right) = \frac{1}{n} \left( \sum_i -\nabla_\theta F_\theta(x_i) \right) - \mathbb{E}_{p_\theta}[-\nabla_\theta F_\theta(x)]$$

$$\begin{aligned} \nabla_{W_{ij}} F_\theta(\mathbf{x}) &= \nabla_{W_{ij}} (\mathbf{c}^T \mathbf{x} + \sum_{j=1}^H \log(1 + \exp(b_j + \mathbf{W}_j \cdot \mathbf{x}))) = \frac{\exp(b_j + \mathbf{W}_j \cdot \mathbf{x})}{1 + \exp(b_j + \mathbf{W}_j \cdot \mathbf{x})} \mathbf{x}_i \\ &= \frac{1}{1 + \exp(-(b_j + \mathbf{W}_j \cdot \mathbf{x}))} \mathbf{x}_i = P(h_j = 1 | \mathbf{x}) \mathbf{x}_i \end{aligned}$$

# How to learn RBM's



Given data  $x_1, x_2, \dots, x_n$ , solve

$$\max_{\theta \in \Theta} \sum_{i=1}^n \log p_\theta(x_i)$$

With this reduction, the undirected model calculations imply:

$$\nabla_\theta \left( \frac{1}{n} \sum_{i=1}^n \log p_\theta(x_i) \right) = \frac{1}{n} \left( \sum_i -\nabla_\theta F_\theta(x_i) \right) - \mathbb{E}_{p_\theta}[-\nabla_\theta F_\theta(x)]$$

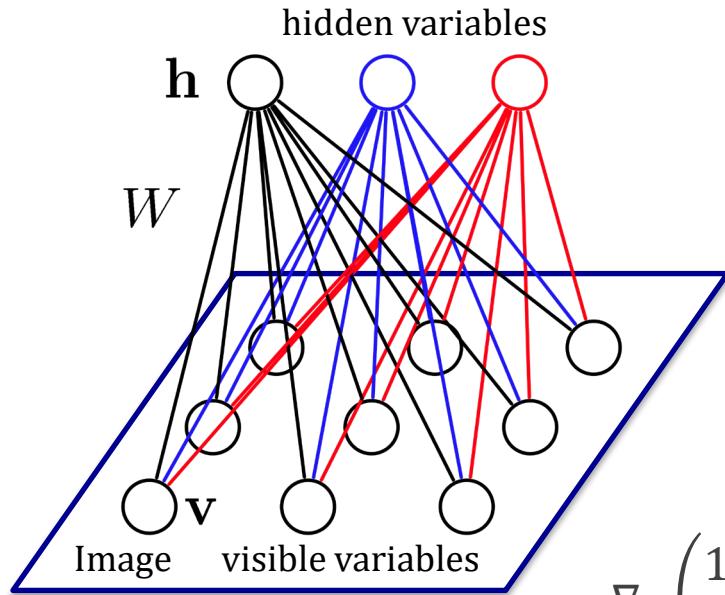
$$\nabla_{W_{ij}} F_\theta(\mathbf{x}) = P(h_j = 1 | \mathbf{x}) \mathbf{x}_i \Rightarrow \nabla_W F_\theta(\mathbf{x}) = \mathbf{h}(\mathbf{x}) \mathbf{x}^T$$

$$\nabla_b F_\theta(\mathbf{x}) = \mathbf{h}(\mathbf{x})$$

$$\nabla_c F_\theta(\mathbf{x}) = \mathbf{x}$$

$$\begin{aligned} \mathbf{h}(\mathbf{x}) &\stackrel{\text{def}}{=} \binom{p(h_1=1|\mathbf{x})}{p(h_H=1|\mathbf{x})} \\ &= \text{sigm}(\mathbf{b} + \mathbf{W}\mathbf{x}) \end{aligned}$$

# How to learn RBM's



Given data  $x_1, x_2, \dots, x_n$ , solve

$$\max_{\theta \in \Theta} \sum_{i=1}^n \log p_\theta(x_i)$$

$$\nabla_\theta \left( \frac{1}{n} \sum_{i=1}^n \log p_\theta(x_i) \right) = \frac{1}{n} \left( \sum_i -\nabla_\theta F_\theta(x_i) \right) - \mathbb{E}_{p_\theta} [-\nabla_\theta F_\theta(x)]$$

The hard term is again:  $\mathbb{E}_{p_\theta} [-\nabla_\theta F_\theta(x)]$  --- we need to draw samples from  $p_\theta$

We will draw samples using a Markov random walk: **Gibbs sampler!**

# Gibbs sampling

Consider sampling a distribution over  $n$  variables  $\mathbf{x} = (x_1, x_2, \dots, x_n)$ , s.t. each of the conditional distributions  $P(x_i | \mathbf{x}_{-i})$  is easy to sample. :

A common way to do this is using **Gibbs sampling**:

Repeat:

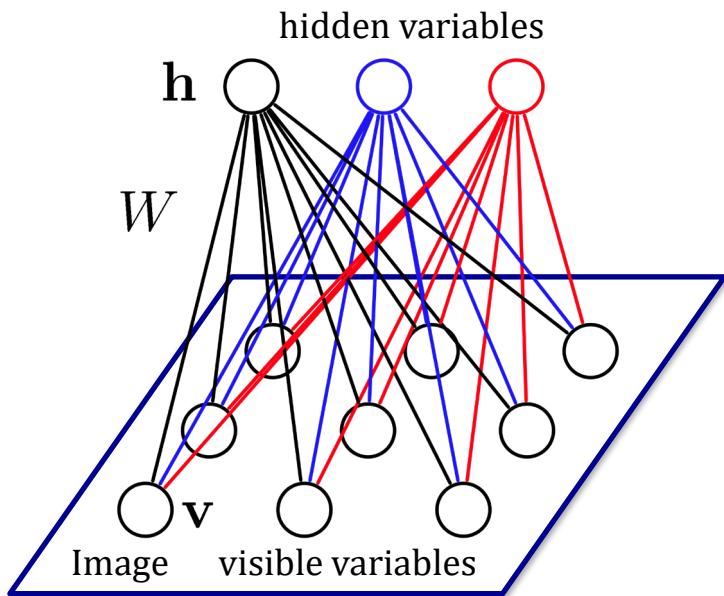
Let current state be  $\mathbf{x} = (x_1, x_2, \dots, x_n)$

Pick  $i \in [n]$  uniformly at random.

Sample  $x \sim P(X_i = x | \mathbf{x}_{-i})$

Update state to  $\mathbf{y} = (x_1, x_2, \dots, x_{i-1}, x, x_{i+1}, \dots, x_n)$

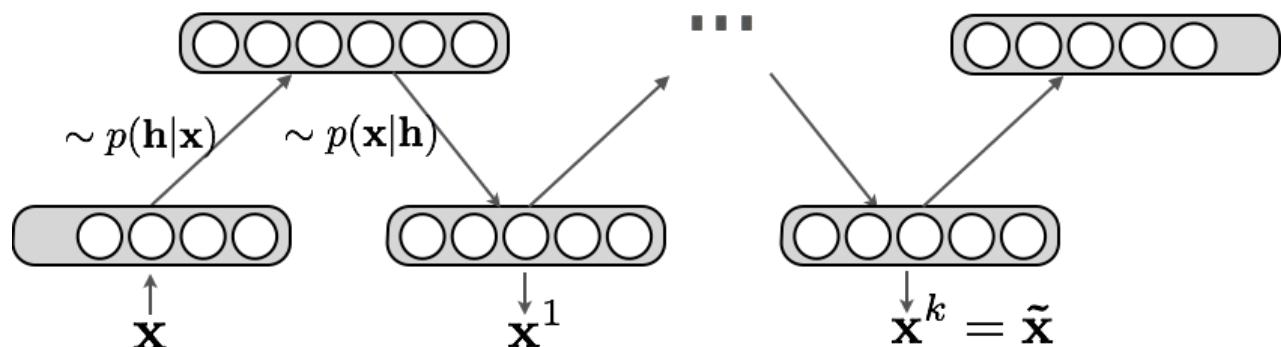
# Gibbs sampling for RBM's



Repeat:

Sample  $\mathbf{h} \sim P(\mathbf{h}|\mathbf{v})$   
Sample  $\mathbf{v} \sim P(\mathbf{v}|\mathbf{h})$

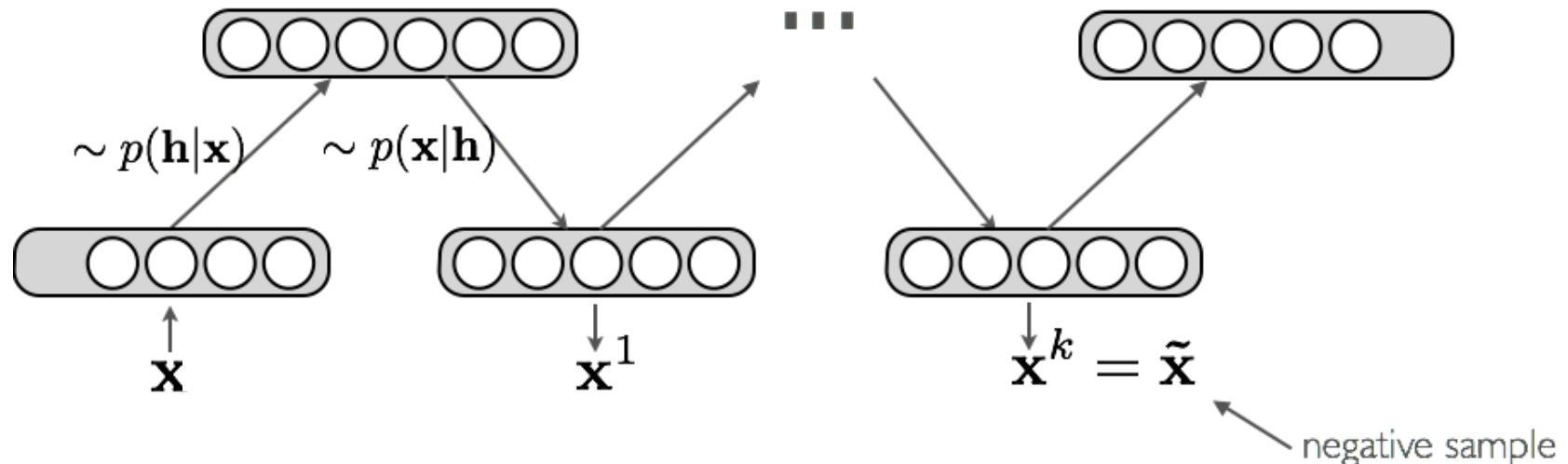
*Pictorially:*



# Contrastive Divergence

*Key idea behind Contrastive Divergence:*

- Replace the expectation by a point estimate at  $\tilde{\mathbf{x}}$
- Obtain the point  $\tilde{\mathbf{x}}$  by Gibbs sampling
- Start sampling chain at  $\mathbf{x}$



$k$  is often taken to be just 1.

# CD-k Algorithm

For each training example  $\mathbf{x}$

- Generate a negative sample  $\tilde{\mathbf{x}}$  using k steps of Gibbs sampling, starting at the data point  $\mathbf{x}$
- Update model parameters:

$$\left. \begin{aligned} \mathbf{W} &\leftarrow \mathbf{W} + \alpha \left( \mathbf{h}(\mathbf{x}^-) \mathbf{x}^{-\top} - \mathbf{h}(\tilde{\mathbf{x}}) \tilde{\mathbf{x}}^\top \right) \\ \mathbf{b} &\leftarrow \mathbf{b} + \alpha \left( \mathbf{h}(\mathbf{x}^-) - \mathbf{h}(\tilde{\mathbf{x}}) \right) \\ \mathbf{c} &\leftarrow \mathbf{c} + \alpha \left( \mathbf{x}^- - \tilde{\mathbf{x}} \right) \end{aligned} \right\}$$

*Gradients we derived before*

- Go back to 1 until stopping criteria

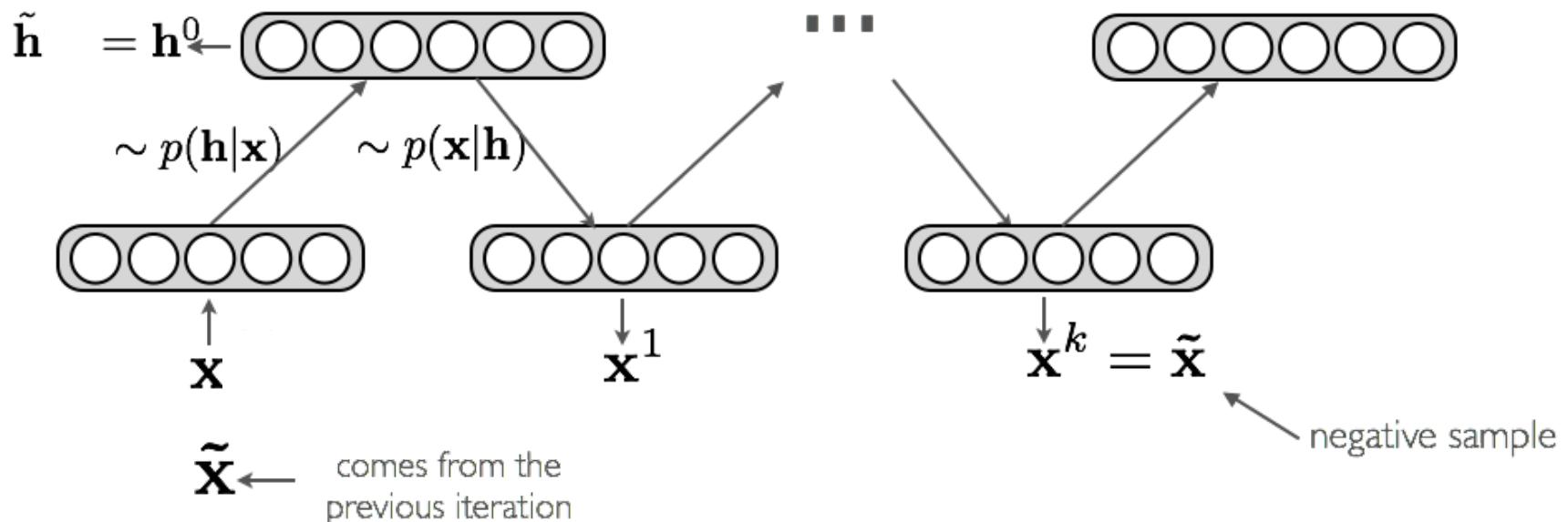
*Step size*

# CD-k Algorithm

- CD-k: contrastive divergence with k iterations of Gibbs sampling
- In general, the bigger k is, the less biased the estimate of the gradient will be
- In practice, k=1 works well for learning good features and for pre-training

# Persistent CD

*Idea:* instead of initializing the chain to  $\mathbf{x}^0$ , initialize the chain to the negative sample of the last iteration



# Tricks and Debugging

Unfortunately, it is not easy to debug training RBMs (e.g. using gradient checks)

We instead rely on approximate “tricks”

- we plot the average stochastic reconstruction  $\|\mathbf{x}^{(t)} - \tilde{\mathbf{x}}\|^2$  and see if it tends to decrease
- for inputs that correspond to image, we visualize the connection coming into each hidden unit as if it was an image
- gives an idea of the type of visual feature each hidden unit detects
- we can also try to approximate the partition function  $Z$  and see whether the (approximated) NLL decreases

(Salakhutdinov, Murray, ICML 2008)

# Learning continuous-space EBMs

Let's assume  $x \in \mathbb{R}^d$ . An obvious way to produce the estimates  $\mathbb{E}_{p_\theta}[-\nabla_\theta E_\theta(x)]$ : sample approximately from  $p_\theta$  using Langevin dynamics.

This was done recently: (Du, Mordatch '19), (Song, Ermon '19):

What's the difficulty?

*Data is multimodal*: Langevin might take long to mix.

# Learning continuous-space EBMs

Maintain buffer  
of previous  
samples to reduce  
mixing time

W/ some probability,  
start from random pt to  
encourage mode exploration

---

**Algorithm 1** Energy training algorithm

---

**Input:** data dist.  $p_D(\mathbf{x})$ , step size  $\lambda$ , number of steps  $K$   
 $\mathcal{B} \leftarrow \emptyset$

**while** not converged **do**

$\mathbf{x}_i^+ \sim p_D$

$\mathbf{x}_i^0 \sim \mathcal{B}$  with 95% probability and  $\mathcal{U}$  otherwise

▷ Generate sample from  $q_\theta$  via Langevin dynamics:

Langevin sampler {

**for** sample step  $k = 1$  to  $K$  **do**

$\tilde{\mathbf{x}}^k \leftarrow \tilde{\mathbf{x}}^{k-1} - \nabla_{\mathbf{x}} E_\theta(\tilde{\mathbf{x}}^{k-1}) + \omega, \quad \omega \sim \mathcal{N}(0, \sigma)$

**end for**

$\mathbf{x}_i^- = \Omega(\tilde{\mathbf{x}}_i^k)$

▷ Optimize objective  $\alpha \mathcal{L}_2 + \mathcal{L}_{ML}$  wrt  $\theta$ :

$\Delta\theta \leftarrow \nabla_\theta \frac{1}{N} \sum_i \alpha(E_\theta(\mathbf{x}_i^+)^2 + E_\theta(\mathbf{x}_i^-)^2) + E_\theta(\mathbf{x}_i^+) - E_\theta(\mathbf{x}_i^-)$

Update  $\theta$  based on  $\Delta\theta$  using Adam optimizer

$\mathcal{B} \leftarrow \mathcal{B} \cup \tilde{\mathbf{x}}_i$

**end while**

---

## The algorithm from (Du, Mordatch '19)



Figure 2: Conditional ImageNet32x32 EBM samples

A bit of  $l_2$   
regularization to  
ensure energy is  
somewhat smooth

# Learning continuous-space EBMs

The algorithm from (Song, Ermon '19)

Not likelihood based, instead minimize:

$$\mathbb{E}_{p_{data}} \|\nabla_x \log p_{data}(x) - (-E_\theta(x))\|^2$$

(Turns out to be writeable in a form friendly for taking gradients)

To alleviate multimodality, they use a variant of ***simulated tempering***, which we saw last time.

They also use ***smoothing by convolving*** with Gaussians to account for points with bad estimates of  $E_\theta(x)$ .

# Learning continuous-space EBMs

The algorithm from (Song, Ermon '19)

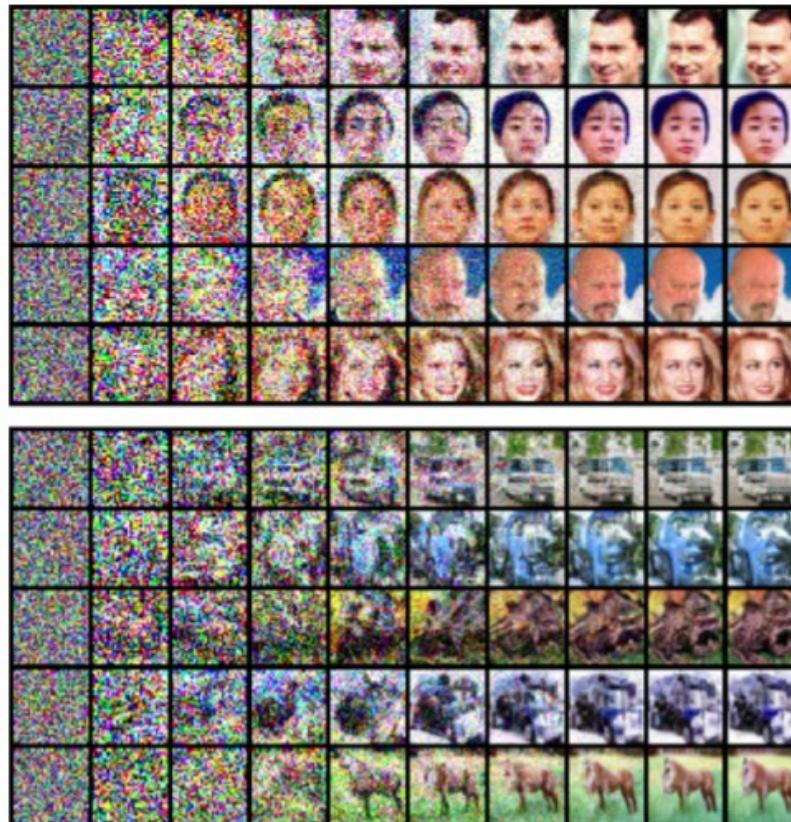


Figure 4: Intermediate samples of annealed Langevin dynamics.

# Learning continuous-space EBMs

The algorithm from (Song, Ermon '19)



(a) MNIST



(b) CelebA



(c) CIFAR-10

Figure 5: Uncurated samples on MNIST, CelebA, and CIFAR-10 datasets.