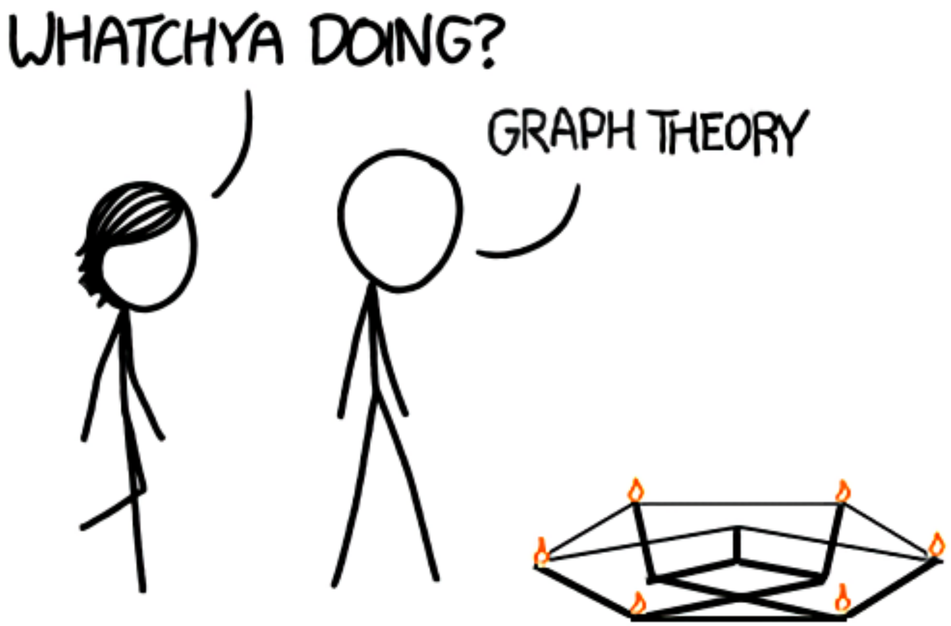
MACHINE LEARNING & DATA MINING



Probabilistic models

- knowing PDF = knowing "everything"
  - classification and regression
  - likelihood of a sample
  - conditional PDF (e.g. PDF for missing values)
  - sampling (unconditional or conditional)
  - you name it

Example: modelling a discrete PDF

- simplest way: tabulating all probabilities
- infeasible for large number of random variables
  - e.g. binary RGB images of 32x32 pixels
  - tabulating all probabilities would require $2^{3072} - 1 \approx (10^3)^{307} > 10^{900}$ numbers
  - compare this to ~10^{80} atoms in the observable Universe

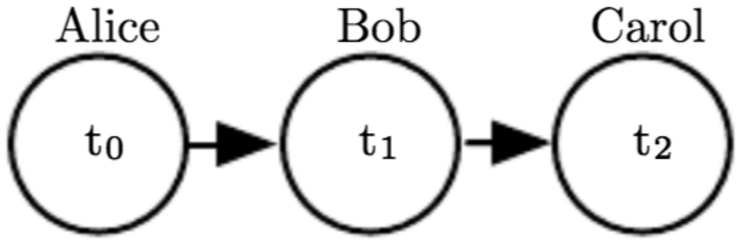
Motivation for structured modeling

- Table-based approach fails due to explicit modeling of interactions between all possible subsets of random variables
  - Real-life distributions are simpler!

- Most variables may be interacting only indirectly
- Example: relay race
  - team of 3 runners each running a single lap
  - the last runner's finishing time related to the first runner's time indirectly
    (only through the second runner's time)
  - is independent if conditioned on the second runner's time

Directed graphical model

(aka belief network or Bavesian network)



- Vertices — random variables
- Edges — relations between them
  - parent-child relationship meaning that one variable ‘influences’ another in a sense that the child’s PDF can be introduced as conditional wrt the parent, e.g.:

$$p(t_0, t_1, t_2) = p(t_0)p(t_1 \mid t_0)p(t_2 \mid t_1)$$

- Generally:

$$p(\mathbf{x}) = \Pi_i p(x_i \mid \mathrm{Pa}_G(x_i))$$

set of parents for x_i

- Acyclic graph

Structuring reduces # of parameters

- suppose we discretize each time variable into 100 intervals
- using the tabulating approach again
- full PDF: $100^3 - 1 = 999999$ parameters
- structured PDF: $(100 - 1) + 100 \cdot (100 - 1) + 100 \cdot (100 - 1) = 19899$ parameters

Sampling from directed graphical model

Ancestral sampling:

- put the variables in hierarchical order (parents before children)
- sample the variables in that order according to their individual PDFs
- ordering always possible for acyclic graphs
- ordering makes sure at each step all the needed conditionals already sampled

Note:

- conditional sampling straightforward for upstream conditions
- generally might be infeasible

Undirected model

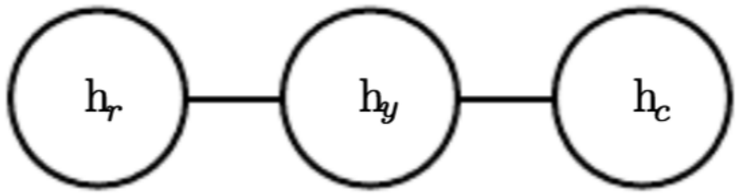
Markov random fields (MRF), Markov network

- Defined by an undirected graph
- Vertices — random variables
- Edges — relations between them
- Satisfy the properties:
  - any two non-adjacent variables are conditionally independent, given all other variables
  - any variable, given its neighbors, is conditionally independent of all other variables
  - any two subsets of variables are conditionally independent given a separating subset

S is a separating subset for subsets **A** and **B** when all paths from **A** to **B** pass through **S**

Example

- Modeling whether you, your roommate and your colleague are healthy or sick (e.g. caught a cold)
- Assuming your roommate and colleague never meet



- Whether your colleague is healthy is dependent of whether your roommate is healthy
- Though, independent given your health condition

PDF for undirected model

- Clique – a subset of vertices such that every two are adjacent
- A clique C in the graph associated with a (multiplicative) term in the PDF
- 'clique potential' $\phi(C)$
- Product of potentials form the unnormalized PDF:

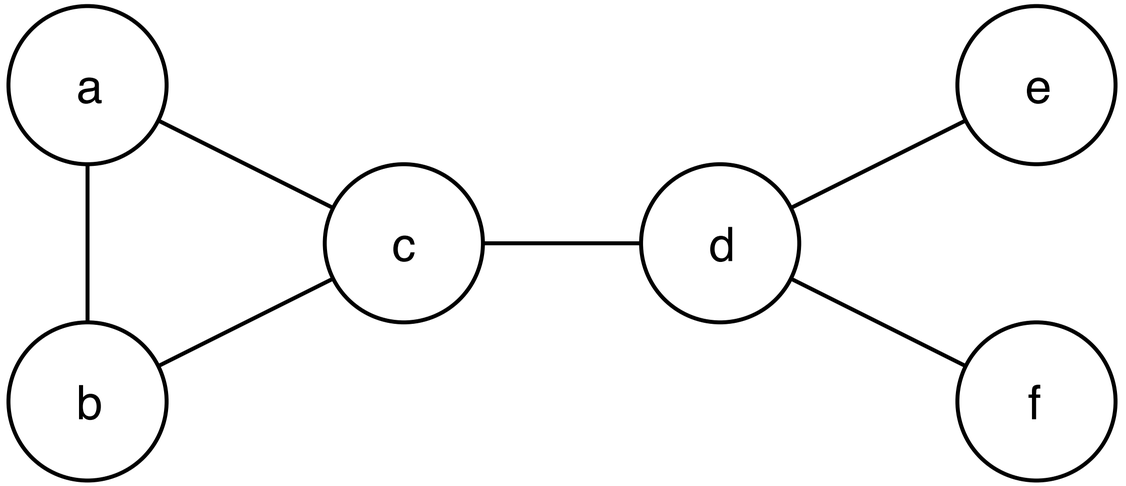
$$\tilde{p}(\mathbf{x}) = \Pi_{C \in G} \phi(C)$$

- The normalized PDF is:

$$p(\mathbf{x}) = \frac{1}{Z} \tilde{p}(\mathbf{x})$$

partition function
(typically intractable)

Example



$$\tilde{p}(a, b, c, d, e, f) = \phi_1(a, b, c) \cdot \phi_2(c, d) \cdot \phi_3(d, e) \cdot \phi_4(d, f)$$

Note

- no PDF is inherently ‘directed’ or ‘undirected’
- directed and undirected graphs are just ways of representing a PDF
- some ways are more suitable for some PDFs

Energy-based models

- Undirected models have nice theoretical properties when all outcomes are possible ($p(\mathbf{x}) > 0$ for all $\mathbf{x}$)
- A possible way to enforce this condition by design:

$$\tilde{p}(\mathbf{x}) = e^{-E(\mathbf{x})}$$

- Distribution of this form is called Boltzmann distribution
- $E(x)$ – energy function
- Due to $\exp(a + b) = \exp(a) \cdot \exp(b)$, each clique potential has an additive term in the energy function
- Partition function (normalizing coefficient) is intractable
  - cancels out in conditional distributions

Sampling from energy-based models

- Unlike directed models, sampling from undirected models is not straightforward
- Typically Markov Chain Monte Carlo (MCMC) methods are used

Markov Chain

- Markov Chain — a ‘chain’ of (possibly vector) random variables:

$$\left\{\mathbf{X}^{(0)}, \mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(k)}, \ldots\right\}$$

- such that:

$$p\left(\mathbf{X}^{(k)} \mid \left\{\mathbf{X}^{(i)} : i \in \mathbb{N}_0, i \neq k\right\}\right) = \underbrace{p\left(\mathbf{X}^{(k)} \mid \mathbf{X}^{(k-1)}\right)}_{\text{transition matrix}}$$

- If transition matrix does not depend on k, the chain is called *homogeneous*

Stationary distribution

- A transition between steps $k-1$ and k can be described as follows:

$$p\left(\mathbf{X}^{(k)}\right) = \sum_{\mathbf{X}^{(k-1)}} p\left(\mathbf{X}^{(k)} \mid \mathbf{X}^{(k-1)}\right) p\left(\mathbf{X}^{(k-1)}\right)$$

- If a distribution doesn't change upon transition, it is called stationary:

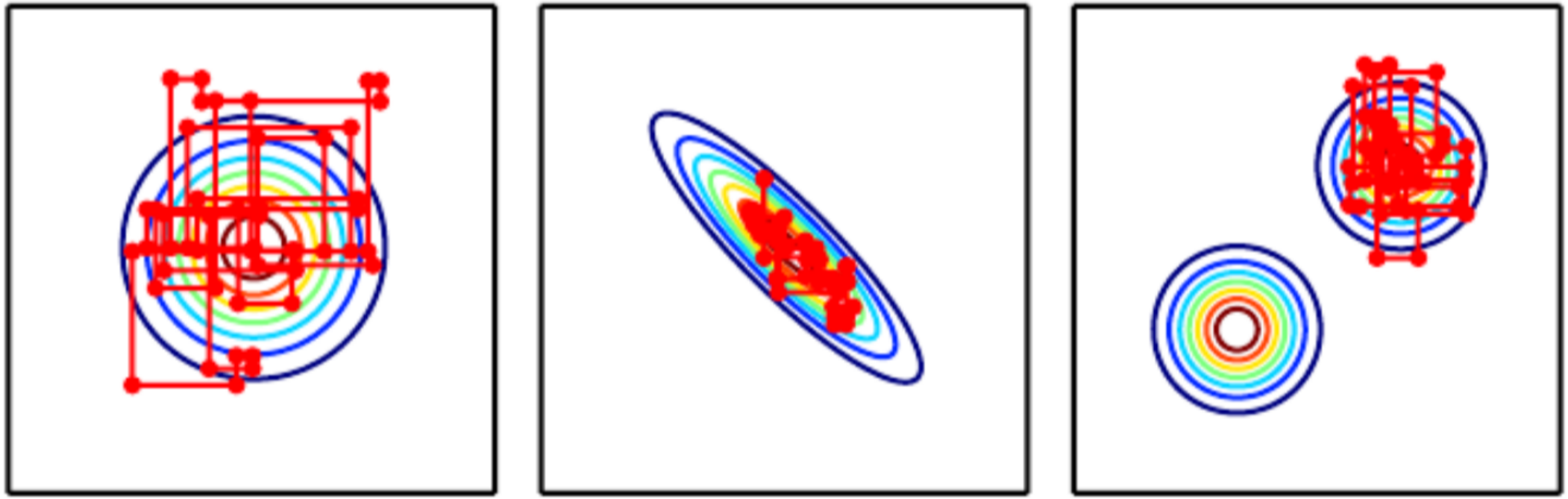
$$\pi\left(\mathbf{X}\right) = \sum_{\mathbf{X}'} p\left(\mathbf{X} \mid \mathbf{X}'\right) \pi\left(\mathbf{X}'\right)$$

- One can show that if a chain is aperiodic and irreducible (possible to get from any state to any state), it will **converge to a stationary distribution**

Markov Chain Monte Carlo

- In a nutshell, MCMC means building a Markov Chain with a desired stationary distribution and then running the chain to converge to it
- Gibbs sampling — a way of building a Markov Chain for energy-based models:
  - Initialize all the variables in the graph randomly
  - Iterate through the variables:
    - At each iteration update current variable based on its conditional PDF wrt its neighbors
    - Note: conditionally independent variables can be updated simultaneously (block Gibbs sampling)
  - Can be done in batches (running several chains simultaneously)
- Note: Gibbs sampling is guaranteed to converge to the PDF defined by the energy-based model, though it might take arbitrary many steps

Mode mixing challenge



- Getting through a high energy barrier may be problematic

Tempering to mix between modes

- Possible solution for the mode mixing challenge — introducing ‘temperature’ 1/β:

$$\tilde{p}(\mathbf{x}) = e^{-E(\mathbf{x})} \quad \rightarrow \quad \tilde{p}_\beta(\mathbf{x}) = e^{-\beta E(\mathbf{x})}$$

- Raise the temperature to make the mixing more likely
- Then gradually reduce the temperature

Training energy-based models

- Depending on whether the variables are discrete or continuous, the partition function is given by:

$$Z = \sum_{\mathbf{x}} \tilde{p}(\mathbf{x}) \qquad Z = \int \tilde{p}(\mathbf{x}) d\mathbf{x}$$

- If we were to maximize the log-likelihood with gradient ascent:

$$\nabla_\theta \log p(\mathbf{x}) = \nabla_\theta \log \tilde{p}(\mathbf{x}) - \nabla_\theta \log Z$$

Training energy-based models

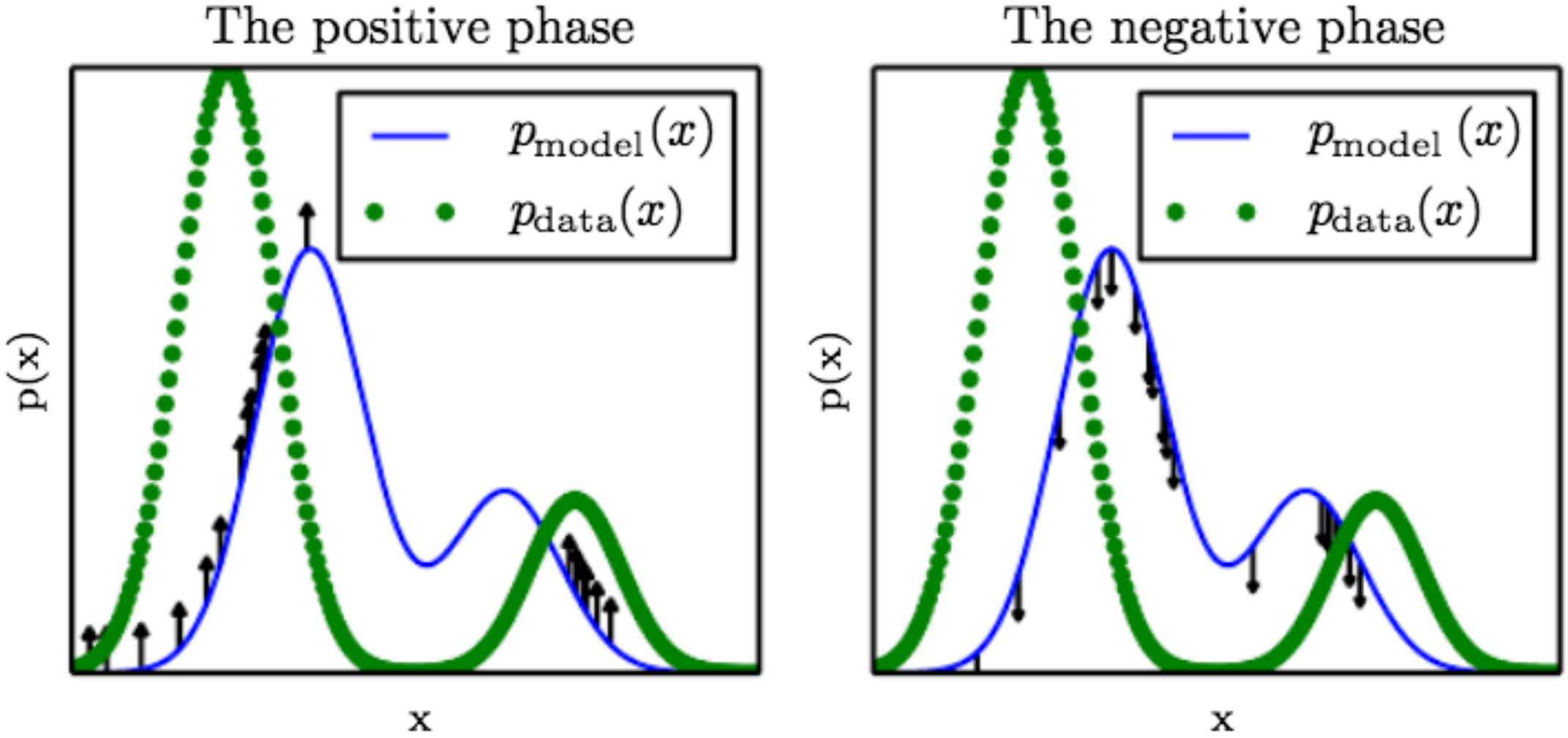
- Consider the gradient of the partition function:

$$\begin{aligned}
\nabla_\theta \log Z = \frac{\nabla_\theta Z}{Z} &= \frac{\nabla_\theta \sum_{\mathbf{x}} \tilde{p}(\mathbf{x})}{Z} \\
&= \frac{\sum_{\mathbf{x}} \nabla_\theta \tilde{p}(\mathbf{x})}{Z} \\
&\underline{=}_{\tilde{p}(\mathbf{x})>0} \frac{\sum_{\mathbf{x}} \tilde{p}(\mathbf{x}) \nabla_\theta \log \tilde{p}(\mathbf{x})}{Z} \\
&= \sum_{\mathbf{x}} p(\mathbf{x}) \nabla_\theta \log \tilde{p}(\mathbf{x}) \\
&= \mathop{\mathbb{E}}_{x \sim p(\mathbf{x})} \nabla_\theta \log \tilde{p}(\mathbf{x})
\end{aligned}$$

Training energy-based models

$$\nabla_\theta \log L = \underbrace{\mathop{\mathbb{E}}_{x \sim p^{\mathrm{data}}(\mathbf{x})} \nabla_\theta \log \tilde{p}(\mathbf{x})}_{\text{positive phase}} - \underbrace{\mathop{\mathbb{E}}_{x \sim p(\mathbf{x})} \nabla_\theta \log \tilde{p}(\mathbf{x})}_{\text{negative phase}}$$

Intuition


The positive phase
$p_{\text{model}}(x)$
$p_{\text{data}}(x)$
p(x)
x
The negative phase
$p_{\text{model}}(x)$
$p_{\text{data}}(x)$
p(x)
x

Naive algorithm

Algorithm 18.1 A naive MCMC algorithm for maximizing the log-likelihood with an intractable partition function using gradient ascent

Set ϵ, the step size, to a small positive number.
Set k, the number of Gibbs steps, high enough to allow burn in. Perhaps 100 to train an RBM on a small image patch.
while not converged **do**
  Sample a minibatch of m examples $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\}$ from the training set
  $\mathbf{g} \leftarrow \frac{1}{m} \sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\mathbf{x}^{(i)}; \boldsymbol{\theta})$.
  Initialize a set of m samples $\{\tilde{\mathbf{x}}^{(1)}, \ldots, \tilde{\mathbf{x}}^{(m)}\}$ to random values (e.g., from a uniform or normal distribution, or possibly a distribution with marginals matched to the model's marginals).
  for $i = 1$ to k **do**
    for $j = 1$ to m **do**
      $\tilde{\mathbf{x}}^{(j)} \leftarrow \text{gibbs\_update}(\tilde{\mathbf{x}}^{(j)})$.
    end for
  end for
  $\mathbf{g} \leftarrow \mathbf{g} - \frac{1}{m} \sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\tilde{\mathbf{x}}^{(i)}; \boldsymbol{\theta})$.
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \epsilon \mathbf{g}$.
end while

- computationally inefficient: each gradient step requires an MCMC to converge

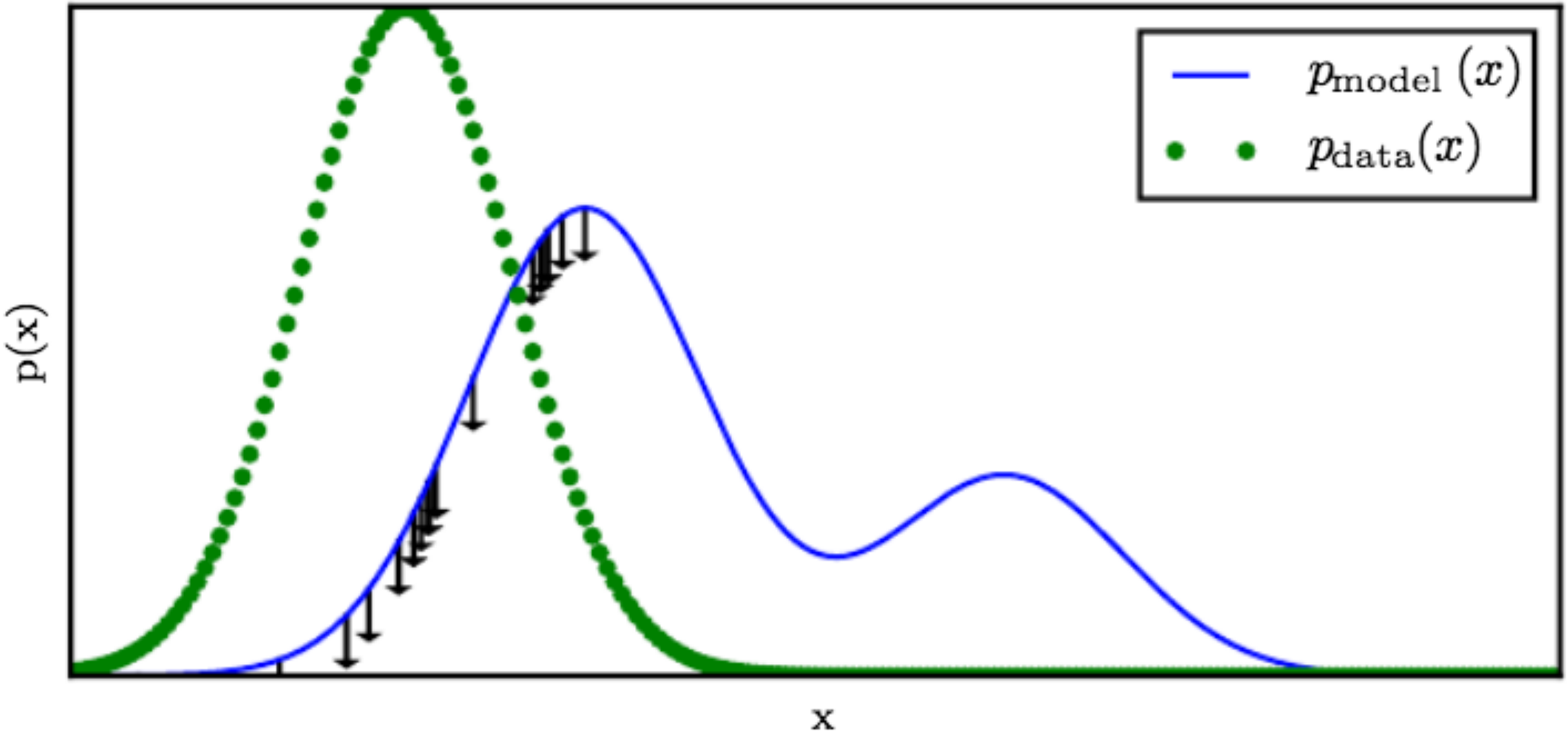
Contrastive divergence

Algorithm 18.2 The contrastive divergence algorithm, using gradient ascent as the optimization procedure

Set ϵ, the step size, to a small positive number.
Set k, the number of Gibbs steps, high enough to allow a Markov chain sampling from $p(\mathbf{x};\boldsymbol{\theta})$ to mix when initialized from p_{data}. Perhaps 1–20 to train an RBM on a small image patch.
while not converged **do**
  Sample a minibatch of m examples $\{\mathbf{x}^{(1)},\dots,\mathbf{x}^{(m)}\}$ from the training set
  $\mathbf{g} \leftarrow \frac{1}{m}\sum_{i=1}^{m}\nabla_{\boldsymbol{\theta}}\log\tilde{p}(\mathbf{x}^{(i)};\boldsymbol{\theta})$.
  for $i = 1$ to m **do**
    $\tilde{\mathbf{x}}^{(i)} \leftarrow \mathbf{x}^{(i)}$.
  end for
  for $i = 1$ to k **do**
    for $j = 1$ to m **do**
      $\tilde{\mathbf{x}}^{(j)} \leftarrow \text{gibbs\_update}(\tilde{\mathbf{x}}^{(j)})$.
    end for
  end for
  $\mathbf{g} \leftarrow \mathbf{g} - \frac{1}{m}\sum_{i=1}^{m}\nabla_{\boldsymbol{\theta}}\log\tilde{p}(\tilde{\mathbf{x}}^{(i)};\boldsymbol{\theta})$.
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \epsilon\mathbf{g}$.
end while

- Idea: MCMC will converge faster if initialized with real data distribution
- Problem: unrealistic model modes might never get sampled from (high energy barrier)

Spurious modes



Stochastic max likelihood / persistent contrastive divergence

Algorithm 18.3 The stochastic maximum likelihood / persistent contrastive divergence algorithm using gradient ascent as the optimization procedure

Set ϵ, the step size, to a small positive number.
Set k, the number of Gibbs steps, high enough to allow a Markov chain sampling from $p(\mathbf{x}; \boldsymbol{\theta} + \epsilon \mathbf{g})$ to burn in, starting from samples from $p(\mathbf{x}; \boldsymbol{\theta})$. Perhaps 1 for RBM on a small image patch, or 5–50 for a more complicated model like a DBM.

Initialize a set of m samples $\{\tilde{\mathbf{x}}^{(1)}, \ldots, \tilde{\mathbf{x}}^{(m)}\}$ to random values (e.g., from a uniform or normal distribution, or possibly a distribution with marginals matched to the model's marginals).
while not converged **do**
  Sample a minibatch of m examples $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\}$ from the training set
  $\mathbf{g} \leftarrow \frac{1}{m} \sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\mathbf{x}^{(i)}; \boldsymbol{\theta})$.
  for $i = 1$ to k **do**
    for $j = 1$ to m **do**
      $\tilde{\mathbf{x}}^{(j)} \leftarrow \text{gibbs\_update}(\tilde{\mathbf{x}}^{(j)})$.
    end for
  end for
  $\mathbf{g} \leftarrow \mathbf{g} - \frac{1}{m} \sum_{i=1}^{m} \nabla_{\boldsymbol{\theta}} \log \tilde{p}(\tilde{\mathbf{x}}^{(i)}; \boldsymbol{\theta})$.
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \epsilon \mathbf{g}$.
end while

- Idea: the model changes just slightly between the gradient update steps
- instead of initializing with real data, MCMC can be continued between gradient steps
- Indeed improves training
- Result might be biased
  - Can be solved by fine-tuning the model by resetting the chains

Boltzmann machines

- Energy based model
- Originally: with binary random variables
  - can be extended to other types
- Energy given by:

$$E(\mathbf{x}) = -\mathbf{x}^T\mathbf{U}\mathbf{x} - \mathbf{b}^T\mathbf{x}$$

- Typically enhanced by introducing visible (observed in data) and hidden variables v and h:

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{v}^T\mathbf{R}\mathbf{v} - \mathbf{v}^T\mathbf{W}\mathbf{h} - \mathbf{h}^T\mathbf{S}\mathbf{h} - \mathbf{b}^T\mathbf{v} - \mathbf{c}^T\mathbf{h}$$

- Adding hidden variables meaning we are learning a representation of data
- Analogous to MLP being a universal function approximator, BM becomes a universal approximator of probability mass functions

Training BMs

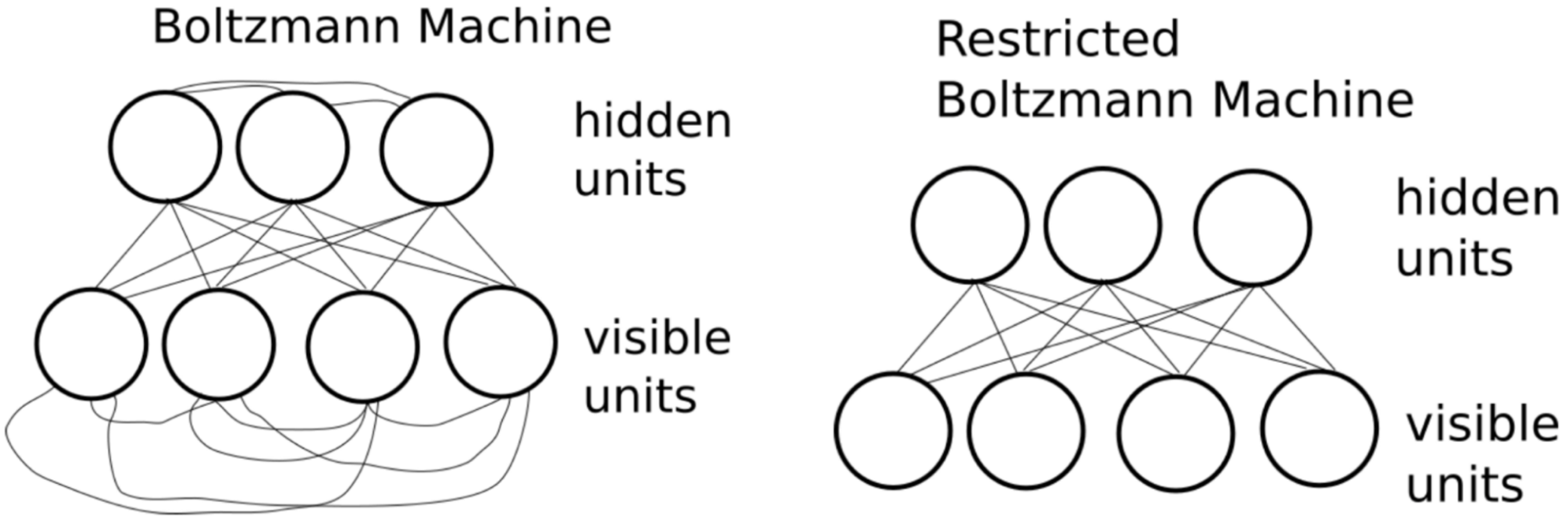
Positive phase:

- sample x from real data
- perform Gibbs sampling of h under x

Negative phase:

- init Gibbs chain with x
- sample both x and h from the model

Restricted Boltzmann Machines



Restricted Boltzmann Machines

- RBMs are BMs without intralayer connections:

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{v}^T \mathbf{W} \mathbf{h} - \mathbf{b}^T \mathbf{v} - \mathbf{c}^T \mathbf{h}$$

- This makes individual units within a layer conditionally independent of each other (given the units of the other layer)
- Gibbs sampling can be done in blocks (updating full layer at a time)

Conditional distributions

$$p(\mathbf{h} \mid \mathbf{v}) = \frac{p(\mathbf{h}, \mathbf{v})}{p(\mathbf{v})} = \frac{1}{p(\mathbf{v})} \frac{1}{Z} \exp\left\{\mathbf{b}^T \mathbf{v} + \mathbf{c}^T \mathbf{h} + \mathbf{v}^T \mathbf{W} \mathbf{h}\right\}$$

$$= \frac{1}{Z'} \exp\left\{\mathbf{c}^T \mathbf{h} + \mathbf{v}^T \mathbf{W} \mathbf{h}\right\}$$

$$= \frac{1}{Z'} \Pi_{j=1}^{n_h} \exp\left\{c_j h_j + (\mathbf{v}^T \mathbf{W})_j h_j\right\}$$

Conditional distributions

$$p(h_j = 1 \,|\, \mathbf{v}) = \frac{\tilde{p}(h_j = 1 \,|\, \mathbf{v})}{\tilde{p}(h_j = 0 \,|\, \mathbf{v}) + \tilde{p}(h_j = 1 \,|\, \mathbf{v})}$$

$$= \frac{\exp\left\{c_j + (\mathbf{v}^T \mathbf{W})_j\right\}}{\exp\{0\} + \exp\left\{c_j + (\mathbf{v}^T \mathbf{W})_j\right\}}$$

$$= \sigma\left(c_j + (\mathbf{v}^T \mathbf{W})_j\right)$$

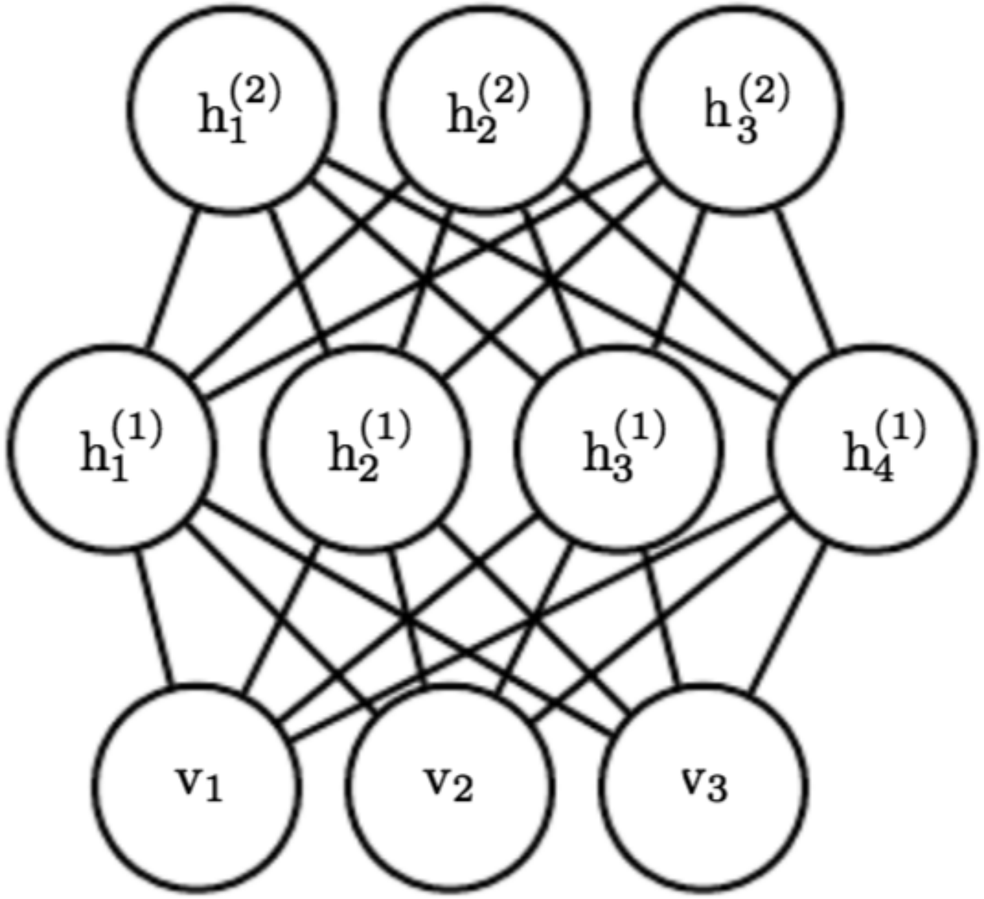
- And similarly for the visible units

Deep belief networks

- Mixture of directed and undirected graphs
- Trained greedily:
  - Training an RBM first
  - Using it's hidden units as 'visible' for a new RBM
  - Repeat stacking RBMs
- After the training all but the deepest two layers replaced with directed conditional distributions

- Historically, this model led to first successful training of MLP (with weights initialized from DBN)

Deep Boltzmann Machines



- ~ RBM with more layers
- Even for many layers, single Gibbs sampling step requires only two updates: updating odd layers, then updating even layers

Deep Boltzmann Machines

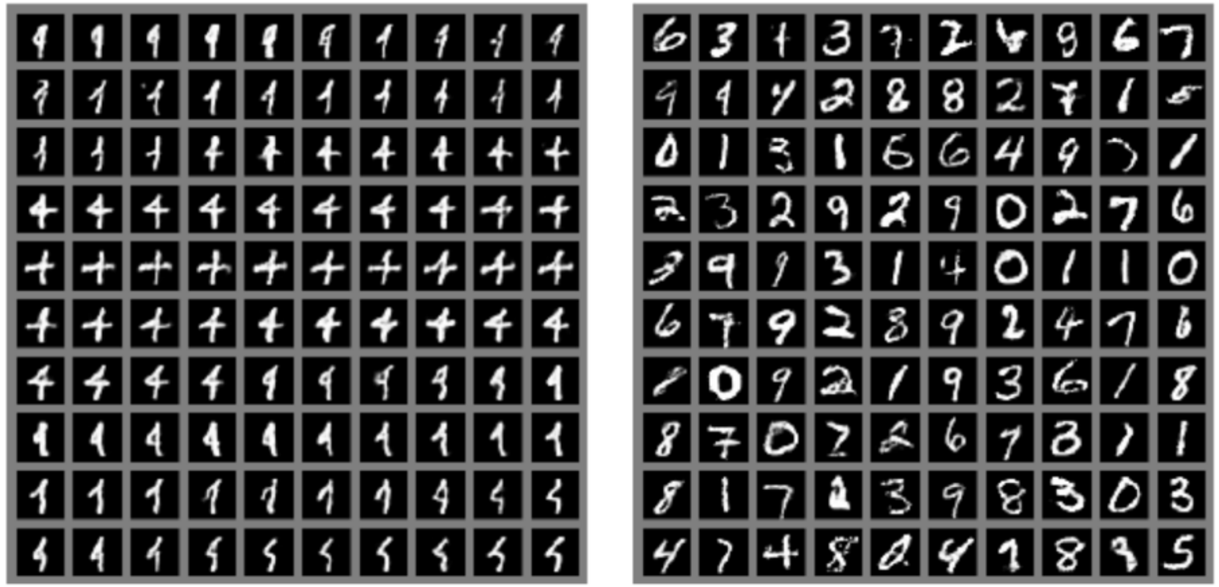


Figure 17.2: An illustration of the slow mixing problem in deep probabilistic models. Each panel should be read left to right, top to bottom. *(Left)*Consecutive samples from Gibbs sampling applied to a deep Boltzmann machine trained on the MNIST dataset. Consecutive samples are similar to each other. Because the Gibbs sampling is performed in a deep graphical model, this similarity is based more on semantic than raw visual features, but it is still difficult for the Gibbs chain to transition from one mode of the distribution to another, for example, by changing the digit identity. *(Right)*Consecutive ancestral samples from a generative adversarial network. Because ancestral sampling generates each sample independently from the others, there is no mixing problem.

Summary

- Structured models — a way of imposing our knowledge about the data to efficiently parametrize the problem
- Boltzmann machines — generative models, probabilistic representation learning
- Restricted Boltzmann Machines can be trained more efficiently compared to BM
  - they are a building block for Deep Belief Networks

Literature

I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016, www.deeplearningbook.org

A. Fischer, C. Igel, An Introduction to Restricted Boltzmann Machines, Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications: 17th Iberoamerican Congress, CIARP 2012, Buenos Aires, Argentina, September 3-6, 2012. Proceedings (pp.14-36)