COMP 790-125, HW3

Calvin Deutschbein

November 21, 2015

Deadline: 11/20/2014 11:59PM EST
Submit `hw3.pdf` by e-mail, `mailto:vjojic+comp790+hw3@cs.unc.edu`.

We will get some practice with using Gibbs sampling to perform inference in a Markov Random Field.

Notation and terminology We will work with models on n random variables $X_1, \ldots, X_n$. We will refer to values of random variables as their state, hence we will say random variable X_i is in state x_i. We will refer to tuples of values, for example $\mathbf{x} = (x_1, \ldots, x_n)$, as a **configuration**. Tuples of values associated with a subset of variables will be called a **partial configuration**.

Given a density on n random variables $X_1, \ldots, X_n$

$$p(X_1 = x_1, \ldots, X_n = x_n).$$

For compactness will use $p(\mathbf{x})$ and $p(x_1, \ldots, x_n)$ to denote probability of a configuration

$$p(X_1 = x_1, \ldots, X_n = x_n).$$

We will use $\mathbf{x}_{[-i]}$ to denote a partial configuration $x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n$. Probability of partial configuration $\mathbf{x}_{[-i]}$ can be obtained by marginalization

$$p(\mathbf{x}_{[-i]}) = \sum_{x_i} p(X_1 = x_1, \ldots, X_n = x_n).$$

Boltzmann distribution Given n random variables $X_1, \ldots, X_n$, a finite energy function

$$E(x_1, \ldots, x_n)$$

and a constant T (temperature), we obtain Boltzmann distribution as

$$p(X_1 = x_1, \ldots, X_n = x_n) = \frac{1}{Z} \exp\left\{-\frac{1}{T} E(x_1, \ldots, x_n)\right\}.$$

where

$$Z = \sum_{x_1} \cdots \sum_{x_n} p(X_1 = x_1, \ldots, X_n = x_n).$$

Pairwise Markov Random Field We will assume that we are given an undirected graph with n nodes and an edge set. We will assume a pairwise energy function θ, and singleton function ψ and obtain full configuration energy

$$E(x_1, \ldots, x_n) = \sum_{(i,j) \in S} \theta(x_i, x_j) + \sum_i \psi_i(x_i).$$

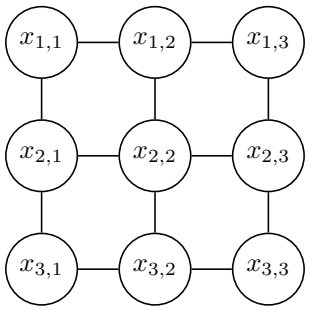
We can then obtain Boltzmann distribution for specific temperature

$$p(X_1 = x_1, \ldots, X_n = x_n) \propto \exp\left\{ -\frac{1}{T} \left[\sum_{(i,j) \in S} \theta(x_i, x_j) + \sum_i \psi_i(x_i) \right] \right\}$$

Note that in our formulation we are using the *same* θ for all edges, but potentially different ψ for each node.

We can interpret the energy function's relationship to probability of a configuration. The higher the energy of a configuration, the lower the probability.

Grid MRF Here is an example of a grid MRF:



We will index nodes by their row and column, $(2, 3)$ is the node in the second row and third column. When enumerating edges you can iterate each row and column, and consider edges $(i, j) - (i - 1, j)$ and $(i, j) - (i, j - 1)$. Hence energy function will look like

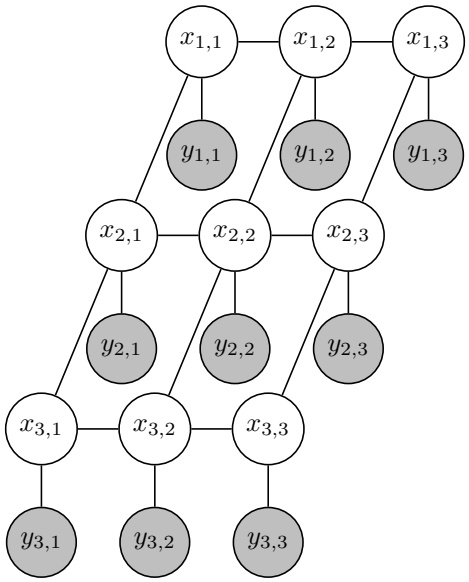
$$E(\mathbf{x}) = \sum_{(i,j)} \theta(x_{i,j}, x_{i-1,j}) + \theta(x_{i,j}, x_{i,j-1}) + \psi_{i,j}(x_{i,j}).$$

Here is a piece of code that can help you compute energy of a configuration in a pairwise MRF

```
function E = energy(theta,psi,T,x)
E = 0;
height = size(x,1);
width = size(x,2);
for i=1:height
    for j=1:width
```

```
        % unary energy terms are specific to (i,j)
        E = E + psi(x(i,j),i,j);
        if i>1
           E = E + theta(x(i,j),x(i-1,j));
        end
        if j>1
           E = E + theta(x(i,j),x(i,j-1));
        end
    end
end
```

Grid MRF with observations



We will add observations attached to each node $x_{i,j}$. These are named $y_{i,j}$. Above graph has a y of size 3×3.

Joint probability of $\mathbf{x}, \mathbf{y}$ We have specified only probability of $\mathbf{x}$, but we will now show that we can easily incorporate the probability of $\mathbf{y}$ into that formulation.

$$p(\mathbf{x}) \propto \exp\left\{ -\frac{1}{T} \sum_{i,j} \theta(x_{i,j}, x_{i-1,j}) + \theta(x_{i,j}, x_{i,j-1}) + \psi_{i,j}(x_{i,j}) \right\}$$

and

$$p(\mathbf{y}|\mathbf{x}) = \prod_i p(y_{i,j}|x_{i,j})$$

Figure 1: A very noisy image of the old well.

Joint probability is then

$$\begin{aligned} p(\mathbf{y},\mathbf{x}) &\propto \exp\left\{-\frac{1}{T}\left(\sum_{i,j}\theta(x_{i,j},x_{i-1,j})+\theta(x_{i,j},x_{i,j-1})+\sum_{i,j}\psi_{i,j}(x_{i,j})-\log p(y_{i,j}|x_{i,j})\right)\right\} \\ &= \exp\left\{-\frac{1}{T}\left(\sum_{i,j}\theta(x_{i,j},x_{i-1,j})+\theta(x_{i,j},x_{i,j-1})+\sum_{i,j}\xi_{i,j}(x_{i,j})\right)\right\} \end{aligned}$$

where

$$\xi_{i,j}(x_{i,j}) = \psi_{i,j}(x_{i,j}) - \log p(y_{i,j}|x_{i,j}).$$

So, even though we changed the name of the unary energy from ψ to ξ there is no real change in the energy computation. Hence, we can absorb the conditional probabilities $\log p(y_{i,j}|x_{i,j})$ into ψ. Thus energy that corresponds to edge between $x_{i,j}$ and $y_{i,j}$ is incorporated in unary energy ψ. Note the sign flip due to the fact that negative energy is exponentiated in Boltzmann distribution.

Denoising setup One simple application of pairwise MRFs is in denoising images. We can think of $\mathbf{y}$ as a noisy image and $\mathbf{x}$ a clean version of it. By sampling

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

we can obtain configurations from the distribution that puts substantial probability on the clean, and nearly clean versions of the image.

Problem 1(2pt)

Specifying θ and ψ We will specify a particular θ and ψ

$$\theta(a,b) = (a-b)^2$$

and

$$\psi(a, i, j) = (a - y_{i,j})^2.$$

Look at the grid MRF above and for $x_{2,2}$ write out all energy terms that involve it: The energy term is given by the following:

$$\{(x_{i,j}-x_{i-1,j})^2+(x_{i,j}-x_{i,j-1})^2+(x_{i,j}-y_{i,j})^2 : i, j \in \{(1,2),(2,1),(2,2),(2,3),(3,2)\}\}$$

There should be 5 of them, since there it has five neighbors.

Interpret what those terms are trying to achieve. You can think about what states of $x_{2,2}$ leads to the smallest energy for different states of its neighbors.

You can use the notion that $x_{2,2}$ is a clean image pixel and $y_{2,2}$ is noisy image pixel to explain what is going on.

The smallest energy state is the state in which there is a least squares difference across all edges.

Problem 2(2pt) Gibbs sampling iterates updates of single variable at a time, by drawing new states from conditional probability

$$p(\mathbf{x}_i|\mathbf{x}_{[-i]}).$$

In our case, we need to compute conditional probability

$$p(x_{i,j}|\mathbf{x}_{[-(i,j)]}, \mathbf{y})$$

First write out the joint probability of $\mathbf{x}, \mathbf{y}$ in terms of θ and ξ. Note that ξ is assumed to have absorbed terms that involve $\mathbf{y}$.

$$p(\mathbf{x}, \mathbf{y}) = \frac{1}{Z} \exp\left\{-\frac{1}{T}\left(\sum_{i,j} \theta(x_{i,j}, x_{i-1,j}) + \theta(x_{i,j}, x_{i,j-1}) + \sum_{i,j} \xi_{i,j}(x_{i,j})\right)\right\}$$

Next, identify the terms that depend on $x_{i,j}$ and collect them here

$$g(x_{i,j}, ...) = \frac{1}{Z} \exp\left\{-\frac{1}{T}\left(\xi_{i,j}(x_{i,j}) + \sum_{S} \theta(x_{i,j}, x_{i-1,j}) + \theta(x_{i,j}, x_{i,j-1})\right)\right\}$$

where

$$S = (i-1, j), (i, j-1), (i, j), (i, j+1), (i+1, j)$$

Previous problem should give you guidance on this.

Let the terms that do not involve $x_{i,j}$ be collected in some function $f(\mathbf{x}_{[-(i,j)]})$ – you do not need to write it out – then we can say that

$$E(\mathbf{x}) = g(x_{i,j}, ...) + f(\mathbf{x}_{[-(i,j)]}).$$

Use Bayes rule to obtain

$$p(x_{i,j}|\mathbf{x}_{[-(i,j)]}, \mathbf{y}) =$$

$$\frac{\frac{1}{Z}\exp\left\{-\frac{1}{T}\xi_{i,j}(x_{i,j})+\sum_S\theta(x_{i,j},x_{i-1,j})+\theta(x_{i,j},x_{i,j-1})+\sum_{i,j}\theta(x_{i,j},x_{i-1,j})+\theta(x_{i,j},x_{i,j-1})+\sum_{i,j}\xi_{i,j}(x_{i,j}\right\}}{\frac{1}{Z}\exp\left\{-\frac{1}{T}\left(\sum_{i,j}\theta(x_{i,j},x_{i-1,j})+\theta(x_{i,j},x_{i,j-1})+\sum_{i,j}\xi_{i,j}(x_{i,j})\right)\right\}}$$

Simplify by canceling multiplicative terms that occur in both numerator and denominator.

$$\exp\left\{-\frac{1}{T}\left(\xi_{i,j}(x_{i,j})+\sum_S\theta(x_{i,j},x_{i-1,j})+\theta(x_{i,j},x_{i,j-1})\right)\right\}=g(x_{i,j},...)$$

If this looks complicated, for example it involves variables that are not neighbors of $x_{i,j}$, you did something wrong.

Problem 3(2pt) Take logarithm of the conditional probability and implement code that computes this probability distribution. You will use logsum

```
function p = conditional(x,i,j,theta,xi,T,k)
% x is the current configuration
% i,j are coordinates of variable to update
% theta and xi are energy functions
% T is temperature
% k is the size of state space for each x(i,j)
width = size(x,2);
height = size(x,1);
g = zeros(k,1);
if i>1
   g = g + theta((1:k)',x(i-1,j));
end
if i<height
    g = g + theta((1:k)',x(i+1,j));
end
if j>1
    g = g + theta((1:k)',x(i,j-1));
end
if j<width
    g = g + theta((1:k)',x(i,j+1));
end
g = g + xi((1:k)',i,j);
logp = -1/T*g;
logp = logp - logsum(logp);
p = exp(logp);
```

To test this out, use

```
k = 5;
T = 10;
y = 3*ones(3,3);
```

```
xi = @(v,i,j) (v - y(i,j)).^2;
theta = @(a,b) (a-b).^2;
p = conditional(y,2,2,theta,xi,T,k)
```

You should see something like this

```
p =
    0.0545
    0.2442
    0.4026
    0.2442
    0.0545
```

Implement code that samples from the conditional

```
function new = sample(x,i,j,theta,xi,T,k)
p = conditional(x,i,j,theta,xi,T,k);
cdf = cumsum(p);
r = rand(1,1);
temp = cdf - r;
temp(temp < 0) = 0;
new = find(temp, 1);
```

Problem 4(2pt) A single iteration of Gibbs sampler sweeps through each variable in the model and updates it.

```
function gibbs(theta,xi,T,k,y)
% initialize x with noisy image
x = y;
% experiment with different iteration number
for it=1:...
    for i=1:size(x,1)
        for j=1:size(y,1)
            x(i,j) = ...
        end
    end
    figure(1);
    subplot(2,1,1);
    imagesc(x)
    energies(it) = energy(x);
    subplot(2,1,2)
    plot(energies(5:it));
    drawnow
end
```

Load image in `hw3.mat` and try

T=5,iter=10,order=default T=5,iter=100,order=default T=5,iter=100,order=rescale

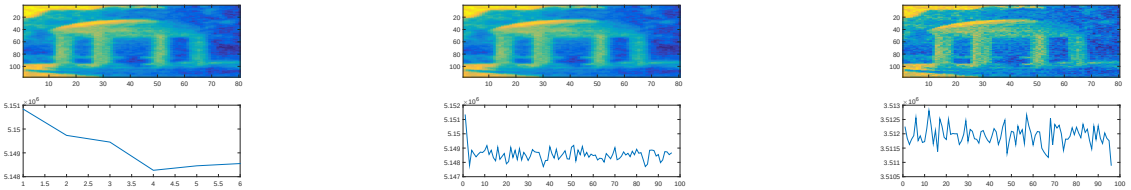


Figure 2: Denoised images for different settings of parameters ...

```
k = max(max(y));
T = 5;
xi = @(v,i,j) (v - y(i,j)).^2;
theta = @(a,b) (a-b).^2;
gibbs(theta,xi,T,k,y)
```

Experiment with Gibbs sampler using the above update. You can change temperature, and you can also rescale θ and ξ.

```
xi = @(v,i,j) 2*(v - y(i,j)).^2;
theta = @(a,b) 1/4*(a-b).^2;
```

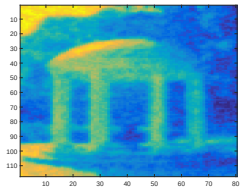
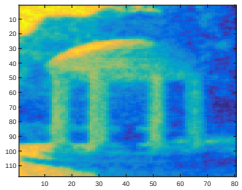
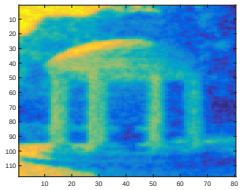
Change number of iterations, temperatures, scales of θ and ξ, order of updates. Insert images for different choices of these parameters.

Problem 5(2pt) We will now look at the behavior of the chain. It is common to throw away first iterations of of Gibbs sampler. This is called burn-in

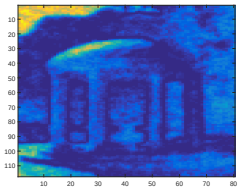
Modify the code to store:

- the best configuration of $\mathbf{x}$.
- store mean of the configurations across all iterations – you do not need to store all the configurations to compute the mean.
- store mean of the configurations across all iterations after iteration B
- compute per pixel variance in configurations

Best Mean Mean after burnin

Show the best image, mean image across all iterations, mean after burn-in,



variance after burn-in.

Problem 6(2pt) Choose the temperature, number of iterations and burn-in that seems to perform well. Change initialization of $\mathbf{x}$ to be random. Start 5 chains – all from different initial configurations–, collect the energies plots across all the iterations and plot them in the same plot but using different colors. Do all chains reach roughly the same energy levels? If not, run them for a longer time. Once you have the chains behaving similarly – they are mixing – generate the plot and add it below.

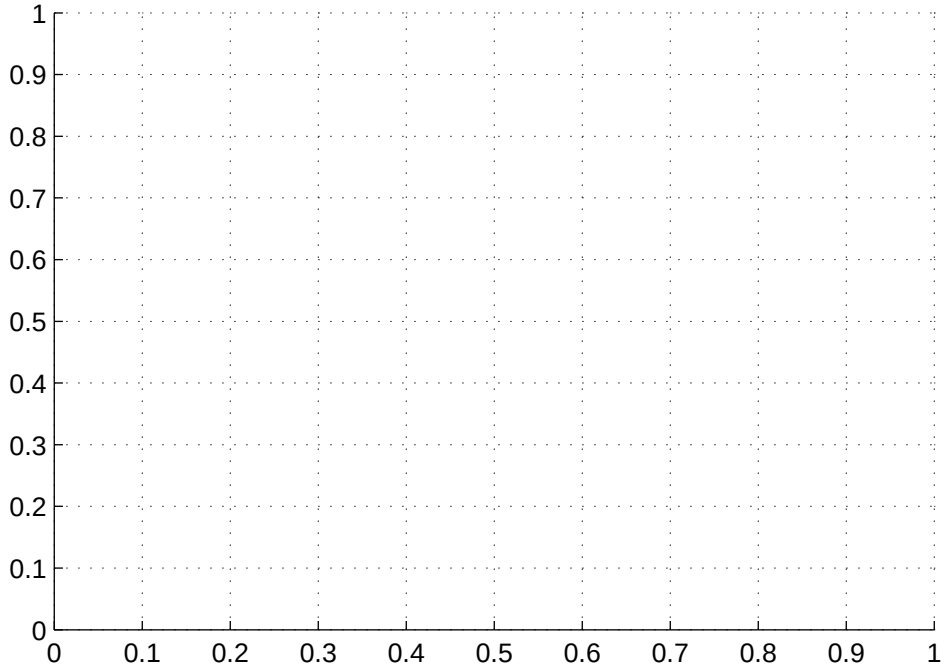


Figure 3: Energies across Gibbs iterations of 5 chains

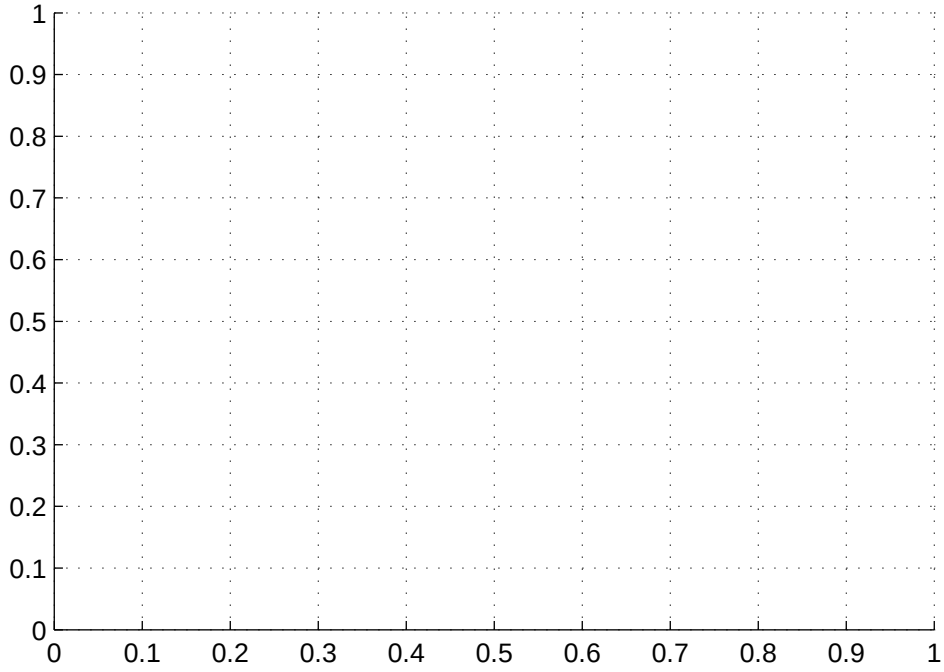


Figure 4: Denoised image