Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science

6.438 ALGORITHMS FOR INFERENCE
Fall 2014

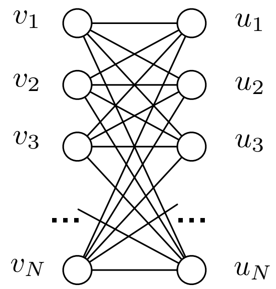
Problem Set 8

Issued: Tuesday, November 18, 2014 **Due:** Tuesday, November 25, 2014

Suggested Reading: Lecture notes 18–19

Problem 8.1

In problem 1 of homework 2, we introduced a distribution over matchings in a graph, i.e. subsets of edges such that no two edges share a vertex. Here we focus on the special case of a complete bipartite graph with vertices $v_1, \ldots, v_N$ on the left and $u_1, \ldots, u_N$ on the right, as shown:



In such a graph, a *perfect matching* is a matching which includes N edges. We are interested in sampling from a distribution over perfect matchings. We can denote a perfect matching using the variables $\sigma = [\sigma_{ij}] \in \{0,1\}^{N \times N}$, where $\sigma_{ij} = 1$ if v_i and u_j are matched and $\sigma_{ij} = 0$ otherwise. Observe that σ is a perfect matching if and only if

$$\begin{aligned} \sum_{k=1}^{N} \sigma_{ik} &= 1 \qquad \text{for all } 1 \le i \le N \\ \sum_{k=1}^{N} \sigma_{kj} &= 1 \qquad \text{for all } 1 \le j \le N. \end{aligned}$$

A perfect matching σ can also be thought of as a permutation $\sigma : \{1, \ldots, N\} \to \{1, \ldots, N\}$. E.g., if $\sigma_{12} = \sigma_{21} = \sigma_{33} = 1$, this would correspond to the permutation $\sigma(1) = 2$, $\sigma(2) = 1$, and $\sigma(3) = 3$. Consider the distribution defined by:

$$\begin{aligned} p(\sigma) &\propto \exp\left(\sum_{i,j} w_{ij}\sigma_{ij}\right) \mathbb{1}\{\sigma \text{ is a perfect matching}\} \\ &= \exp\left(\sum_{i} w_{i\sigma(i)}\right) \mathbb{1}\{\sigma \text{ is a perfect matching}\} \end{aligned}$$

where $w_{ij} \ge 0$ for all i, j.

(a) In this part, consider the uniform distribution over perfect matchings, i.e. $w_{ij} = 0$ for all i, j. Describe a simple procedure to sample σ from this uniform distribution.

(b) Consider the Metropolis-Hastings rule defined by: choose $i, i' \in \{1, \ldots, N\}$ uniformly at random. If $i = i'$, do nothing, otherwise with probability

$$R = \min(1, \exp(w_{i\sigma'(i)} + w_{i'\sigma'(i')} - w_{i\sigma(i)} - w_{i'\sigma(i')})$$

swap $\sigma(i)$ and $\sigma(i')$ (i.e. set $a \triangleq \sigma(i)$ and $b \triangleq \sigma(i')$ then define a new permutation $\sigma'(j) = \sigma(j)$ for $j \neq i, i'$ and $\sigma'(i) = b$ and $\sigma'(i') = a$).

(i) Show that for any perfect matching σ

$$p(\sigma) \geq \frac{1}{N! \exp(Nw^*)},$$

where $w^* = \max_{i,j} w_{ij}$.

(ii) Show that under the above Markov chain, for any valid transition $\sigma \to \sigma'$

$$\mathbf{P}_{\sigma\sigma'} \geq \exp(-2w^*) \frac{1}{N^2}.$$

(iii) For the conductance of this Markov chain, argue using (i) and (ii) that

$$\begin{aligned}\Phi &= \min_S \frac{\sum_{\sigma \in S, \sigma' \in S'} p(\sigma) \mathbf{P}_{\sigma\sigma'}}{p(S)p(S')} \\ &\geq \frac{1}{N! \exp(Nw^*)} \frac{1}{N^2} \exp(-2w^*).\end{aligned}$$

(iv) Using (iii), obtain a bound on the mixing time of the Markov chain.

Problem 8.2 (Practice)

In this problem, we develop an efficient algorithm for sampling from a two-dimensional Ising model. In particular, suppose all variables x_{ij} take values in $\{-1, +1\}$. Using the graph structure $\mathcal{G}$ shown in Figure Figure 8.2(a), define the distribution

$$p(x; \theta) \propto \exp\left\{ \sum_{(i,j) \in \mathcal{E}} \theta x_i x_j \right\}.$$

(a) Derive the update rules for a node-by-node Gibbs sampler for this model. Implement the sampler in MATLAB® and run it for 1000 iterations (sweeps over all the
on an Ising model of size 60×60 with coupling parameter $\theta = 0.45$. Show
of the variables after every 100 iterations.

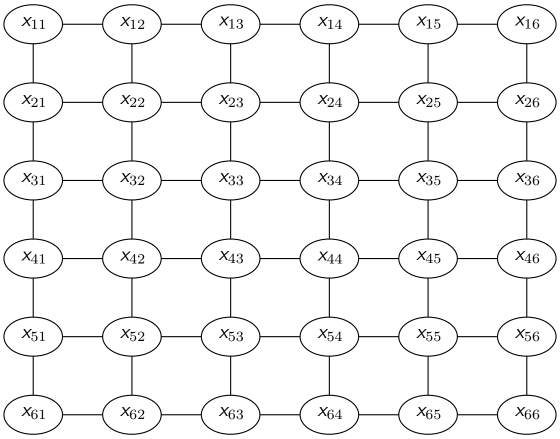

x_{11} x_{12} x_{13} x_{14} x_{15} x_{16}
x_{21} x_{22} x_{23} x_{24} x_{25} x_{26}
x_{31} x_{32} x_{33} x_{34} x_{35} x_{36}
x_{41} x_{42} x_{43} x_{44} x_{45} x_{46}
x_{51} x_{52} x_{53} x_{54} x_{55} x_{56}
x_{61} x_{62} x_{63} x_{64} x_{65} x_{66}

Figure 8.2(a)

(b) Suppose we are given a tree-structured undirected graphical model $\mathcal{T}$ with variables $\mathbf{\mathsf{y}} = (\mathsf{y}_1, \ldots, \mathsf{y}_N)$. Give an efficient procedure for sampling from the joint distribution $p_{\mathbf{\mathsf{y}}}(\mathbf{y})$.

Hint: One way to generate an exact sample from a distribution over random variables $\mathbf{\mathsf{y}} = (\mathsf{y}_1, \ldots, \mathsf{y}_N)$ is to sample each y_i from the distribution $p_{\mathsf{y}_i|\mathsf{y}_1,\ldots,\mathsf{y}_{i-1}}(\cdot|y_1, \ldots, y_{i-1})$ in sequence. Show that with a suitable variable ordering, all of these conditional distributions can be computed by running belief propagation once.

(c) In *block Gibbs sampling*, we partition a graph into r subsets $A_1, \ldots, A_r$. In each iteration, for each A_i, we sample x_{A_i} from the conditional distribution $p_{\mathsf{x}_{A_i}|\mathsf{x}_{\mathcal{V}\setminus A_i}}$.

For the Ising model $\mathcal{G}$ described above, consider the two comb-shaped subsets A and B shown in Figure Figure 8.2(b) (b). Describe how to use your sampler from part (b) to perform the block Gibbs updates. (For this part, you may assume a black-box implementation of your sampling procedure from part (b).)

(d) Implement the block Gibbs sampler from part (c) in MATLAB As in part (a), set $\theta = 0.45$ and run the sampler for 1000 iterations, showing the state of the variables after every 100 iterations. Which of the two samplers appears to mix faster?

To make your life easier, we have provided you with a MATLAB routine (comb_sum_product.m) for computing the sum-product messages on the comb graph. Feel free to use this routine as a black box, even though it does more work than necessary.

Problem 8.3

This problem explores how the partitioning scheme can be utilized for approximate estimation in related situations.

(a) Consider a rectangular 2D grid graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ over N nodes (an example of such a

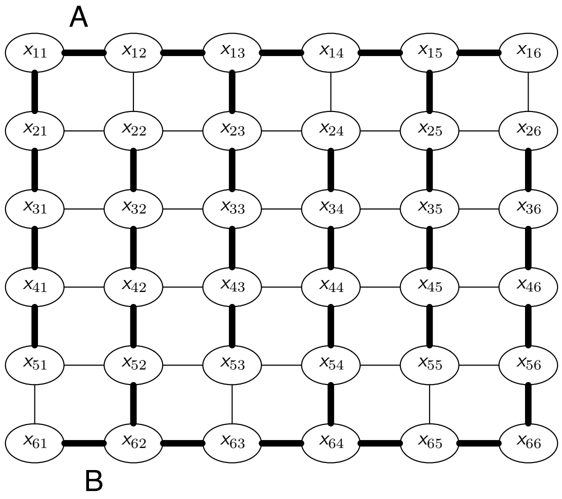


Figure 8.2(b)

graph is shown in Figure 8.2(a)) and a distribution with $\mathcal{G}$ as its graphical model:

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

where $Z(\theta)$ is the normalization constant or partition function. Suppose that $0 \leq \theta_i, \theta_{ij} \leq 1$ for all $i \in \mathcal{V}, (i,j) \in \mathcal{E}$ and $\mathbf{x}_i$ are binary (i.e. $x_i \in \{0,1\}$). We know that $\mathcal{G}$ possesses a randomized partition, in other words, for any $\epsilon > 0$, there exists a randomized partition of $\mathcal{V}$ into disjoint subsets $\mathcal{V}_1, \ldots, \mathcal{V}_M$ such that

- $\mathcal{V} = \bigcup_{m=1}^{M} \mathcal{V}_m$,
- $|\mathcal{V}_m| \leq \frac{4}{\epsilon^2}$ for all $1 \leq m \leq M$,
- $p(e \in \mathcal{E}^c) \leq \epsilon$ where $\mathcal{E}^c = \mathcal{E} \setminus \cup_{m=1}^{M} \mathcal{E}_m$ and $\mathcal{E}_m = \mathcal{E} \cap (\mathcal{V}_m \times \mathcal{V}_m)$.

Use the elimination algorithm on the subproblems to derive an efficient algorithm to approximate $\log Z(\theta)$ that is similar to the MAP case. Determine bounds on how well it approximates $\log Z(\theta)$.

(b) Now suppose that:

- $\theta_i \geq 0$ can take any non-negative value.
- $\theta_{ij} = -\infty$ for all $(i,j) \in \mathcal{E}$.
- x_i is binary for all i.

We shall define $0 \times (-\infty) = 0$. Thus for any $\mathbf{x}$:

$$p_{\mathbf{x}}(\mathbf{x}) \propto \begin{cases} 0 & \text{if any } (i,j) \in \mathcal{E} \text{ and } x_i = x_j = 1 \\ \exp\left(\sum_{i \in \mathcal{V}} \theta_i x_i\right) & \text{otherwise} \end{cases}.$$

(i) Why doesn't the MAP partition algorithm derived in lecture apply here?

(ii) Through a variation of the MAP partition algorithm, derive a new algorithm to compute an approximate MAP for this setup. Determine bounds on how well it approximates the MAP. *Hint: Fix the values of specific x_i so that (i) is not a problem.*

Problem 8.4
Suppose we have the following sequence of scalar observations

$$\mathsf{y}_n = f(\mathsf{x}) + \mathsf{v}_n, \quad n = 0, 1, \ldots, \tag{1}$$

where $f(\cdot)$ is a known *invertible* function and $\mathsf{v}_n \sim$ iid, $\mathcal{N}(0, \sigma_{\mathsf{v}}^2)$, and from which we want to estimate the random variable x whose a priori distribution is $\mathsf{x} \sim \mathcal{N}(0, \sigma_{\mathsf{x}}^2)$. The variances σ_{v}^2 and σ_{x}^2 are known.

An estimate of x can be obtained via a recursive estimation algorithm based on particle filtering. Recall that the particle filtering algorithm computes approximations to the relevant posterior distributions according to the following recursion

$$\hat{p}_{\mathsf{x}_n|\mathsf{y}_1,\ldots,\mathsf{y}_n}(x_n) = \sum_{i=1}^{N} w_{n|n}^{(i)} \delta(x_n - x_n^{(i)})$$

$$\hat{p}_{\mathsf{x}_{n+1}|\mathsf{y}_1,\ldots,\mathsf{y}_n}(x_{n+1}) = \sum_{i=1}^{N} w_{n+1|n}^{(i)} \delta(x_{n+1} - x_{n+1}^{(i)})$$

where $x_n^{(i)}$ is the ith particle at time n and $w_{n|n}^{(i)}$ is the associated weight after having incorporated observation y_n, and where

$$\delta(x) = \begin{cases} 1, & \text{if } x = 0 \\ 0, & \text{otherwise.} \end{cases}$$

To simplify the analysis, assume that no resampling is involved.

From the posterior approximations so generated, a sequence of estimates $\hat{\mathsf{x}}_n = \hat{x}(\mathsf{y}_1, \ldots, \mathsf{y}_n)$ is produced by treating the approximations as perfect and generating the MAP estimate corresponding to this posterior.

(a) Write a set of state space equations corresponding to this estimation problem. (Formally, this means determine a dynamical state space model such that the MAP estimate of the state at time n in this model based on data through time n coincides with the MAP estimate of x based on the same data.)

Next, suppose we initially prepare N particles $\{x_0^{(i)}\}$, $i = 1, 2, \ldots, N$, each of which is drawn independently from the prior distribution $\mathcal{N}(0, \sigma_{\mathsf{x}}^2)$. Let $\epsilon = \min_i |\mathsf{x} - x_0^{(i)}|$.

(b) Determine the particle filter propagate and update equations. Specifically, in as simple a form as possible, express $w_{n|n}^{(i)}$, $w_{n+1|n}^{(i)}$, and $x_n^{(i)}$ in terms of the function $f(\cdot)$, the parameters N, σ_{v}^2, the initial set of particles $\{x_0^{(i)}\}$ and the observations $\{y_k\}$.

(c) Show that $\lim_{n\to\infty} \hat{\mathsf{x}}_n \neq \mathsf{x}$ whenever $\epsilon > 0$.

(d) As an alternative method of part (b), let us design the particle filter algorithm for the following state-space model

$$\mathsf{x}'_{n+1} = \mathsf{x}'_n + \mathsf{u}_n$$

with

$$\mathsf{y}_n = f(\mathsf{x}'_n) + \mathsf{v}_n$$

and $\mathsf{x}'_0 \sim \mathcal{N}(0, \sigma_{\mathsf{x}}^2)$, where $\mathsf{u}_n \sim$ iid, $\mathcal{N}(0, \sigma_{\mathsf{u}}^2)$ for the known variance σ_{u}^2 and are independent of $\{\mathsf{v}_k\}$. We use the importance density $p_{\mathsf{x}'_{n+1}|\mathsf{x}'_n}(x'_{n+1}|x'_n)$ to obtain $x'^{(i)}_{n+1}$ in the propagate step and start with the same initial particles $\{x_0^{(i)}\}$ as in part (b). We similarly treat the posterior approximations so produced as perfect and generate the corresponding MAP estimate $\hat{\mathsf{x}}'_n = \hat{x}'(\mathsf{y}_1, \ldots, \mathsf{y}_n)$ of x'_n based on the same data.

Show that even for $n = 1$, the x'_n generated by this new procedure can give a better estimate of our original x than the algorithm of part (b). Specifically, show that $\Pr[|\mathsf{x} - \hat{\mathsf{x}}'_1| < |\mathsf{x} - \hat{\mathsf{x}}_1|] > 0$ whenever $\epsilon > 0$.

Problem 8.5
In this problem, you will implement the particle filter to estimate states from measurements. Some things are left ambiguous, so state your assumptions in your solution.

Consider the following state model

$$\mathsf{x}[t+1] = 0.97\mathsf{x}[t] + \mathsf{v}[t],$$

where $\mathsf{v}[t]$ is independently drawn Gaussian with variance σ_{v}^2.

And consider this non-linear noisy measurement model

$$\mathsf{y}[t] = \mathsf{x}^2[t] + \mathsf{w}[t],$$

where $\mathsf{w}[t]$ is independently drawn Gaussian with variance σ_{w}^2.

Furthermore, the initial state has the following distribution:

$$p_{\mathsf{x}[0]}(x) = \begin{cases} 1, & -1 \leq x < -0.5 \text{ or } 0.5 \leq x < 1 \\ 0, & \text{otherwise} \end{cases}$$

(a) Set up a simulation of this system to generate state and measurement sequences for $t = 1, 2, \ldots, T$, to use in part (b). Use $T = 50$, $\sigma_{\mathsf{v}}^2 = 10^{-4}$, and $\sigma_{\mathsf{w}}^2 = 0.25$. Plot a sample path of the states along with measurements.

(b) Implement and run a particle filter on the data. Store the particles and weights for all time instances. Resample when $N_{\text{eff}} < 0.6N$.

Experiment with different values for N, the number of particles. What are the tradeoffs involved?

For two different values of N, provide a plot of the particles shaded with their weight as a function of time. Mark instances of resampling on the plot. Your plots may use commands similar to:

```
times = kron(1:T,ones(N,1)); scatter(times(:),particles(:),12,weights(:));
```

(c) Now, suppose the measurement is sometimes completely erased; thus, the new measurement model is:

$$\mathsf{y}[t] = \gamma[t]\mathsf{x}^2[t] + \mathsf{w}[t],$$

where $\gamma[t]$ is a binary random variable, independent among all times, that takes value 1 with probability p and value 0 with probability $1 - p$. Repeat part (b) for this new measurement model, taking p to be 0.9.

6.438 Algorithms for Inference
Fall 2014