Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science

6.438 Algorithms For Inference
Fall 2014

Problem Set 5

Issued: Friday, October 17, 2014 **Due:** Tuesday, October 28, 2014

Suggested Reading: Lecture notes 11–13

Problem 5.1

Consider the 2-node undirected graphical model in Figure 5.1 , where the variables, x_a, x_b are binary, and the compatibility functions are given by:

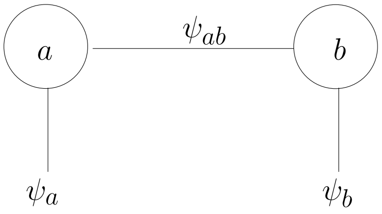


Figure 5.1

$$\psi_a(0) = \psi_a(1) = \psi_b(0) = \psi_b(1) = 1 \tag{1}$$

$$\psi_{ab}(0,0) = \psi_{ab}(1,1) = 1 \quad , \quad \psi_{ab}(1,0) = \psi_{ab}(0,1) = 10 \tag{2}$$

(a) Compute the max-marginals for each variable, and show that there is no unique maximizing value for each of the variables. Explain why independently choosing the maximizing values for each of the variables does not lead to the maximum of the joint distribution.

(b) In general, we shall define *edge* max-marginals $\bar{p}_{ij}(x_i, x_j) = \max_{\mathbf{x}\setminus\{x_i,x_j\}} p_{\mathbf{x}}(\mathbf{x})$ for every edge (i, j) in a tree.

Choose one of the maximizing values at one node, say node “a”. Show that in order to maximize the joint probability, you can use the edge max-marginal to determine the value of the other node.

Problem 5.2

Consider a hidden Markov model (HMM) with states x_i and observations y_i for $i = 1, 2, \ldots, n$ for some n. The states are binary-valued, i.e., $x_i \in \{0, 1\}$. Moreover, the model is homogeneous, i.e., the potentials are given by

$$\psi_i(x_s, x_{s+1}) = \psi(x_s, x_{s+1}), \quad \psi'_i(x_s, y_s) = \psi'(x_s, y_s), \quad \text{for } i = 1, 2, \ldots, n$$

Given the observations $y_i = y_i$ for $i = 1, \ldots, n$, we are interested in state estimates $\hat{x}_i(y_1, \ldots, y_n)$ for $i = 1, \ldots, n$ that maximize the probability that at least one of those state estimates $\hat{x}_i$ is correct.

(a) The desired state estimates can be expressed in the form

$$(\hat{x}_1, \ldots, \hat{x}_n) = \arg\min_{\hat{x}_1,\ldots,\hat{x}_n} p_{\mathsf{x}_1,\ldots,\mathsf{x}_n|\mathsf{y}_1,\ldots,\mathsf{y}_n}(f(\hat{x}_1), \ldots, f(\hat{x}_n)|y_1, \ldots, y_n).$$

Determine the function $f(\cdot)$.

(b) Show that if only the marginal distributions $p_{\mathsf{x}_i|\mathsf{y}_1,\ldots,\mathsf{y}_n}(x_i|y_1, \ldots, y_n)$, $i = 1, \ldots, n$, for the model are available, the desired state estimates cannot be determined. In particular, construct two HMMs whose marginals coincide, but whose state estimates differ. *Hint:* It suffices to consider a model with $n = 2$, and in which the observations are independent of the states and thus can be ignored. Accordingly, express your answer in the form of two distributions $p_{\mathsf{x}_1,\mathsf{x}_2}(\cdot,\cdot)$ and $p'_{\mathsf{x}_1,\mathsf{x}_2}(\cdot,\cdot)$.

(c) Construct an example of an HMM in which our desired estimates are not the same as the MAP estimates obtained from running the Viterbi (i.e., max-product or min-sum) algorithm on our model. The same hint as in part (b) applies, so again give your answer in the form of a distribution $p_{\mathsf{x}_1,\mathsf{x}_2}(\cdot,\cdot)$.

(d) You are given two pieces of code (e.g., MATLAB® scripts).

The first routine implements the forward-backward algorithm, taking as input potential functions that describe a homogeneous HMM, and an associated list of n observations. It produces as output the list of marginal distributions for each of the associated n states conditioned on the full set of n observations, for the specified HMM.

The second routine implements the Viterbi algorithm, taking the same inputs as the forward-backward routine, but producing as output the most probable sequence of states x_i given the full set of n observations.

Describe how to use one or both of these routines to compute the desired estimates $\hat{\mathsf{x}}_i(y_1, \ldots, y_n)$ for $i = 1, \ldots, n$ for our model of interest, assuming the potentials are strictly positive. You are free to use these routines with any input values you like (whether or not related to the model of interest), and you can further process the outputs of these routines to compute the desired state estimates. However, in such further processing, you are not allowed to (re)use the model's potential functions or observations.

Problem 5.3 (Practice)

This problem explains how the problem of computing MAP and normalization constant for an undirected graphical model is related to the problem of computing marginals of variables. To that end, consider an undirected graphical model $G = (V, E)$ of N variables $\mathsf{x}_1, \ldots, \mathsf{x}_N$ with each $\mathsf{x}_i \in \{0, 1\}$. Let the associated probability distribution be

$$\begin{aligned} p_{\mathbf{x}}(\mathbf{x}) &\propto \exp\left(\sum_{i\in V} F_i(x_i) + \sum_{(i,j)\in E} G_{ij}(x_i, x_j)\right) \\ &\propto \exp(H(\mathbf{x})) = \frac{1}{Z}\exp\big(H(\mathbf{x})\big), \end{aligned} \tag{3}$$

where normalization constant (also known as partition function)

$$Z = \sum_{\mathbf{x} \in \{0,1\}^N} \exp(H(\mathbf{x})),$$

with potential function

$$H(\mathbf{x}) \triangleq \sum_{i \in V} F_i(x_i) + \sum_{(i,j) \in E} G_{ij}(x_i, x_j), \tag{4}$$

where $F_i : \{0,1\} \to \mathbb{R}_+$ and $G_{ij} : \{0,1\}^2 \to \mathbb{R}_+$ are arbitrary non-negative valued functions ($\mathbb{R}_+$ represents non-negative real values).

Oracle. Throughout, we assume that we have access to an *oracle* that provides answers to following query instantly (consider $O(1)$ computation):

For any graphical model of the form described above, for any given variable i, $1 \leq i \leq N$, it provides its marginal distribution.

In other words, given input H over N variables of form (4), for any $i,\ 1 \leq i \leq N$, the oracle will provide answer to $p_{\mathsf{x}_i}(0)$ with respect to the graphical model defined as per (3) for the given H.

Part 1. In the first part of this problem, we shall utilize the above mentioned oracle to find MAP assignment. Recall, that the MAP assignment $\mathbf{x}^*$ is such that $p_{\mathbf{x}}(\mathbf{x}^*) \geq p_{\mathbf{x}}(\mathbf{y})$ for all $\mathbf{y} \in \{0,1\}^N$, i.e. $H(\mathbf{x}^*) \geq H(\mathbf{y})$.

For the purpose of answering (a)-(c), assume the special structure:

$$F_i : \{0,1\} \to \{0,1,2\}, \text{ for all } i \in V, \text{and}$$

$$G_{ij} : \{0,1\}^2 \to \{0,1,2,3\}, \text{ for all } (i,j) \in E.$$

We shall assume that MAP is unique, i.e. $H(\mathbf{x}^*) > H(\mathbf{y})$ for all $\mathbf{y} \in \{0,1\}^N,\ \mathbf{y} \neq \mathbf{x}^*$.
To find $\mathbf{x}^*$, consider

$$p^{\beta}_{\mathbf{x}}(\mathbf{x}) \propto \exp(\beta H(\mathbf{x})), \qquad \text{for} \ \ \beta > 0. \tag{5}$$

Let $\mathbf{x}^*(\beta)$ be the MAP assignment of $p^{\beta}_{\mathbf{x}}$, that is,

$$\mathbf{x}^*(\beta) \in \operatorname*{argmax}_{\mathbf{x} \in \{0,1\}^N} p^{\beta}_{\mathbf{x}}(\mathbf{x}).$$

Note that, our interest is in finding $\mathbf{x}^*(1) = \mathbf{x}^*$.

(a) Show that, for all $\beta \geq 1$, $\mathbf{x}^*(\beta) = \mathbf{x}^*(1) = \mathbf{x}^*$.

(b) Argue that, for some $\beta^* = O(N^2)$, $p^{\beta^*}_{\mathbf{x}}(\mathbf{x}^*) > 1/2$.

Hint: Note that for this sub-part of the problem, $H(\mathbf{y})$ is integer valued for all $\mathbf{y} \in \{0,1\}^N$ and $\mathbf{x}^*$ is unique.

(c) Using (a) and (b), conclude that the *oracle* for p^{β^*} can be used to find $\mathbf{x}^*$.

Hint: For each i, the oracle for $p^{\beta^*}_{\mathsf{x}_i}$ should decide whether x_i^* equal to 0 or 1.

Part 2. Now, we shall use the oracle to find Z.

(d) For this subpart only, suppose $G_{ij}(x_i, x_j) = 0$ for all $(i,j) \in E$ and $(x_i, x_j) \in \{0,1\}^2$. Show that

$$Z = \prod_{i=1}^{N} \frac{\exp(F_i(0))}{p_{\mathsf{x}_i}(0)}.$$

Note that this suggests that when $\mathsf{x}_1, \ldots, \mathsf{x}_N$ are independent, Z can be written as a product of the inverse of node marginals (along with easy to evaluate quantities). Thus, the oracle for computing node marginals is useful.

(e) Explain how one can use the oracle to compute:

$$p_{\mathsf{x}_1}(0), p_{\mathsf{x}_2|\mathsf{x}_1}(0|0), \ldots, p_{\mathsf{x}_N|\mathsf{x}_{N-1},\ldots,\mathsf{x}_1}(0|0,\ldots,0).$$

Hint: You want to apply oracle for N different graphical models. Please be specific when defining these different graphical models.

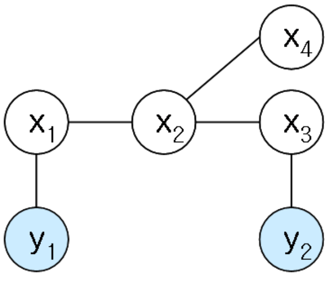
(f) Write down Z as a function of quantities from (e) and quantities that involve evaluation of H at a specific assignment in $\{0,1\}^N$.

Problem 5.4

Let $\mathbf{x} \sim \mathcal{N}^{-1}(\mathbf{h}_\mathbf{x}, \mathbf{J}_\mathbf{x})$, and $\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{v}$, where $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, \mathbf{R})$.

(a) Find the potential vector and the information matrix of $p(\mathbf{x}, \mathbf{y})$ and $p(\mathbf{x}|\mathbf{y})$.

(b) Consider the following Gaussian graphical model:



Let $\mathsf{y}_1 = \mathsf{x}_1 + \mathsf{v}_1$, $\mathsf{y}_2 = \mathsf{x}_3 + \mathsf{v}_2$, and $\mathbf{R} = I$. Find $\mathbf{C}$. Represent messages $h_{\mathsf{x}_3 \to \mathsf{x}_2}$ and $J_{\mathsf{x}_3 \to \mathsf{x}_2}$ in terms of the elements of $\mathbf{h}_\mathbf{x}$ and $\mathbf{J}_\mathbf{x}$.

(c) Now assume that we have an additional measurement $\mathsf{y}_3 = \mathsf{x}_3 + \mathsf{v}_3$, where v_3 is a zero-mean Gaussian variable with variance 1 and is independent from all other variables. Represent messages $h_{\mathsf{x}_3 \to \mathsf{x}_2}$ and $J_{\mathsf{x}_3 \to \mathsf{x}_2}$ in terms of the elements of $\mathbf{h}_\mathbf{x}$ and $\mathbf{J}_\mathbf{x}$.

Problem 5.5
In this problem we explore connections between Kalman filtering/smoothing and belief propagation. In particular, consider our standard state-space model:

$$\begin{aligned} x[n+1] &= ax[n] + v[n] \\ y[n] &= cx[n] + w[n], \end{aligned}$$

where for simplicity $x[n]$ and $y[n]$ are scalars. We assume that $x[0]$, $v[n]$, and $w[n]$ are all independent of each other, with zero-means and $\operatorname{var} x[0] = \sigma^2$, $\operatorname{var} v[n] = q$, $\operatorname{var} w[n] = r$. Also, assume that $x[0]$, $v[n]$, and $w[n]$ are all Gaussian random variables.

(a) Fix a time N, and assume that we are given observations $y[0], \ldots, y[N]$. We are interested in computing the marginal distribution of each $x[i]$ given the observations $y[0], \ldots, y[N]$. Draw an (undirected) graphical model that can be used to compute these estimates.

(b) (**Practice**) Write out the Gaussian belief propagation equations for your graph from part (a). Explain why the final estimates computed by Gaussian belief propagation should be the same as the estimates computed by using the Kalman smoother. Comment on the similarities and differences in the Gaussian belief propagation equations compared to the Kalman smoothing equations in information form (Refer to Jordan Chapter 15 for the Kalman smoothing equations in information form or derive them from the covariance form). Since Gaussian belief propagation computes $\mathbf{J}$ and $\mathbf{h}$ and the Kalman smoother computes $\mathbf{J}$ and $\mathbf{m}$, it is fine if you just comment on the way the two algorithms compute $\mathbf{J}$, i.e., don't worry about the differences in $\mathbf{h}$ and $\mathbf{m}$. Here $\mathbf{J}$ and $\mathbf{h}$ denote the usual information parameters used in Gaussian belief propagation, while $\mathbf{m}$ denotes the smoothed estimate of the mean that is calculated by the Kalman smoother.

(c) Can you find a method to generate filtered estimates (instead of smoothed estimates) using Gaussian belief propagation? That is, can you slightly modify Gaussian belief propagation to develop a recursive algorithm for computing the marginal distribution of $x[n]$ given $y[0], \ldots, y[n]$. (Hint: we basically just want you think about the messages passed in Gaussian belief propagation and how they relate to the Kalman filter).

(d) (**Practice**) Solve the following system of equations. You should solve this system by hand rather than using MATLAB.Explain why this is related to Gaussian belief propagation.

$$\begin{bmatrix} 1 & 0 & 0 & -4 & 1 & -3 & 0 \\ 0 & 4 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 2 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 3 & 0 & 0 & 1 \\ 2 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & -1 & -1 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & -3 & 0 & 0 & 6 \end{bmatrix} \mathbf{x} = \begin{bmatrix} -32 \\ 32 \\ 8 \\ 24 \\ 5 \\ 12 \\ 12 \end{bmatrix}$$

Problem 5.6
In this problem you'll implement a discrete time Kalman filter in MATLAB. Some built-in functions that you may find useful include: `rand`, `randn`, `inv`, `chol`, `qr`, `eps`, and `plot`. Consider the system:

$$\begin{aligned}\mathbf{x}[n+1] &= \begin{bmatrix} 0.9 & 0 \\ 0 & 0.9 \end{bmatrix} \mathbf{x}[n] + \begin{bmatrix} \epsilon^2 & 0 \\ 0 & \epsilon^2 \end{bmatrix} \mathbf{w}[n] \\ \mathbf{y}[n] &= \begin{bmatrix} 1 & 0 \\ 1 & \epsilon \end{bmatrix} \mathbf{x}[n] + \mathbf{v}[n],\end{aligned}$$

where $\mathbf{w}[n]$ and $\mathbf{v}[n]$ are zero-mean white noise processes uncorrelated with each other, both having unit variance.

Note that with ϵ small, one mode of the state variable $\mathbf{x}$ is highly "observable" from the output $\mathbf{y}$ and another is only weakly "observable." This type of ill-conditioning can lead to numerical difficulties (especially with more complex systems than in this simple example). You will see a modest indication of the problem here.

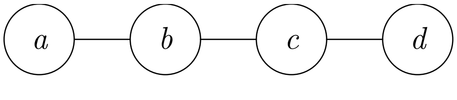
(a) Set up a simulation of this system to generate state and measurement sequences for $t = 1, 2, \ldots, 200$ for use in part (b). State your assumptions about the non-deterministic parts of the system.

(b) Implement and run a discrete time Kalman filter on the data. Initialize your filter with covariance:

$$\Lambda_{\mathbf{x}}[1] = \frac{1}{\epsilon} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Write code to implement the discrete time Kalman filter and plot the original data, the estimates, the estimation errors, and the one-step-prediction error covariances, for choices of ϵ that are (i) larger than, (ii) comparable to, and (iii) smaller than machine precision (see `eps` in MATLAB).

Problem 5.7 (Practice)
Consider the 4-node chain, as shown below, where each node can have a ternary state (0, 1, or 2), and where the (joint) probabilities for all 81 possible state sequences are *distinct.*



Suppose that we run the direction-free version of the max-product algorithm (i.e. the version that does not include the δ messages) to obtain the max-marginals $\bar{p}_i(x_i) = \max_{\mathbf{x}\setminus x_i} p_{\mathbf{x}}(\mathbf{x})$ at each node i, the results of which are indicated in the following table:

i	$\bar{p}_i(0)$	$\bar{p}_i(1)$	$\bar{p}_i(2)$
a	0.2447	0.0753	0.0234
b	0.2447	0.0118	0.1199
c	0.2447	0.1199	0.0169
d	0.2447	0.0346	0.0141

We seek the kth most likely state sequence $\mathbf{x}^k$, for $k = 1, 2, 3$. Here, the kth most likely state sequence means a specific joint state $\mathbf{x}^k(x_a^k, x_b^k, x_c^k, x_d^k)$ whose joint probability is the kth largest among all 81 state sequences.

(a) Find the most likely state sequence from the given set of max-marginals. Remember to provide your reasoning.

(b) Find the second most likely state sequence from the given set of max-marginals. Remember to provide your reasoning.

(c) Given the set of max-marginals above, list all the sequences that could be the third most likely state sequence. Explain your reasoning. (For this list, you may ignore constraints on the joint probability imposed by the graph's chain structure).

(d) Suppose that instead of gathering the *node* max-marginal data above, we had instead run a different kind of direction-free max-product algorithm on the same chain—one that produces *edge* max-marginals $\bar{p}_{ij}(x_i, x_j) = \max_{\mathbf{x} \setminus \{x_i, x_j\}} p_{\mathbf{x}}(\mathbf{x})$ for every edge (i, j). With the edge-max-marginal data, can we uniquely determine the third most likely state sequence? Explain.

Problem 5.8 (Practice)
Suppose we run the direction-free version of max-product algorithm on a tree with N nodes ($N \gg 1$). After the algorithm ends, K nodes face ties that cannot be broken without additional communication through the graph.

(a) What is the largest K for which this could happen? Give an example that results in such K and explain why K could not be larger.

(b) What is the smallest K for which this could happen? Give an example that results in such K and explain why K could not be smaller.

Problem 5.9 (Practice)
This problem deals with a common problem in biology — locating genes in a DNA sequence.

A DNA sequence is a string of nucleotides (of which there are four: A, T, G, C). Genes are contiguous parts of a DNA sequence that code for protein. Between genes are *intragenic regions* that do not code for protein. A gene itself consists of three regions in this order: the *promoter*, the *coding region*, and the *polyA*. For the sake of this exercise, assume the average number of nucleotides in each region is as follows: promoter - 5; coding region - 10; polyA - 6; intergenic region - 10. Also assume each region is associated with a different frequency with which nucleotides occur (i.i.d.), given by this chart:

	A	T	G	C
promoter:	0.1	0.1	0.4	0.4
coding region:	0.1	0.2	0.3	0.4
polyA:	0.7	0.1	0.1	0.1
intergenic region:	0.25	0.25	0.25	0.25

(a) Construct and simulate a hidden Markov model (HMM) for DNA sequences. Draw and fully label the state diagram for the hidden states. (*Hint*: You need to calculate the transition probabilities by using the average lengths of the regions. What assumptions do you need to make about the distributions of the lengths?)

Assume there is an equal probability of starting in any region at the beginning of the sequence. In MATLAB, generate a DNA sequence of 500 nucleotides with the true region labels. Draw the trellis diagram of the HMM and mark the state transitions for the first 5 sequence letters you generated.

(b) Suppose we want to infer the region labels from the DNA sequence. We can find the MAP estimates of the labels at each position in the sequence (from their marginals). Implement the forward-backward algorithm (or use your codes from Problem Set 4) and compute this labeling. How many labels were found incorrectly? What is more fundamentally wrong with this labeling?

(c) Implement the Viterbi algorithm and find the most probable sequence of region labels with it. Provide your code. How many labels were found incorrectly this time? Where did the errors occur?

(d) How would you modify the Viterbi algorithm to find the *second* most probable sequence of regions? How does computational and storage complexity compare with part (c)?

(e) (**practice**) Suppose we want to infer the number of genes in an observed DNA sequence, but don't care about the actual labeling. We can of course estimate a labeling first, then count the number of *promoter* regions, but can you think of a slight modification to the Viterbi algorithm that gives the answer directly? Make the modification and plot the number of genes found on each of the 4 partial paths (i.e., the best path ending at each of the M states at each time step) as the algorithm proceeds.

Problem 5.10 (Practice)

Consider a discrete state-space model given by

$$x[k+1] = a[k]x[k] + b[k]v[k],$$

where $v[k]$ is zero-mean, unit-variance, white, Gaussian, and independent of $x[m]$ for all $k \geq m$. $a[k]$ and $b[k]$ are unkown parameters.

We have the measurement sequence

$$y[k] = x[k] + w[k],$$

where $w[k]$ is zero-mean, unit variance, white, Gaussian, and independent of $x[m]$ for all $k \geq m$.

(a) Suppose that for this process, $E[x[k]x[k+l]] = e^{-0.02|l|}$ for all k. Determine $a[k]$ and $b[k]$ in the model above.

For the following parts, make sure you can simulate your model in MATLAB using the parameters you obtain in part (a). Assume that $\mathsf{x}[0]$ has Gaussian distribution with zero mean and unit variance.

(b) Implement Guassian Belief Propagation to compute the marginal of $x[k]$ after observing the y s (i.e. $p(x[k]|y[0], \ldots, y[K])$). Alternatively, you can implement Kalman smoothing (Rauch-Tung-Striebel algorithm) to compute the marginals. Include your code. (You may structure your code however you want, but if you choose to implement the Kalman filter as a standalone function, please exclude that portion of the code from your submission.)

(c) We know that $p(x[k]|y[0], \ldots, y[K]) = \mathcal{N}(x_{k|K}, P_{k|K})$ and $p(x[k]|y[0], \ldots, y[k]) = \mathcal{N}(x_{k|k}, P_{k|k})$. Calculate $x_{k|K}, P_{k|K}, x_{k|k}, P_{k|k}$ for $k = 0$ to $k = 50$ using either Gaussian Belief Propagation or the Kalman filtering and smoothing equations. Plot the true states and the state estimates $x_{k|K}$ and $x_{k|k}$. Also plot the variances $P_{k|K}$ and $P_{k|k}$ of the two methods. Comment on what you see.

Hint: If you're using Gaussian Belief Propagation, use your answer to 5.8(c) to compute $p(x[k]|y[0], \ldots, y[k])$.

6.438 Algorithms for Inference
Fall 2014