IE 598 Inference in Graphical Models

Mid-term Quiz

Spring 2015

This is a 24-hour take-home exam. you are allowed to use any of the course materials (homework, homework solutions, lecture notes, textbooks), but you are not allowed to consult anyone or search for hints on the web. Any questions should be directed to `swoh@illinois.edu`. I will be mostly responsive within an hour, except from 1:00AM till 7:00AM. There are 4 problems, check if you have all 4 when you receive this exam.

Problem 1. As mentioned in class, Gaussian BP allows to compute the minimum of a quadratic function

$$\widehat{x} = \arg\min_{x\in\mathbb{R}^n} \left\{ \frac{1}{2}\langle x, Qx\rangle + \langle b, x\rangle \right\}. \tag{1}$$

for $Q \in \mathbb{R}^{n\times n}$ positive definite, where $\langle a, b\rangle = a^T b$ indicates the standard inner product of two vectors. In this homework we will consider a case in which Q is not positive definite, but is symmetric and has full rank. in this case we can still define

$$\widehat{x} = -Q^{-1}b\,. \tag{2}$$

which is a stationary point (a saddle point) of the above quadratic function. The BP update equations are exactly the same as for the minimization problem with a positive definite Q. We claim that, when BP converges, it still computes the correct solution $\widehat{x}$.

We consider a specific model. An unknown signal $s_0 \in \mathbb{R}^n$ is observed in Gaussian noise

$$y = As_0 + w_0\,. \tag{3}$$

Here $y \in \mathbb{R}^m$ is a vector of observations, $A \in \mathbb{R}^{m\times n}$ is a measurement matrix, and $w_0 \in \mathbb{R}^m$ is a vector of Gaussian noise, with i.i.d. entries $w_{0,i} \sim \mathcal{N}(0, \sigma^2)$. We are given y and A, and would like to reconstruct the unknown vector s_0, and hence w_0.

A popular method consists in solving the following quadratic programming problem (known as *ridge regression*):

$$\widehat{s} = \arg\min_{s\in\mathbb{R}^n} \left\{ \frac{1}{2}\|y - As\|_2^2 + \frac{1}{2}\lambda\|s\|_2^2 \right\}. \tag{4}$$

We will do something equivalent. For $x \in \mathbb{R}^{m+n}$, $x = (z, s)$, $z \in \mathbb{R}^m$, $s \in \mathbb{R}^n$, we define a cost function

$$\mathcal{C}_{A,y}(x = (z, s)) \;\; = \;\; -\frac{1}{2}\|z\|_2^2 + \frac{1}{2}\lambda\|s\|_2^2 + \langle z, y - As\rangle\,. \tag{5}$$

We will look for the stationary point of $\mathcal{C}_{A,y}$.

(a) Show that the cost function $\mathcal{C}_{A,y}(x)$ can be written in the form

$$\mathcal{C}_{A,y}(x) = \frac{1}{2}\langle x, Qx\rangle + \langle b, x\rangle\,. \tag{6}$$

Write explicitly the form of the matrix $Q \in \mathbb{R}^{(m+n)\times(m+n)}$ and the vector $b \in \mathbb{R}^{m+n}$.

(b) Let $\widehat{x} = (\widehat{z}, \widehat{s})$ be the stationary point of $\mathcal{C}_{A,y}(z, s)$. Assuming it is unique, show that $\widehat{s}$ does coincide with the ridge estimator (4).

(c) Write the update rule for the BP algorithm (equivalent to the sum-product algorithm) to compute the stationary point $\widehat{x} = (\widehat{z}, \widehat{s})$ of $\mathcal{C}_{A,y}(x)$. [hint: use the same ideas from the Gaussian belief propagation for positive definite Q.]

(d) Prove the above claim that, if BP converges, then it computes $\widehat{x}$, cf. Eq. (2) even if Q is not positive definite.

Problem 2. Consider a stochastic process that transitions among a finite set of states $s_1, \ldots, s_k$ over time steps $i = 1, \ldots, N$. The random variables $X_1, \ldots, X_N$ representing the state of the system at each time step are generated as follows:

- Sample the initial state $X_1 = s$ from an initial distribution p_1, and set $i := 1$.
- Repeat the following:
  - Sample a duration d from a duration distribution p_D over the integers $\{1, \ldots, M\}$, where M is the maximum duration.
  - Remain in the current state s for the next d time steps, i.e., set
$$X_i := X_{i+1} := \ldots := X_{i+d-1} := s$$
  - Sample a successor state s' from a transition distribution $p_T(\cdot|s)$ over the other states $s' \neq s$ (so there are no sef-transitions).
  - Assign $i := i + d$ and $s := s'$.

This process continues indefinitely, but we only observe the first N time steps. You need not worry about the end of the sequence to do any of the problems. As an example calculation with this model, the probability of the sample state sequence $s_1, s_1, s_1, s_2, s_3, s_3$ is

$$p_1(s_1)p_D(3)p_T(s_2|s_1)p_D(1)p_T(s_3|s_2)\sum_{2\leq d\leq M} p_D(d) \ .$$

Finally, we do not directly observe the X_i's, but instead observe emissions y_i at each step sampled from a distribution $p_{Y_i|X_i}(y_i|x_i)$.

(*a*) For this part only, suppose $M = 2$, and $p_D(d) = \begin{cases} 0.6 & \text{for } d = 1 \\ 0.4 & \text{for } d = 2 \end{cases}$, and each X_i takes on a value from an alphabet $\{a, b\}$. Draw a minimal directed I-map for the first five time steps using the variables $(X_1, \ldots, X_5, Y_1, \ldots, Y_5)$. Explain why none of the edges can be removed.
[Note: you do not need to solve part (*a*) in order to solve part (*b*) and (*c*).]

(*b*) This process can be converted to an HMM using an *augmented state representation*. In particular, the states of this HMM will correspond to pairs (x, t), where x is a state in the original system, and t represents the time elapsed in that state. For instance, the state sequence $s_1, s_1, s_1, s_2, s_3, s_3$ would be represented as $(s_1, 1), (s_1, 2), (s_1, 3), (s_2, 1), (s_3, 1), (s_3, 2)$. the transition and emission distribution for the HMM take the forms

$$\tilde{p}_{X_{i+1},T_{i+1}|X_i,T_i}(x_{i+1}, t_{i+1}|x_i, t_i) = \begin{cases} \phi(x_i, x_{i+1}, t_i) & \text{if } t_{i+1} = 1 \text{ and } x_{i+1} \neq x_i \\ \xi(x_i, t_i) & \text{if } t_{i+1} = t_i + 1 \text{ and } x_{i+1} = x_i \\ 0 & \text{otherwise} \end{cases}$$

and $\tilde{p}_{Y_i|X_i,T_i}(y_i|x_i, t_i)$, respectively. Express $\phi(x_i, x_{i+1}, t_i)$, $\xi(x_i, t_i)$, and $\tilde{p}_{Y_i|X_i,T_i}(y_i|x_i, t_i)$ in terms of parameters p_1, p_D, p_T, $p_{Y_i|X_i}$, k, N, and M of the original model.

(*c*) We wish to compute the marginal probability for the final state X_N given the observations $Y_1, \ldots, Y_N$. If we naively apply the sum-product algorithm to the construction in part (*b*), the computational complexity is $O(Nk^2M^2)$. Show that by exploiting additional structure in the model, it is possible to reduce the complexity to $O(N(k^2+kM))$. In particular, give the corresponding rules for computing the forward messages $\nu_{i+1\to i+2}(x_{i+1}, t_{i+1})$ from the previous message $\nu_{i\to i+1}(x_i, t_i)$. Do not worry about the beginning or the end of the sequence and restrict your attention to $2 \leq i \leq N - 1$.
[Hint: substitute your solution from part (*b*) into the standard update rule for HMM messages and simplify as much as possible.]
[Note: If you cannot fully solve this part of the problem, you can receive substantial partial credit by constructing an algorithm with complexity $O(Nk^2M)$.]

Problem 3. In this problem we consider using Low-Density Parity Check (LDPC) codes to encode bits to be sent over a noisy channel.

Encoding. LDPC codes are defined by a factor graph model over a bipartite graph $G(V, F, E)$, where V is the set of variable nodes, each representing the bit to be transmitted, and F is a set of factor nodes describing the code and E is a west of edges between a bit-node and a factor node. The total number of variable nodes in the graph define the length of the code (also known as the block length), which we denote by $n \triangleq |V|$.

We consider binary variables $x_i \in \{-1, +1\}$ for $i \in V$, and all codewords that are transmitted satisfy

$$\prod_{i \in \partial a} x_i \quad = \quad +1 \ ,$$

which means that there are even number of -1's in the neighborhood of any factor node.

Channel. We consider a Binary Symmetric Channel, known as BSC(ε), where one bit is transmitted over the channel at each discrete time step, and each transmitted bit is independently flipped with probability ε. Precisely, let $x_i \in \{+1, -1\}$ be a transmitted bit and $y_i \in \{+1, -1\}$ be the received bit (at time i), then

$$\begin{aligned}
\mathbb{P}(y_i = +1 | x_i = +1) &= 1 - \varepsilon \ , \\
\mathbb{P}(y_i = -1 | x_i = +1) &= \varepsilon \ , \\
\mathbb{P}(y_i = -1 | x_i = -1) &= 1 - \varepsilon \ , \\
\mathbb{P}(y_i = +1 | x_i = -1) &= \varepsilon \ .
\end{aligned}$$

The conditional probability distribution over $x_1^n = [x_1, \ldots, x_n]$ given the observed received bits $y_1^n = [y_1, \ldots, y_n]$ is

$$\mu(x_1^n \,|\, y_1^n) \quad = \quad \frac{1}{Z} \prod_{i \in V} \psi_i(x_i, y_i) \prod_{a \in F} \mathbb{I}(\otimes x_{\partial a} = +1) \ ,$$

where $\psi_i(x_i, y_i) = \mathbb{P}(y_i | x_i)$ and $\otimes$ indicates product of binary numbers such that if $x_{\partial a} = \{x_1, x_2, x_3\}$ then $\otimes x_{\partial a} = x_1 \times x_2 \times x_3$ (to be precise we need to take $\psi_i(x_i | y_i) = \mathbb{P}(x_i | y_i)$, but this gives the exactly same conditional distribution as above since any normalization with respect to y_i's are absorbed in the partition function Z). This is naturally a graphical model on a factor graph $G(V, F, E)$ defined by the LDPC code.

(a) Write down the belief propagation updates (also known as the (parallel) sum-product algorithm) for this factor graph model for the messages $\{\nu_{i \to a}^{(t)}(\cdot)\}_{(i,a) \in E}$ and $\{\tilde{\nu}_{a \to i}^{(t)}(\cdot)\}_{(i,a) \in E}$.

(b) What is the computational complexity (how many operations are required in terms of the degrees of the variable and factor nodes) for updating one message $\nu_{i \to a}(\cdot)$ and one message $\tilde{\nu}_{a \to i}(\cdot)$ respectively? Explain how one can improve the computational complexity, to compute the message $\tilde{\nu}_{a \to i}^{(t)}(\cdot)$ exactly in runtime $O(d_a)$, where d_a is the degree of the factor node a.

(c) Now, we consider a different message passing algorithm introduced by Robert Gallager in 1963. The following update rule is a message passing algorithm known as the **Gallager A algorithm**. Similar to the belief propagation for BEC channels we studied in class, this algorithm also sends discrete messages (as opposed to real-valued messages in part (a)). Both $\nu_{i \to a}^{(t)}$'s and $\tilde{\nu}_{a \to i}^{(t)}$'s are binary, i.e. in $\{+1, -1\}$.

$$\begin{aligned}
\nu_{i \to a}^{(t+1)} &= \begin{cases} +1 & \text{if } \tilde{\nu}_{b \to i}^{(t)} = +1 \text{ for all } b \in \partial i \setminus a \ , \\ -1 & \text{if } \tilde{\nu}_{b \to i}^{(t)} = -1 \text{ for all } b \in \partial i \setminus a \ , \\ y_i & \text{otherwise} \ , \end{cases} \\
\tilde{\nu}_{a \to i}^{(t)} &= \prod_{j \in \partial a \setminus i} \nu_{j \to a}^{(t)} \ .
\end{aligned}$$

The interpretation of this update rule is that $\nu_{i\to a}$ messages trust the received bit y_i unless all of the incoming messages disagree with y_i, and $\tilde{\nu}_{a\to i}$ messages make sure that the consistency with respect to $\mathbb{I}(\otimes x_{\partial a})$ is satisfied. In this algorithm, the messages take values in $\{+1, -1\}$ and are the estimated values of x_i's, as opposed to the distribution over those values as in belief propagation.

We assume that random (ℓ, r)-regular bipartite graph is used to generate the LDPC code. In the resulting random graph, all variable nodes have degree ℓ and all factor nodes have degree r. Among all such graphs, a random graph is selected uniformly at random.

Define $W^{(t)}$ to be the (empirical) distribution of the messages $\{\nu^{(t)}_{i\to a}\}_{(i,a)\in E}$ and $Z^{(t)}$ to be the (empirical) distribution of the messages $\{\tilde{\nu}^{(t)}_{a\to i}\}_{(i,a)\in E}$. We assume the messages are initialized in such way that $\nu^{(0)}_{i\to a} = y_i$ for all $i \in V$. We also assume, without loss of generality, that all $+1$ messages were sent, i.e. $x_i = +1$ for all i. Then, let $w^{(t)} = \mathbb{P}(W^{(t)} = -1)$ be the probability that a message $\nu^{(t)}_{i\to a}$ is -1 for a randomly chosen edge (i, a), and let $z^{(t)} = \mathbb{P}(Z^{(t)} = -1)$ be the probability that a message $\tilde{\nu}^{(t)}_{a\to i}$ is -1 for a randomly chosen edge (i, a).

Write the **density evolution** *equations for $w^{(t)}$ and $z^{(t)}$, describing how the random distribution of the messages $w^{(t)}$ and $z^{(t)}$ evolve.* [We are looking for a clean answer. Specifically, the number of operations required to compute $z^{(t)}$ from $w^{(t)}$ should be $O(1)$. The same technique that reduced computation in part (b) should be helpful.]

(d) Write the density evolution equation for a single scalar variable $w^{(t)}$, by substituting $z^{(t)}$. This gives a fixed point equation in the form of $w^{(t)} = F(w^{(t-1)})$ for some F. Plot (using MATLAB to your favorite numerical analysis tool) the function $y = F(x)$ and the identify function $y = x$, for $\ell = 3$ and $r = 4$, and for two values of $\varepsilon = 0.05$ and $\varepsilon = 0.1$ Explain the figure in terms of the error probability of the (3,4)-code on those two BSC(ε)'s.

Problem 4 In this problem, we explore the connections between minimum cut of a graph and pairwise Markov random fields in binary alphabets. Consider a graphical model defined on an undirected graph $G(V, E)$,

$$\mu(x) \quad = \quad \frac{1}{Z} \exp\{-\sum_{i \in V} \phi_i(x_i) - \sum_{(i,j) \in E} \phi_{ij}(x_i, x_j)\} \ ,$$

for $x = [x_1, \ldots, x_n] \in \{0, 1\}^n$. We further assume for now that $\phi_{ij}(0, 0) = \phi_{ij}(1, 1) = 0$ for all $(i, j) \in E$ (meaning they are zero-diagonal when we consider the functions as 2×2 matrices) such that

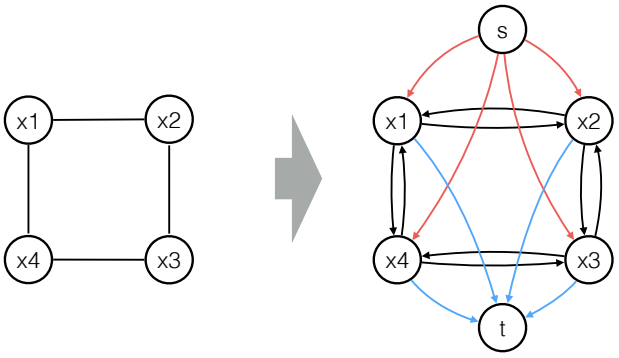
$$\phi_i(\cdot) \ = \ \begin{bmatrix} \phi_i(0) \\ \phi_i(1) \end{bmatrix} \ , \quad \text{and} \quad \phi_{ij}(\cdot, \cdot) \ = \ \begin{bmatrix} 0 & \phi_{ij}(0,1) \\ \phi_{ij}(1,0) & 0 \end{bmatrix} \ .$$

Our goal is to find the maximum likelihood estimate, the one that maximizes the above joint distribution. In order to find the maximizer, we pose this question as a problem of finding the minimum cut of a graph.

Given a pairwise MRF on $G(V, E)$ and the compatibility functions $\phi_{ij}(\cdot, \cdot)$'s, we first create a new **directed** and **weighted** graph as follows.

- Add one node for the source s and one node for the sink t.
- Add an edge from source s to all nodes in V (red edges in the figure below).
- Add an edge from all nodes in V to the sink t (blue edges in the figure below).
- make all edges in E reciprocal (by taking the undirected edge E and making them in to two edges in opposite directions; black edges in the figure below).

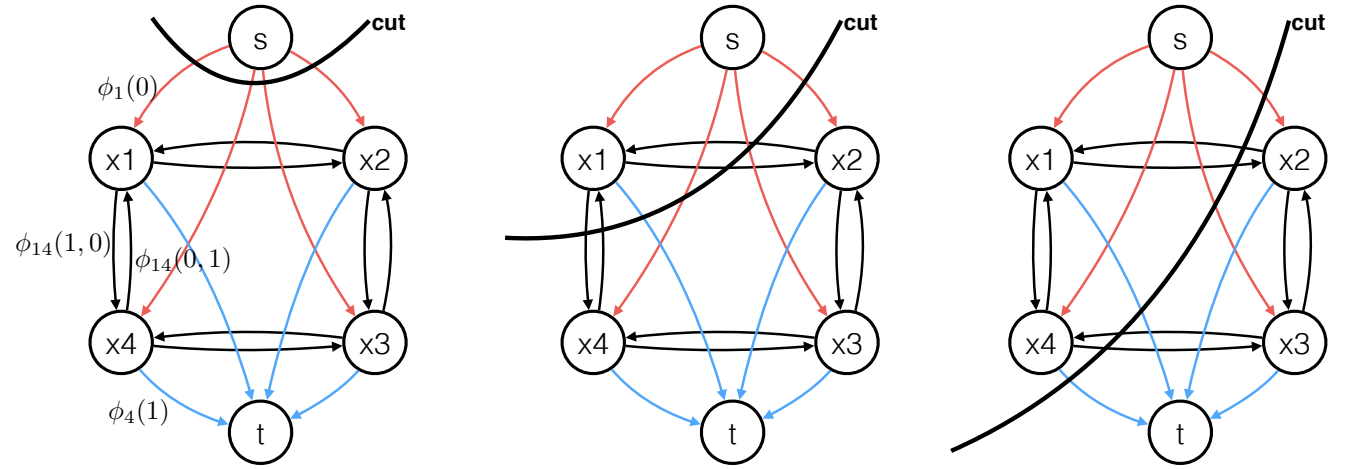
An example of a 2×2 grid G, that is transformed is shown below. The colors do not have particular meanings, it is there to help you understand the creation of the new graph. We will find the minimum cut in this transformed graph, after putting appropriate non-negative weights on the edges. A **cut** in a graph is partition of the nodes into two disjoint sets, one containing the source and the other containing the sink. **The value of a cut** is the total weight of the edges that start from a node in the same partition as the source and end in a node in the sink side of the partition, i.e. those that go from the source side of the partition to the other. Note that in the minimum cut, for each node in V, EITHER the edge connecting to the sink will be cut, OR the edge connecting from the source will be cut, but NOT BOTH (since the source and the sink are constrained to be on different sides of the cut). Once we find the minimum cut in this graph, we will assign ZERO to the sink side of the cut and ONE to the source side. This defines a one-to-one mapping between an assignment of binary values in the MRF and a cut in the transformed graph $H(V \cup \{s, t\}, D)$.



Our goal is to minimize $E(x) \triangleq \sum_{i \in V} \phi_i(x)i) + \sum_{(i,j) \in E} \phi_{ij}(x_i, x_j)$ (which is equivalent as finding the most likely assignment). The following costs on the edges (also called capacities in max-flow min-cut context) ensures that the min-cut of the transformed graph H corresponds to the minimizer of $E(x)$.

- Assign $\phi_i(0)$ to the edge from the source (s, i).
- Assign $\phi_i(1)$ to the edge to the sink (i, t).
- Assign $\phi(1, 0)$ to the edge (i, j) and $\phi_{ij}(0, 1)$ to the edge (j, i).

An example below shows that this assignment ensures that the value of the cut corresponds to the energy $E(x)$ of the corresponding assignment. In general, cut values are equal to the energy $E(x)$ pf the corresponding assignment x.



assignment: $x = [0, 0, 0, 0]$
cut value: $\phi_1(0) + \phi_2(0) + \phi_3(0) + \phi_4(0)$

$x = [1, 0, 0, 0]$
$\phi_1(1) + \phi_{14}(1, 0) + \phi_{12}(1, 0)$
$+\phi_2(0) + \phi_3(0) + \phi_4(0)$

$x = [1, 0, 0, 1]$
$\phi_1(1) + \phi_{12}(1, 0)$
$+\phi_2(0) + \phi_3(0) + \phi_4(1) + \phi_{34}(01)$

It is known that when the cost on the edges are non-negative, the minimum cut can be found efficiently. Hence, when all $\phi_{ij}(0, 0) = \phi_{ij}(1, 1) = 0$ and $\phi_i(x_i)$'s, $\phi_{ij}(0, 1)$'s and $\phi_{ij}(1, 0)$'s are all non-negative, then the costs on the edges are all non-negative and the minimizer of $E(x)$ can be found efficiently by running the off-the-shelf min-cut solvers on H.

(a) Suppose $\phi_1(0) < 0$, and the rest of the compatibility functions are all non-negative, and $\phi_{ij}(0, 0) = \phi_{ij}(1, 1) = 0$ for all $(i, j) \in E$. Find a new $\phi'_1(x_1)$ such that

- $\phi'_1(0)$ and $\phi'_1(1)$ are non-negative; and
- the minimizer of $E'(x) = \phi'_1(x_1) + \sum_{i \in V \setminus \{1\}} \phi_i(x_i) + \sum_{(i,j) \in E} \phi_{ij}(x_i, x_j)$ is the minimizer of $E(x)$.

Then, the corresponding transformed graph H with the new costs from $\phi'_1(x_1)$ can be solved for min-cut, since all costs are non-negative.

(b) Now, consider a general case when $\phi_{ij}(0, 0)$'s and $\phi_{ij}(1, 1)$'s are not necessarily zero. Explain how to assign costs to the directed edges of H (not just for the example given above, but for general $H(V \cup \{s, t\}, D)$ defined from general $G(V, E)$), such that **the value of a cut in this new H is equal to the energy $E(x) = \sum_{i \in V} \phi_i(x_i) + \sum_{(i,j) \in E} \phi_{ij}(x_i, x_j)$ for the corresponding assignment** x. Note that we do not worry about computational complexity of finding the minimum-cut in this part, and focus in posing the problem as a min-cut problem.

[hint: consider changing $\phi_i(x_i)$'s and $\phi_{ij}(x_i, x_j)$'s in order to get new $\phi'_{ij}(x_i, x_j)$'s such that the diagonals are zero.]

(*c*) Suppose $\phi_i(x_i)$'s are all non-negative and $\phi_{ij}(x_i, x_j)$'s are also all non-negative. Assigning costs to the edges of H as per the solution of part (*b*), it is possible that some edges are assigned negative costs. This is problematic, since min-cut cannot be efficiently solved. However, when all pairwise compatibility functions are **sub-modular**, then the minimizer of $E(x)$ can be found efficiently. We will prove that this is possible, by constructing a new graph H with non-negative costs under sub-modularity assumption.

A function $f(\cdot)$ over two binary variables is said to be sub-modular if and only if

$$f(0,0) + f(1,1) \quad \leq \quad f(0,1) + f(1,0) \; .$$

Suppose $\phi_i(x_i)$'s are non-negative and $\phi_{ij}(x_i, x_j)$'s are non-negative and sub-modular. Explain how to assign costs to the directed edges of H (not just for the example given above, but for general $H(V \cup \{s,t\}, D)$ defined from general $G(V,E)$), such that

- the value of a cut in this new H is equal to the energy $E(x) = \sum_{i \in V} \phi_i(x_i) + \sum_{(i,j) \in E} \phi_{ij}(x_i, x_j)$ for the corresponding assignment x; and
- all costs are non-negative.

[hint: consider changing $\phi_i(x_i)$'s and $\phi_{ij}(x_i, x_j)$'s in order to get new $\phi'_{ij}(x_i, x_j)$'s such that the diagonals are zero and the off-diagonals are non-negative.]