CS 281A/Stat 241A Homework Assignment 6 (due December 6)

- This homework assignment will count 1.5 times as much as the previous assignments.
- Do **one** of problems 1 and 2, and then do all of problems 3, 4, and 5.

1. *Triangulation ≡ Junction tree.*

   In Chapter 16 we prove that the property of being *triangulated* is equivalent to the property of *having a junction tree*. The proof involves an intermediate concept—that of *decomposability*. (That is, we prove that a graph is triangulated iff it is decomposable, and a graph is decomposable iff it has a junction tree). Provide a direct proof that shows that a graph is triangulated iff it has a junction tree.

2. *Kalman filter implementation.*

   The files `kfx.dat` and `kfy.dat` on the course website contain $x_1, \ldots, x_{500}$ and $y_1, \ldots, y_{500}$ respectively (one entry per line) from data generated from a state-space model. The model is of a particle moving in a plane under random forces and damping. More specifically, the 4-dimensional state x of the particle at a given time step is:

   $$x = \begin{pmatrix} x^1 \\ \dot{x}^1 \\ x^2 \\ \dot{x}^2 \end{pmatrix} \tag{1}$$

   where x^1 and x^2 represent the particle's coordinates in the plane, and $\dot{x}^1$ and $\dot{x}^2$ are its velocities in the two dimensions. The state in each dimension ($i = 1, 2$) evolves as $x^i_{t+1} = x^i_t + \dot{x}^i_t$, and the velocity in each dimesion evolves as $\dot{x}^i_{t+1} = 0.98\dot{x}^i_t + w^i_t$. Lastly, the observations y are noisy measurements of the particle's position. In detail, we have:

   $$A = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0.98 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0.98 \end{bmatrix}$$

   $$G = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$$

   $$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

   $$\begin{aligned} R &= 2500I \\ Q &= I \end{aligned}$$

   For the initial condition, x_1 is distributed $\mathcal{N}(0, I)$.

   In this problem, we will implement the Kalman filter to estimate the particle's state from the noisy measurements y_t.

(a) Plot the evolution of the particle's true position (x_t^1, x_t^2) using the ground-truth data from `kfx.dat`. All other plots should be plotted on top of this one.

(b) Plot the noisy observations y_t of the particle's positions.

(c) Implement the Kalman filter, and find $\hat{x}_{t|t}$ for each t using the model above and the observations y_t. Plot the resulting estimates of the particle's positions $(\hat{x}_{t|t}^1, \hat{x}_{t|t}^2)$ across time. (Also see note below.)

(d) Implement the RTS smoother, and find $\hat{x}_{t|T}$ for each t using the model above and the observations y_t. Plot the resulting estimates of the particle's position $(\hat{x}_{t|T}^1, \hat{x}_{t|T}^2)$.

(e) Comment on the relative qualities of using the measurements y_t directly as estimates of the particle position, the Kalman filter estimates, and the RTS smoother estimates.

(f) (Bonus) Implement the approximation described in the note below (assuming you didn't use it earlier), and repeat parts (c) and (d) using it. How much does this change your results?

Note. The Kalman filter requires calculating $P_{t|t}$ and $P_{t|t-1}$ to find the gain matrices K_t. A common approximation to this is to iterate the update equations for $P_{t|t}$ and $P_{t|t-1}$ until they converge, and then to substitute in the values to which they converged wherever we see $P_{t|t}$ and $P_{t|t-1}$. This results in a constant gain matrix K_t that does not depend on t. Likewise, for smoothing, the same approximation gives a constant L_t that does not depend on t. If you find that these approximations gives a simpler implementation (say because it does not require keeping around many matrices), you are welcome to use them in (c) and (d).

3. *Noisy OR.*

Consider a directed graph in which a node Y has parents $\{X_i\}$, for $i \in \{1, 2, \ldots, I\}$, and let Y be parametrized as a "noisy-OR" of its parents (cf. Chapter 6). Note that if we moralize this graph, then we connect all of the parents. Thus the computational complexity appears to be $O(2^I)$, which is impractically large for modest values of I.

Show how to convert the noisy-OR graph (in which a single node Y has I parents X_i) into a chain-structured graph in which no node has more than two parents. (Hint: make use of the Z_i variables used to define the noisy-OR in Chapter 6). What happens when you moralize this graph? I.e., what is the complexity of inference in this graph? Is this inconsistent with the $O(2^I)$ complexity cited above? Why or why not?

4. *Ising model.*

Consider an undirected graphical model $\mathcal{G}(\mathcal{V}, \mathcal{E})$ in which binary nodes $X_v \in \{0, 1\}$, for $v \in \mathcal{V}$, are arranged in a two-dimensional grid. We define $\mathcal{E}$ by letting each node be connected to its four nearest neighbors: the node to the left, the node to the right, the node above and the node below. (Let the leftmost nodes in the grid be connected to the rightmost nodes in the grid, and let the topmost nodes be connected to the bottommost nodes. I.e., the grid lies on a torus).

We define a *nearest-neighbor Ising model* via the usual definition of joint probability:

$$p(x_{\mathcal{V}}) = \frac{1}{Z} \prod_{u \in \mathcal{V}} \psi(x_u) \prod_{(u,v) \in \mathcal{E}} \psi(x_u, x_v),$$

where the potentials are as follows:

$$\begin{aligned} \psi(x_u) &= \exp\{\theta_{u0} x_u\} \\ \psi(x_u, x_v) &= \exp\{\theta_{uv} x_u x_v\}, \end{aligned}$$

for $u \in \mathcal{V}$ and $(u, v) \in \mathcal{E}$.

Consider a nearest-neighbor Ising model with 100 nodes arranged in a 10x10 grid. Let the θ_{i0} parameters be 1 and -1 in alternating columns. Let the θ_{ij} parameters be equal to 0.5.

(a) Implement Gibbs sampling for this Ising model. Run the Gibbs sampler for a "long time." Collect statistics and print out a matrix of the estimated expected values of each of the nodes.

(b) Implement the mean-field approximation to the Ising model. Initialize the variational parameters to random values near 0.5. Run the fixed-point equations until convergence and print out a matrix of the estimated expected values of each of the nodes.

(c) Compare the matrices obtained in the two cases by computing the average squared difference between matrix entries.

(d) Redo the problem, but with a fully-connected Ising model. (This is an Ising model in which $\mathcal{E} = \mathcal{V} \times \mathcal{V}$.) The θ_{i0} parameters should be the same as in the first problem, but the θ_{ij} should be equal to 0.1.

5. Consider the following *factorial model* for clustering. There are K binary latent variables $Z_i \in \{0, 1\}$, and a real-valued observed vector $Y \in \mathbb{R}^M$. The probability model is:

$$p(z_1, z_2, \ldots, z_K \mid \pi) = \prod_{k=1}^{K} \pi_i^{z_i} (1 - \pi_i)^{1 - z_i}$$

$$p(y \mid z_1, z_2, \ldots, z_K, \mu, \sigma^2) = \mathcal{N}(\sum_{k=1}^{K} z_i \mu_i, \sigma^2 I),$$

where each μ_i is an M-dimensional vector and I is the identity matrix of dimension $M \times M$. Let $\theta = (\mu_1, \ldots, \mu_K, \pi_1, \ldots, \pi_K, \sigma^2)$ denote the set of all of the parameters.

Note that there are 2^K possible configurations of the latent variables. Thus, for modest values of K, summing over all configurations of the latent variables is infeasible, and we need to consider approximations.

(a) Derive a mean-field approximation for inference in the factorial model. That is, consider a variational distribution:

$$q(z_1, z_2, \ldots, z_k \mid \lambda) = \prod_{k=1}^{K} \lambda_i^{z_i} (1 - \lambda_i)^{1 - z_i},$$

and derive a fixed-point algorithm that attempts to find values of the variational parameters $\{\lambda_i\}$ that minimize the KL divergence $D(q(z \mid \lambda) \mid\mid p(z \mid y, \theta))$.

(b) Derive a Gibbs sampler for inference in the factorial model. That is, show how to compute:

$$p(z_i \mid z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_K, y, \theta).$$

(c) Derive the M step of an EM algorithm for this model.

(d) Implement (in Matlab or Splus) two versions of an approximate EM algorithm for this model, one using the mean-field algorithm for the E step, and the other using the Gibbs sampler for the E step.

(e) For some fixed setting of the parameters θ and the observation y, plot various statistics of the Z_i variables as a function of sampling iterations for the Gibbs sampler. Try various values of σ^2. How is convergence time affected?

(f) Examine the data `images.jpg` shown on the course website. This shows 100 greyscale 4×4 images generated by randomly combining several features and adding some noise. Try to guess what these features are by inspecting the images. How many are there? Would you expect factor analysis to do a good job modeling these images? How about a mixture of Gaussians? Explain your reasoning.

(g) Run your algorithms (EM based on Gibbs or mean field) on the data set generated by `genimages.m`. What features μ do the algorithms find? (Rearrange them into 4×4 images). How do the Gibbs and mean field results differ? Explain any choices you make along the way and the rationale beyond them (e.g., how to set K, how to initialize parameters, latent states, and variational parameters).