Grad AI. 15-780 Fall, 2007

Homework 3

- *Homework deadline: 10:30am on November 1*
- *Please print your code and hand it in with the hard copy of your homework. Also send a copy of your code by e-mail to both TAs (gholling@andrew.cmu.edu and thlin@cs.cmu.edu).*

1. **Conditional Independence in Bayesian Networks (30 pts)**

   (a) The following Bayesian networks are all part of the alarm network introduced in class and in Russell and Norvig. Write the factored joint distribution implied by the following Bayesian networks, in the form of $p(X, Y) = p(X)p(Y|X)$. (2 pts each)

$$\begin{tikzcd} B \arrow[r] & A & E \arrow[l] \end{tikzcd}$$

i.

$$\begin{tikzcd} B \arrow[r] & A \arrow[r] & M \end{tikzcd}$$

ii.

$$\begin{tikzcd} M & A \arrow[l] \arrow[r] & J \end{tikzcd}$$

iii.

   (b) We use the notation $X \perp Y$ to denote the variable X being independent of Y, and $X \perp Y|Z$ to denote X being independent of Y given Z. Now prove that $B \perp E$ in the above Bayesian network of 1(a).i., $B \perp M|A$ in the Bayesian network of 1(a).ii., and $M \perp J|A$ in the Bayesian network of 1(a).iii., based on above joint distribution factoring. Also indicate whether these are the only independence assumptions embedded in the above Bayesian networks, and list other independence assumptions if you believe there is any. (3 pts each)

   (c) List all the independence assumptions embedded in the original alarm network (shown below), e.g. $B \perp M|A$ and $B \perp M|\{A, E\}$. Write a SHORT sentence about an intuitive reason why you listed or not listed $B \perp E|M$. Hint: there are more than 15 independent assumptions. (10 pts)

$$\begin{tikzcd} B \arrow[dr] & & E \arrow[dl] \\ & A \arrow[dl] \arrow[dr] & \\ M & & J \end{tikzcd}$$

(d) A *Markov blanket* of a variable X, denoted as MB(X), consists of its parents, children, and children's co-parents (i.e. children's parents other than X). After the exercises above, you should be able to see that a variable is conditionally independent of all other variables given its Markov blanket. See Russell and Norvig chapter 14.2 (pp. 499) for more details. Please list the Markov blanket of each variable in the alarm Bayesian network. (5 pts)

2. **Maximum Likelihood Estimation and Hidden Markov Models (30 pts)**

(a) The *exponential distribution* provides a good representation for time intervals between random events (e.g. bus arrivals). The probability density function (PDF) of an exponential distribution is,

$$f_{exp}(x;\lambda) = \lambda e^{-\lambda x} \ , \ x \geq 0$$

where λ is a rate parameter, corresponding to the number of events per unit time (e.g. buses per hour). Assume you have T data points $x_1, x_2, \cdots, x_T$, what is the likelihood of the data given λ, $L(x_1, \cdots, x_T|\lambda)$? (5 pts)

(b) Based on your data and likelihood above, derive the MLE for λ. (8 pts)

(c) A continuous density hidden Markov model is a HMM whose observations (and emissions) are continuous. For this problem we would like to learn the parameters for a HMM with a discrete set of states where each state outputs (emits) values from using an exponential distribution model.

Following conventional notation, $\{s_1, \cdots, s_N\}$ is the set of states. Assume you have K observation sequence, $\mathbf{X} = [\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \cdots, \mathbf{X}^{(K)}]$, where $\mathbf{X}^{(k)} = [x_1^{(k)}, \cdots, x_{T_k}^{(k)}]$ is the kth sequence. For the kth sequence,

- let $[q_1^{(k)}, \cdots, q_{T_k}^{(k)}]$ be the sequence of hidden states.
- the initial probability is $\pi_i = p(q_1^{(k)})\ \forall k$.
- the transition probability is $a_{ij} = p(q_t^{(k)} = s_j|q_{t-1}^{(k)} = s_i)\ \forall k, t$.
- the emission probability is $p(x_t^{(k)}|q_t^{(k)} = s_i) = f_{exp}(x_t^{(k)};\lambda_i)\ \forall k, t$.
- define $\alpha_t^{(k)}(i) = p(x_1^{(k)}, \cdots, x_t^{(k)}, q_t^{(k)} = s_i)$, $\beta_t^{(k)}(i) = p(x_{t+1}^{(k)}, \cdots, x_{T_k}^{(k)}|q_t^{(k)} = s_i)$.

Please write down the forward and backward algorithm of this HMM, i.e. how to calculate $\alpha_t^{(k)}(i)$ and $\beta_t^{(k)}(i)$ given π_i, a_{ij}, λ_i. (7 pts)

(d) The forward-backward algorithm you developed is the E-step of the EM algorithm; the MLE of λ you developed in problem 2(b) is the basis of the M-step. Now develop the complete EM algorithm for this HMM. (10 pts)

3. **PROGRAMMING PROBLEM: Inference in Bayesian Networks (40 pts)**

Please e-mail your code for this section in a zip file to both TAs. To avoid mixup, please e-mail with the following subject line and archive file name:

Subject: 15-780 Homework 1 Submission
Archive name: (yourID)-hw1.zip
Where (yourID) is your andrew or cs ID.

WARNING: This problem takes considerably more time than the rest of the assignment. Do NOT leave this until the last minute.

(a) Implement exact inference by enumeration (see below for explanation). (8 pts)

Write a function *enumeration_ask* that calculates the conditional probability distribution of **one query variable** given a set of evidence variables in a Bayesian network. See the pseudocode in Russell and Norvig chapter 14.4 (pp. 504) for reference. For simplicity, all variables are binary, so all "distributions" mentioned in Russell and Norvig, including the returned value of the function, can be represented by the probability of the variable being true.

Two sample Bayesian network will be given to you in the support archive, named *alarm* and *pedigree*, details below. The following Matlab functions are provided, to access the Bayesian network data structure:

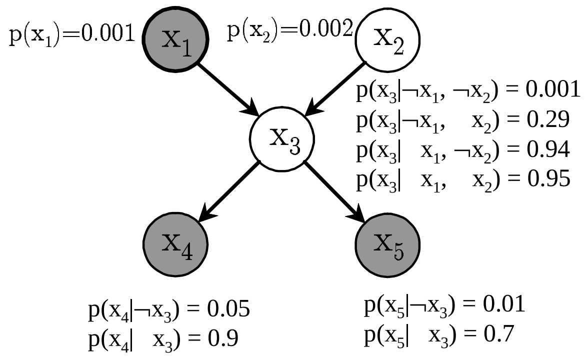
- *create_alarm_bn*: creates the *alarm* Bayesian network.
- *create_pedigree_bn*: creates the *pedigree* Bayesian network.
- *bn_vars*: returns the variables in a Bayesian network, partially ordered from parents to children.
- *bn_parents*: returns the parents of a variable.
- *bn_cpt*: returns the conditional distribution of a variable, given the specified values of its parents. This corresponds to one row of the conditional probability table (CPT).

See README and type "help *func_name*" in Matlab (or read corresponding scripts) for documentations and examples.

Note that the recursive enumeration must be performed from parents to children, i.e. the list of variables must be partially ordered such that parents are always before their children. The function *bn_vars* will provide the ordered variables list (actually the variable index itself in the Bayesian networks given below are already ordered this way, so the ordered list is just $1, 2, \cdots, N$).

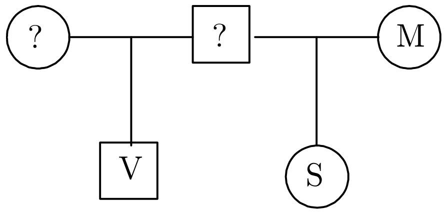
Please write your code by modifying the provided file enumeration_ask.m, which contains suggested API with documentation. Matlab is STRONGLY recommended, particularly because writing necessary support code in other languages is time consuming.

(b) Run your exact inference implementation on the following two Bayesian network inference problems. It should be straightforward for you to convert the question below into a function call to your code (by using appropriate variable indexes labeled on the graph below); type "help enumeration_ask" for an example. Both questions ask the conditional probability of one query variable being true given a set of evidence variables. Report run time and results.

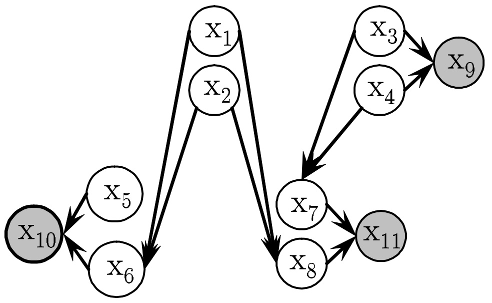


i. The first Bayesian network, created by *create_alarm_bn*, is the alarm network in Russell and Norvig. As shown in the graph above, variable are indexed from 1 to 5 (denoted as X_1 to X_5), and CPT are the same as in Russell and Norvig figure 14.2. Evidence variables are shaded, while query variables are shaded and circled by a thick line. Calculate the conditional probability $p(X_1|X_4, \neg X_5)$. Hint: verify with some queries that you know the answer first, e.g. $p(X_1|X_4, X_5)$ should be about 0.284. (4 pts)

ii. The second Bayesian network, created by *create_pedigree_bn*, is about genetic inference. Consider a victim V in a plane crash, whose only family members are his half-sister S and the sister's mother M (not V's mother). Their pedigree is shown below. You need to determine whether certain remains belong to V based on genetic fingerprints of S and M. This can be solved by a Bayesian network shown below, indexed from 1 to 11. Evidence and query variables are shaded, while normal circles are hidden variables. You do not need to worry about the CPT if you are using Matlab; otherwise the CPT is explained in the documentation of *create_pedigree_bn.m*.



Pedigree



Bayesian network

The hidden variables $(X_1, X_2, \cdots, X_8)$ correspond to unobserved genetic information in the so-called Mendelian inheritance: humans have two copies of each chromosome, one from the father and one from the mother. During reproduction, one copy (chosen randomly) will be passed to the next generation. Assume you cannot determine which copy is from which parent, but can only obtain partial information in the observed variables (X_9, X_{10}, X_{11}). However, you do not have to understand Mendelian inheritance to solve this problem. Now using the structure and CPT provided in the archive, calculate the conditional probability $p(X_{10}|\neg X_9, X_{11})$. (4 pts)

(c) Recall the Markov blanket MB(X) described in problem 1(d). For a general Bayesian network, let *parents*(X) denotes the parent variables of a variable X, and *children*(X) denotes the children variables of X. Write the formula of the conditional probability of $X = x$ given its Markov blanket, $p(x|\text{MB}(X))$. (4 pts)

(d) Implement approximate inference by MCMC. (10 pts)

Read "Inference by Markov chain simulation" in Russell and Norvig chapter 14.5 carefully, and write a function *mcmc_ask* that implements a Gibbs sampler as described. Again please write your code by modifying the provided file mcmc_ask.m, which contains API with documentation. We provide a function to calculate $p(x|\text{MB}(X))$ for you (type help for documentation); you can compare the code with your answer for problem 3(c).

- *bn_cond_mb*: calculates the conditional probability of a variable given its Markov blanket.

(e) Run your MCMC implementation on the two problems above, *alarm* and *pedigree*. Report run time and results.

Note: these two Bayesian networks are small, so run time of MCMC will be longer than that of exact inference. You can check the scalability of both methods by comparing their complexities, if interested.

i. Again on *alarm* Bayesian network, calculate $p(X_1|X_4, \neg X_5)$ by running *mcmc_ask* for 5,000 iterations. Hint: verify with some queries that you know the answer first, more than once, but except differences because this is approximate inference. (5 pts)

ii. Again on *pedigree* Bayesian network, calculate $p(X_{10}|\neg X_9, X_{11})$ by running *mcmc_ask* for 50,000 iterations. (5 pts)