Onur Payrort
2010401036

29.05.2016
Sunday

EE583 - Take Home Final

Q-1)

a) There are $L^n - 1$ independent parameters over X.

b) If we assign every nodes with as possible as much parents we could reach the maximum number of independent parameters. So, there is;

$$(L-1)+L(L-1)+L^2(L-1)+\dots+L^{n-2}(L-1)+L^{n-1}(L-1)$$

$$= (L-1)\sum_{i=1}^{n} L^{i-1} = (L-1)\frac{(1-L^n)}{(1-L)} = L^n - 1$$ independent parameters at most.

Here we set $k=n-1$.

c) This question has same solution with above part.

$$(L-1)+L(L-1)+L^2(L-1)+\dots+L^{n-1}(L-1)$$

$$= (L-1)\sum_{i=1}^{n} L^{i-1} = (L-1)\frac{(1-L^n)}{(1-L)} = L^n - 1$$ independent parameters.

Q 2-)

Def.1 — Undirected graphical model G is called Markov Random Field if two nodes are conditionally independent whenever they are seperated by evidence nodes. So, for any node X_i, following conditional property hold:

$$P(X_i \mid X_{G \setminus i}) = P(X_i \mid X_{N_i})$$

$X_{G \setminus i}$ = all nodes except X_i $\qquad$ X_{N_i} = neighborhood of X_i

Def.2 — A probability distribution $P(X)$ on an undirected graphical model G is called a Gibbs distribution if it can be factorized into positive functions defined on cliques that cover all nodes and edges of G. That is;

$$P(X) = \frac{1}{Z} \prod_{c \in C} \phi_c(X_c)$$

C: set of all cliques

Z: normalization constant.

. The Hammersley Clifford Theorem claims that above 2-definitions are equivalent.

. So we can represent any positive distribution in log-domain;

$$P(X) = \frac{1}{Z} \prod_{c \in C} \phi_c(X_c)$$

$$\Rightarrow \log P(X) = \sum_{c \in C} \log \phi_c(X_c) - \log Z$$

$$\Rightarrow \log P(X) = \sum_{c \in C} w_c f_c(X_c) - \log Z$$

$$\Rightarrow P(X) = \frac{1}{Z} \exp\Big(\sum_{c \in C} w_c f_c(X_c)\Big)$$

Q3)

In this question I start the factors order 0 for each. In question Grades factor include g1,g2,g3 however I take them g0,g1,g2 as others for write the code efficiently.

- The joint probability of the non-loopy networks for given parameters are;

0.6000 0.3000 0.0036 0.0600 0.000036

The joint probabilities are ordered such that the first is for joint probability of node 1, second for node 2 and so on.

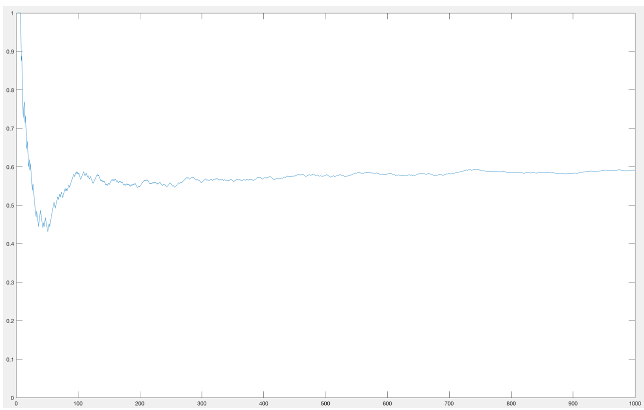
- The joint probability of the loopy networks for given parameters are;

0.6000 0.3000 0.0036 0.0600 0.00000216

My joint probability algorithm and the running codes are in Appendix Q3.

Q4)
a)
In this question I find the marginal probability of the vector for n=2,3,….,10. I put the marginal probabilities here in order from 2 to 10.

0.6120 for n=2
0.6010 for n=3
0.6150 for n=4
0.6020 for n=5
0.6010 for n=6
0.6060 for n=7
0.6220 for n=8
0.6040 for n=9
0.6000 for n=10



I also put here graph for n=10. This graph is contain number of iteration versus the average value marginal probability of X. Since I take probabilities random the starting point of graph change every run but final convergences are almost same.

My direct (top-down) sampling algorithm in Appendix Q4a.

b)
In this question I find the marginal probability of the vector for n=2,3,….,10. I put the marginal probabilities here in order from 2 to 10.

0.6160 for n=2
0.6054 for n=3
0.6044 for n=4
0.6044 for n=5
0.6044 for n=6
0.6044 for n=7
0.6044 for n=8
0.6044 for n=9
0.6044 for n=10

I don't put any graph because there is no convergences. From messages we could find the marginal probabilities directly.

My belief propagation (sum-product) algorithm in Appendix Q4b.

Q4)

c)

$$p_s(x|y,z) = \left[a^x(1-a)^{1-x}\right]^{(1-(y-z)^2)} \cdot \left[b^x(1-b)^{1-x}\right]^{(y-z)^2}$$

$$\log p_s(x|y,z) = \log\left(\left[a^x(1-a)^{1-x}\right]^{1-(y-z)^2} \cdot \left[b^x(1-b)^{1-x}\right]^{(y-z)^2}\right)$$

$$= (1-(y-z)^2)\log(a^x(1-a)^{1-x}) + (y-z)^2\log(b^x(1-b)^{1-x})$$

$$= (1-(y-z)^2)\left[x\log\left(\frac{a}{1-a}\right)+\log(1-a)\right] + (y-z)^2\left[x\log\left(\frac{b}{1-b}\right)+\log(1-b)\right]$$

$$= x\log\left(\frac{a}{1-a}\right)+\log(1-a)-x(y-z)^2\log\left(\frac{a}{1-a}\right)-(y-z)^2\log(1-a)+x(y-z)^2\log\left(\frac{b}{1-b}\right)+$$
$$(y-z)^2\log(1-b)$$

$$\Rightarrow p_s(x|y,z) = (1-a)\cdot\exp\left[\log\left(\frac{b(1-a)}{a(1-b)}\right)x(y-z)^2+\log\left(\frac{1-b}{1-a}\right)(y-z)^2+\log\left(\frac{a}{1-a}\right)x\right]$$

$$\Rightarrow \boxed{C_{s_1}=1-a\ ,\quad \alpha=\log\left[\frac{b(1-a)}{a(1-b)}\right]\ ,\quad \beta=\log\left(\frac{1-b}{1-a}\right)\ ,\quad \gamma=\log\left(\frac{a}{1-a}\right)}$$

$$p_s(x) = c^x(1-c)^{1-x}$$

$$\log p_s(x) = \log[c^x(1-c)^{1-x}]$$

$$= x\log\left(\frac{c}{1-c}\right)+\log(1-c)$$

$$p_s(x) = (1-c)\exp\left[\log\left(\frac{c}{1-c}\right)x\right]$$

$$\Rightarrow \boxed{C_{s_2}=1-c\ ,\quad \delta=\log\left(\frac{c}{1-c}\right)}$$

d.)

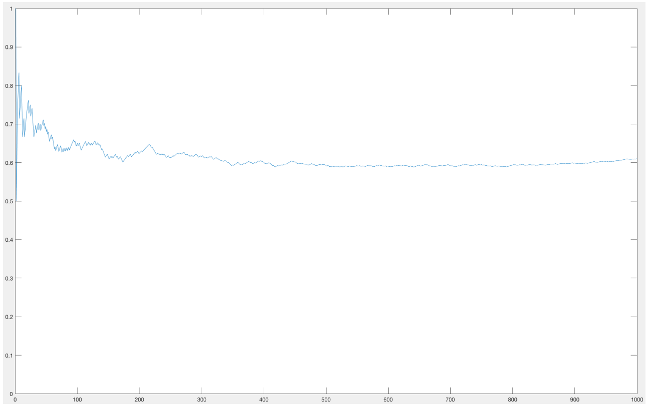
$$\pi(x) = C_{s_1}\cdot C_{s_2}\exp\left[\alpha x(y-z)^2+\beta(y-z)^2+\gamma x+\delta x\right]$$

$$\Rightarrow Z = \frac{1}{(1-a)(1-c)}\ ,\quad \boxed{\begin{aligned} H_\alpha(x) &= (y-z)^2x \\ H_\beta(x) &= (y-z)^2 \\ H_\gamma(x) &= x \\ H_\delta(x) &= x \end{aligned}}$$

e)

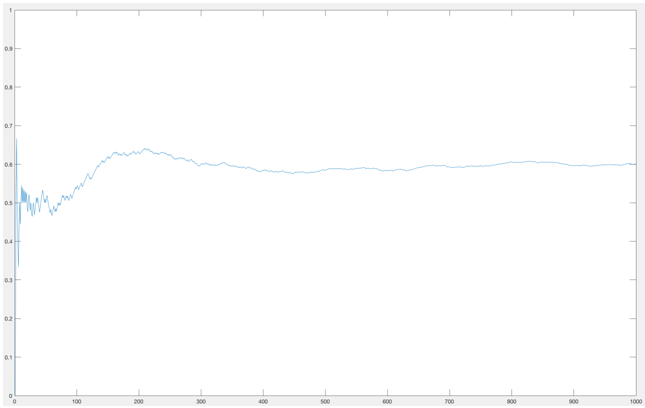
In here I implement the Gibbs Algorithm. For n = 5 I put the Probabilities and the plot of number of iterations versus average marginal probability of X.

0.6160	for n=2
0.5880	for n=3
0.6050	for n=4
0.6100	for n=5



Here the results and graphs for the n=10.

0.6060	for n=2
0.5870	for n=3
0.5910	for n=4
0.5990	for n=5
0.6220	for n=6
0.5850	for n=7
0.6100	for n=8
0.5920	for n=9
0.6000	for n=10



Graphs shape are changes because of I take the inputs randomly. Both graphs converges to same value with same as Q4a.

My Gibbs Sampling algorithm in Appendix Q4e.

Appendix

Q3.) *Algorithm*

```
function JointProb = JointProbability(Edges,Factors,x)

len = length(Edges(:,1));
num_of_nodes = max(Edges(:));

for i=1:num_of_nodes
    if any(Edges(:,2)==i) == 0
        JointProb(i) = Factors{i}(x(i)+1);
    else
        Parents = [];
        length_parents_factor = [];
        for j=1:len
            if i == Edges(j,2);
                Parents = [Parents Edges(j,1)];
                length_parents_factor =  [length_parents_factor length(Factors{Edges(j,1)}(1,:))];
            end
        end
        num_of_parents = length(Parents);
        for j=1:num_of_parents
            order = length(Factors{i}(:,1))/length_parents_factor(num_of_parents+1-j);
            Factors{i} = Factors{i}(order*x(Parents(num_of_parents+1-j))+1:order*(x(Parents(num_of_parents+1-
j))+1),:);
        end
        JointProb(i) = Factors{i}(x(i)+1);
        for k=1:num_of_parents
            JointProb(i) = JointProb(i)*JointProb(Parents(k));
        end
    end
end
```

Running code for non-loopy networks;

```
clear;
clc;
Edges = [1,3;2,3;2,4;3,5];
Factors{1} = [0.6,0.4];
Factors{2} = [0.7,0.3];
Factors{3} = [0.3 0.4 0.3; 0.05 0.25 0.7; 0.9 0.08 0.02; 0.5 0.3 0.2];
Factors{4} = [0.95,0.05;0.2,0.8];
Factors{5} = [0.1,0.9;0.4,0.6;0.99,0.01];
x = [0,1,2,0,1];
JointProb = JointProbability(Edges,Factors,x)
```

Running code for loopy networks;

```
clear;
clc;
Edges = [1,3;2,3;2,4;3,5;4,5];
Factors{1} = [0.6,0.4];
Factors{2} = [0.7,0.3];
Factors{3} = [0.3 0.4 0.3; 0.05 0.25 0.7; 0.9 0.08 0.02; 0.5 0.3 0.2];
Factors{4} = [0.95,0.05;0.2,0.8];
Factors{5} = [0.4,0.6;0.6,0.4;0.99,0.01;0.01,0.99;0.4,0.6;0.8,0.2];
x = [0,1,2,0,1];
JointProb = JointProbability(Edges,Factors,x)
```

Q4.)

a.)

```
clear;
clc;
a = 0.7;
b = 0.5;
c = 0.3;
n = 10;
G = zeros(n,n);
sample = 1000;

sum = zeros(n-1,1);
for index=1:sample
    for i=1:n
        for j=1:(n+1-i)
            G(i,j) = rand;
            if i == 1 && G(i,j) > 1-c
                G(i,j) = 1;
            elseif i == 1
                G(i,j) = 0;
            elseif G(i-1,j) == G(i-1,j+1) && G(i,j) > 1-a
                G(i,j) = 1;
            elseif G(i-1,j) ~= G(i-1,j+1) && G(i,j) > 1-b
                G(i,j) = 1;
            else
                G(i,j) = 0;
            end
        end
    end
    den = G(2:n,1);
    sum = sum + G(2:n,1);
    X(index) = sum(n-1) / index;
end
Probabilities = sum/sample
Range=[0 sample 0 1];
num_of_iter=1:sample;
figure(1)
plot(num_of_iter,X(num_of_iter));
axis(Range);
```

b.)

```
clear;
clc;
n = 10;
Message = [0.7 0.3];
Factors = [0.3 0.7; 0.5 0.5; 0.5 0.5; 0.3 0.7];
Coefficient = zeros(4,2);
for i=1:(n-1)
    row = 1;
    for j=1:2
        for k=1:2
            Coefficient(row,:) = Message(j)*Message(k);
            row = row + 1;
        end
    end
    Message = sum(Coefficient.*Factors);
    JointProbability(i) = Message(2);
end
JointProbability
```

e.)

```
clear;
clc;
a = 0.7;
b = 0.5;
c = 0.3;
n = 10;
G = zeros(n,n);
sample = 1000;

Cs = 1-a;
alfa = log(b*(1-a)/(a*(1-b)));
beta = log((1-b)/(1-a));
gama = log(a/(1-a));
sigma = log(c/(1-c));

sum = zeros(n-1,1);
X = zeros(sample,1);
for num_of_iter=1:sample
    for i=1:n
        for j=1:(n-i+1)
            G(i,j) = rand;
            if i==1
                Ps = [Cs Cs*exp(sigma*1)];
            else
                y = G(i-1,j);
                z = G(i-1,j+1);
                x = 1;
                Ps = [Cs*exp(beta*(y-z)^2) Cs*exp(alfa*x*(y-z)^2+beta*(y-z)^2+gama*x)];
            end
            PsX = Ps(2)/(Ps(1)+Ps(2));
            if G(i,j) < PsX
                G(i,j) = 1;
            else
                G(i,j) = 0;
            end
        end
    end
    sum = sum + G(2:n,1);
    X(num_of_iter) = sum(n-1)/num_of_iter;
end
Probabilities = sum/sample
Range=[0 sample 0 1];
num_of_iter=1:sample;
figure(1)
plot(num_of_iter,X(num_of_iter));
axis(Range);
```