Tutorial 8 - Belief Propagation and Sampling

COMP9418 – Advanced Topics in Statistical Machine Learning

Lecturer: Gustavo Batista

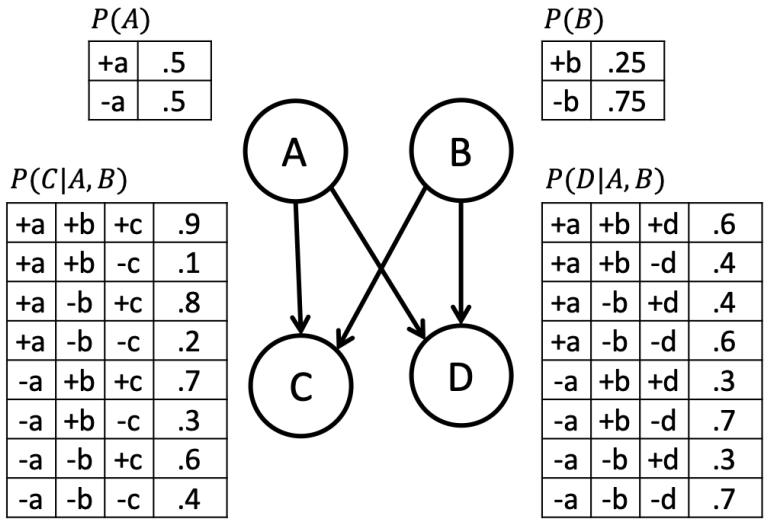
Lecture: Belief Propagation and Sampling

Topic: Questions from lecture topics

Last revision: Monday 23rd November, 2020 at 20:50

Question 1

Consider the following Bayesian network:



Suppose we condition on evidence $e : D = true$. Suppose that we have run the Parallel Iterative Belief Propagation (IBP) algorithm on the network, where it converges and yields the following set of messages and family marginals:

A	$\pi_C(A)$	$\pi_D(A)$	$\lambda_C(A)$	$\lambda_D(A)$
+a		.5	.5	.6
-a		.5	.5	.4

B	$\pi_C(B)$	$\pi_D(B)$	$\lambda_C(B)$	$\lambda_D(B)$
+b	.3	.25	.5	
-b	.7	.75	.5	

A	B	C	$\beta(A,B,C)$
+a	+b	+c	.162
+a	+b	-c	.018
+a	-b	+c	
+a	-b	-c	.084
-a	+b	+c	.084
-a	+b	-c	.036
-a	-b	+c	.168
-a	-b	-c	.112

A	B	D	$\beta(A,B,D)$
+a	+b	+d	.2
+a	+b	-d	0
+a	-b	+d	
+a	-b	-d	0
-a	+b	+d	.1
-a	+b	-d	0
-a	-b	+d	.3
-a	-b	-d	0

a. Fill in the missing values for IBP messages.
b. Fill in the missing values for family marginals.
c. Compute marginals $\beta(A)$ and $\beta(B)$ using the IBP messages and those computed in (a).
d. Compute marginals $\beta(A)$ and $\beta(B)$ by summing out the appropriate variables from the family marginals $\beta(ABC)$ as well as $\beta(ABD)$.
e. Compute joint marginal $\beta(AB)$ by summing out the appropriate variables from family marginals $\beta(ABC)$ as well as $\beta(ABD)$.
f. Are the marginals computed in (d) consistent? What about those computed in (e)?

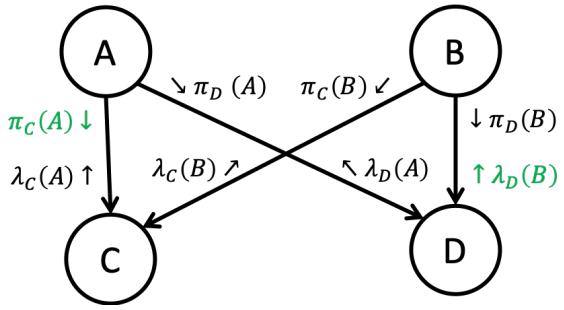
Consider the following IBP algorithm:

Data: N: Bayesian network
Data: **e**: evidence
Result: Approximate marginals, $\beta(X\mathbf{U})$ of $P(X\mathbf{U}|e)$ for each family $X\mathbf{U}$ in N

begin
  $t \leftarrow 0$;
  initialize all messages;
  while *messages have not converged* **do**
    $t \leftarrow t+1$;
    for *each node X with parents $\boldsymbol{U}$* **do**
      for *each parent U_i* **do**
        $\lambda_X^t(U_i) = \eta \sum_{X\mathbf{U}\setminus\{U_i\}} \lambda_{\mathbf{e}}(X)\phi_X(X,\mathbf{U}) \prod_{k\neq i} \pi_X^{t-1}(U_k) \prod_j \lambda_{Y_j}^{t-1}(X)$;
      end
      for *each child Y_j* **do**
        $\pi_{Y_j}^t(X) = \eta \sum_{\mathbf{U}} \lambda_{\mathbf{e}}(X)\phi_X(X,\mathbf{U}) \prod_i \pi_X^{t-1}(U_i) \prod_{k\neq j} \lambda_{Y_k}^{t-1}(X)$;
      end
    end
  end
end
return $\beta(X\mathbf{U}) = \eta\lambda_{\mathbf{e}}(X)\phi_X(X,\mathbf{U}) \prod_i \pi_X^t(U_i) \prod_j \lambda_{Y_j}^t(X)$

Answer

a. The following figure shows all messages:



Let's start computing $\pi_C(A)$. According to the IBP algorithm:

$\pi_C(A) = \eta\phi_A(A)\lambda_D(A)$

This simplified equation comes from the fact that node A has no parents and no evidence. Also, this message does not consider the message $\lambda_C(A)$ coming from node C because we are computing a message directed to the same node C.

Therefore,

A	$\pi_C(A)$
+a	.6
-a	.4

Remember that the multiplication by the constant η renormalizes the message $\pi_C(A)$ to sum to one.

Now, we compute $\lambda_D(B)$ using the equation from the IBP algorithm:

$\lambda_D(B) = \eta \sum_{A,B} \lambda_{\mathbf{e}}(D)\phi_D(D, A, B)\pi_D(A)$

Also, notice that node B has no children, and we should not consider the message $\pi_D(B)$ when computing a message to B.

We first compute the multiplication $\lambda_{\mathbf{e}}(D)\phi_D(D, A, B)\pi_D(A)$.

A	B	D	$\lambda_{\mathbf{e}}(D)\phi_D(D, A, B)\pi_D(A)$
+a	+b	+d	.3
+a	+b	-d	0
+a	-b	+d	.2
+a	-b	-d	0
-a	+b	+d	.15
-a	+b	-d	0
-a	-b	+d	.15
-a	-b	-d	0

After the elimination of variables A and B and renormalization, we get the message $\lambda_D(B)$

B	$\lambda_D(B)$
+b	.5625
-b	.4375

b. We also use the equation from IBP algorithm to compute the family marginals

$\beta(+a,-b,+c) = \eta\phi_C(+c,+a,-b)\pi_C(+a)\pi_C(-b) = \eta 0.8 \times 0.6 \times 0.7 = \eta 0.336$. Since this factor is natually normalized, $\eta = 1$.

$\beta(+a,-b,+d) = \eta\phi_D(+d,+a,-b)\pi_D(+a)\pi_D(-b) = \eta 0.4 \times 0.5 \times 0.75 = \eta 0.15$. This factor does not sum to one naturally since we have evidence on D. In this case $\eta \approx 2.667$ and $\beta(+a,-b,+d) = 0.4$.

c. According to the IBP algorithm:

$\beta(A) = \eta\phi_A(A)\lambda_C(A)\lambda_D(A)$. Therefore:

A	$\beta(A)$
+a	.6
-a	.4

Also, $\beta(B) = \eta\phi_B(B)\lambda_C(B)\lambda_D(B)$.

B	$\beta(B)$
+b	.3
-b	.7

d. We can compute $\beta(A)$ by summing out variables B and C from $\beta(A,B,C)$:

$\beta(A) = \sum_{B,C} \beta(A,B,C)$

A	B	C	$\beta(A,B,C)$
+a	+b	+c	.162
+a	+b	-c	.018
+a	-b	+c	.336
+a	-b	-c	.084
-a	+b	+c	.084
-a	+b	-c	.036
-a	-b	+c	.168
-a	-b	-c	.112

Summing out C, we get:

A	B	$\beta(A,B)$
+a	+b	.18
+a	-b	.42
-a	+b	.12
-a	-b	.28

Summing out B, we get:

A	$\beta(A)$
+a	.6
-a	.4

We can also compute $\beta(B)$ from $\beta(A, B)$ eliminating A

B	$\beta(B)$
+b	.3
-b	.7

We will now sum out variables from the family marginal $\beta(A, B, D)$. We can compute $\beta(A)$ by summing out variables B and D from $\beta(A, B, D)$:

A	B	D	$\beta(A, B, D)$
+a	+b	+d	.2
+a	+b	-d	0
+a	-b	+d	.4
+a	-b	-d	0
-a	+b	+d	.1
-a	+b	-d	0
-a	-b	+d	.3
-a	-b	-d	0

Summing out D, we get:

A	B	$\beta(A, B)$
+a	+b	.2
+a	-b	.4
-a	+b	.1
-a	-b	.3

Summing out B, we get:

A	$\beta(A)$
+a	.6
-a	.4

We can also compute $\beta(B)$ from $\beta(A, B)$ eliminating A

B	$\beta(B)$
+b	.3
-b	.7

e. We did this as an intermediate step of the previous answer. We copy the answer above to facilitate reference.

A	B	$\beta(A, B)$ from $\beta(A, B, C)$
+a	+b	.18
+a	-b	.42
-a	+b	.12

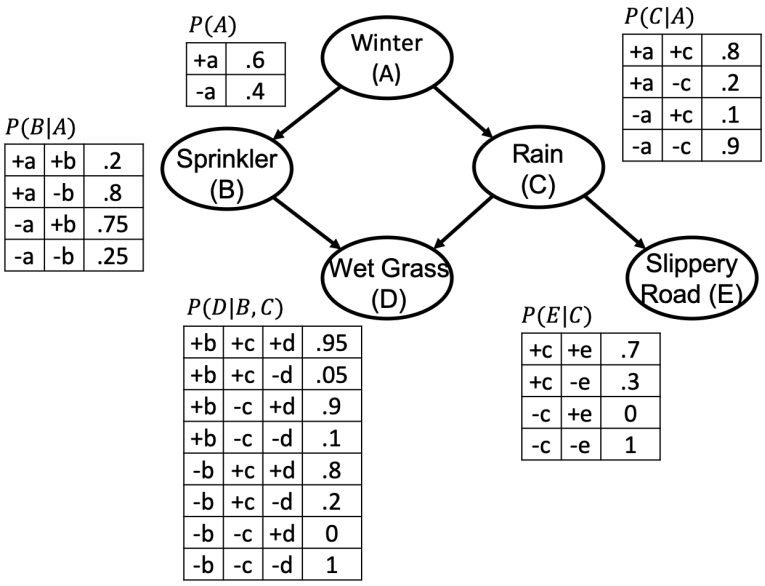
A	B	$\beta(A,B)$ from $\beta(A,B,C)$
-a	-b	.28

A	B	$\beta(A,B)$ from $\beta(A,B,D)$
+a	+b	.2
+a	-b	.4
-a	+b	.1
-a	-b	.3

f. The marginals computed in (d) are consistent, while the ones in computed in (e) are not consistent.

Question 2

Consider the following Bayesian network:



and the parameter $\theta_{\bar{a}}$ representing the probability of $A = false$ (i.e., it is not winter). For each of the following values of this parameter, 0.01, 0.4 and 0.99 do the following:

a. Compute the probability of $P(+d, +e)$: wet grass and slippery road. You can use the code from previous tutorials.
b. Estimate $P(+d, +e)$ using forward sampling with sample sizes ranging from $n = 100$ to $n = 15,000$.
c. Generate a plot with n on the x-axis and the exact value of $P(+d, +e)$ and the estimate for $P(+d, +e)$ on the y-axis.
d. Generate a plot with n on the x-axis and the exact variance of the estimate for $P(+d, +e)$ and the sample variance on the y-axis.

Answer

a. We computed $P(d, e)$ using our code from previous tutorials. We obtained the following results: $P(+d, +e) = 0.4608$ for $\theta_{\bar{a}} = 0.01$ $P(+d, +e) = 0.3044$ for $\theta_{\bar{a}} = 0.4$ $P(+d, +e) = 0.0679$ for $\theta_{\bar{a}} = 0.99$

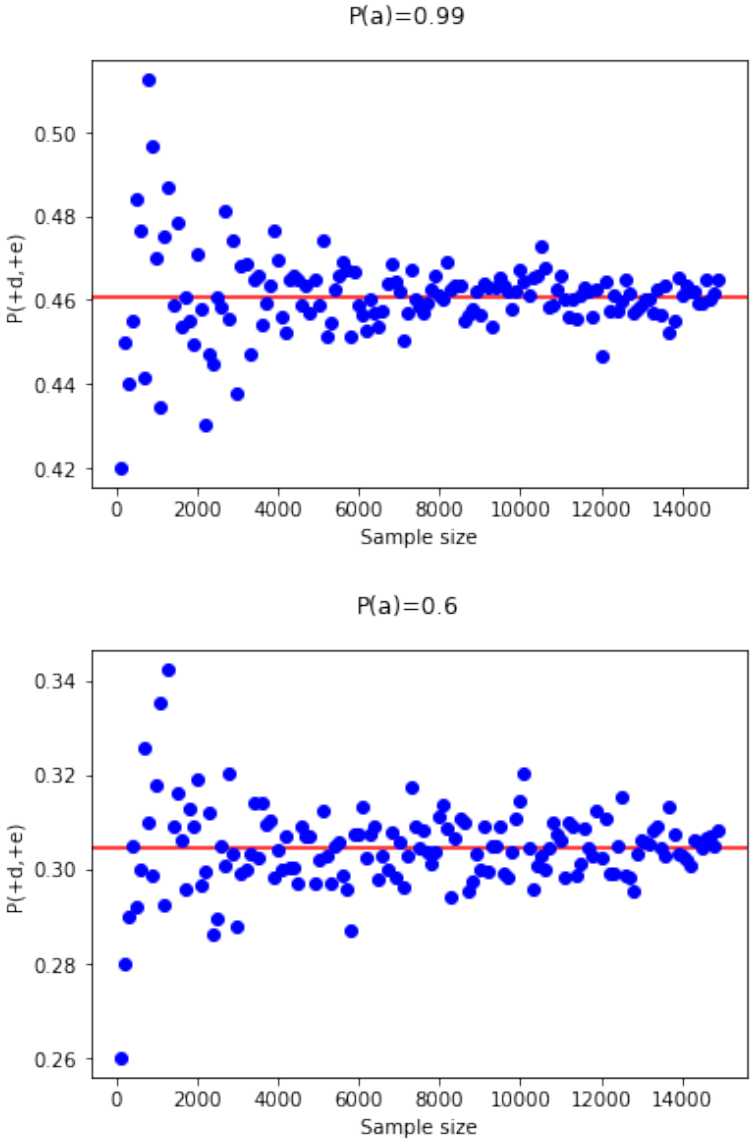
b. The following code will estimate $P(+d, +e)$ using forward sampling:

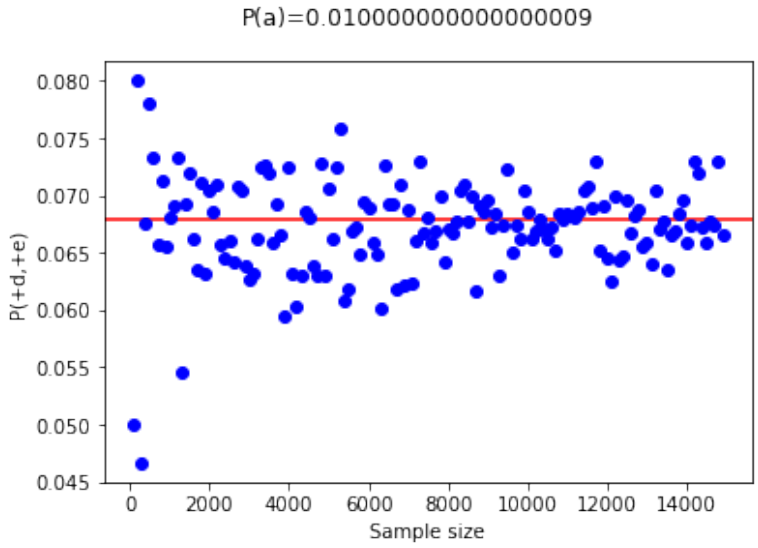
```
from collections import OrderedDict as odict
from random import random
import matplotlib.pyplot as plt
import numpy as np


factors = {
    'A': {
        'dom': ('H'),
        'table': odict([
            ((0,), 0.40),
            ((1,), 0.60),
        ])
    },

    'D': {
        'dom': ('B', 'C', 'D'),
        'table': odict([
            ((0, 0, 0), 1),
            ((0, 0, 1), 0),
            ((0, 1, 0), .2),
            ((0, 1, 1), .8),
            ((1, 0, 0), .1),
            ((1, 0, 1), .9),
            ((1, 1, 0), .05),
            ((1, 1, 1), .95),
        ])
    },

    'B' : {
        'dom': ('A', 'B'),
        'table': odict([
            ((0, 0), 0.25),
            ((0, 1), 0.75),
            ((1, 0), 0.8),
            ((1, 1), 0.2),

        ])
    },

    'E' : {
        'dom': ('C', 'E'),
        'table': odict([
            ((0, 0), 1),
            ((0, 1), 0),
            ((1, 0), .3),
            ((1, 1), .7),

        ])
    },

    'C' : {
        'dom': ('A', 'C'),
```

```
        'table': odict([
            ((0, 0), .9),
            ((0, 1), .1),
            ((1, 0), .2),
            ((1, 1), .8),

        ])
    },
}

# Declare `factors`: a factor dictionary for the Bayesian network parameters
theta_na_list = [.01, .4, .99]
true_p_d_e_list = [0.4608, 0.3044, 0.0679]
sample_sizes = range(100,15000,100)
for j, theta_na in enumerate(theta_na_list):
    p_d_e_list = []
    sample_var_list = []
    sample_mean_var_list = []
    factors['A']['table'][1] = 1 - theta_na
    for s in sample_sizes:
        repetitions = []
        for _ in range(20):
            d_e_count = 0
            for i in range(s):
                a = random() < factors['A']['table'][1]
                b = random() < factors['B']['table'][(a,1)]
                c = random() < factors['C']['table'][(a,1)]
                d = random() < factors['D']['table'][(b,c,1)]
                e = random() < factors['E']['table'][(c,1)]
                if d and e:
                    d_e_count = d_e_count + 1

            p_d_e = d_e_count / s
            repetitions.append(p_d_e)
        p_d_e_list.append(p_d_e)
        sample_var_list.append(np.var(repetitions))
        sample_mean_var_list.append(true_p_d_e_list[j] * (1-true_p_d_e_list[j])/s)

    plt.figure()
    plt.axhline(y=true_p_d_e_list[j], color='r', linestyle='-')
    plt.suptitle('P(a)='+str(1-theta_na))
    plt.plot(range(100,15000,100), p_d_e_list, 'bo')
    plt.xlabel("Sample size")
    plt.ylabel("P(+d,+e)")
    plt.show()

    plt.figure()
    plt.suptitle('P(a)='+str(1-theta_na))
    plt.plot(range(100,15000,100), sample_var_list, 'bo')
    plt.plot(range(100,15000,100), sample_mean_var_list, 'k')

    plt.xlabel("Sample size")
    plt.ylabel("Sample variance")
```

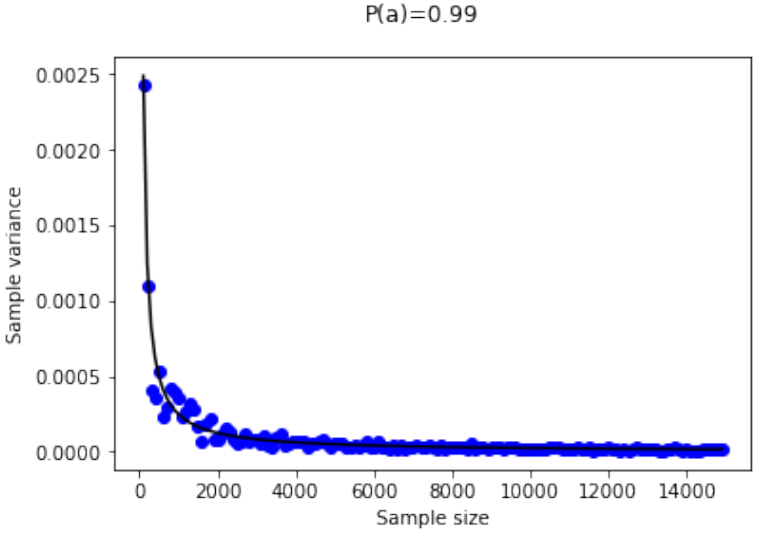
```
plt.show()
```

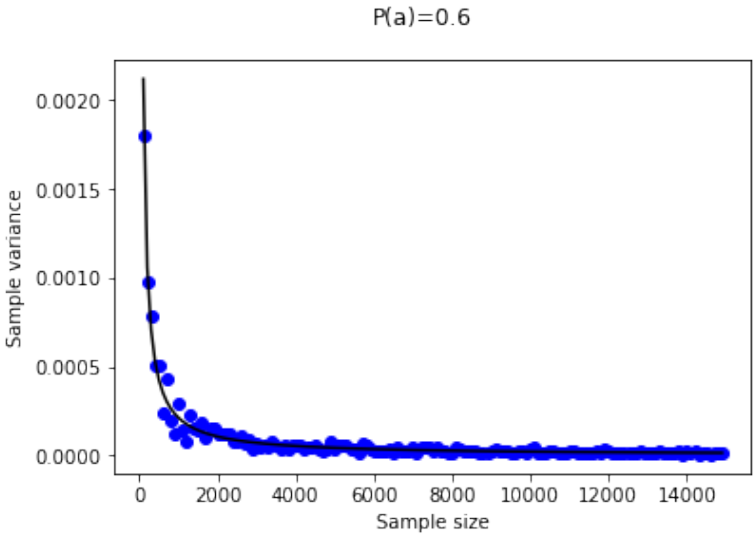
c. You should get a plot like this, with the above code

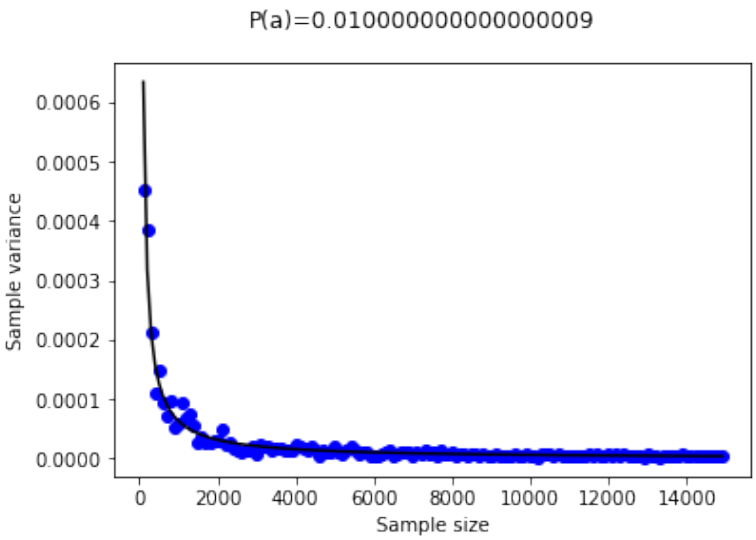




d. Keep in mind that although the variance of the estimate decreases at a rate of $O(1/n)$ (as shown in the figures), the standard deviation of the estimate decreases at a rate of $O(\frac{1}{\sqrt{n}})$. The standard deviation of the estimate (usually called standard error) is what we would use if we wanted to know in advance how accurate we can expect our estimate to be for a given number of samples. Using this information we can make a trade off between the speed of the algorithm and the quality of the estimate.




P(a)=0.6
Sample variance
0.0020
0.0015
0.0010
0.0005
0.0000
0
2000
4000
6000
8000
10000
12000
14000
Sample size


P(a)=0.010000000000000009
Sample variance
0.0006
0.0005
0.0004
0.0003
0.0002
0.0001
0.0000
0
2000
4000
6000
8000
10000
12000
14000
Sample size