CS3105 P2 - 200007626

Question 1

1. $p_0 = 0.48$ as $1 - 0.003 - 0.027 - 0.034 - 0.136 - 0.02 - 0.18 - 0.12 = p_0 = 0.48$
2. Verifying independence claims
    a. Are A, B marginally independent?

    If A and B are marginally independent, then $P(A, B) = P(A)P(B)$

    $P(A) = \sum_b \sum_c P(A, b, c)$ - therefore,

    - 0.02 + 0.18 + 0.12 + 0.48, A = 1
    - 0.003 + 0.027 + 0.034 + 0.136, A = 0

    $P(A = 1) = 0.8$

    $P(A = 0) = 0.2$

    $P(B) = \sum_a \sum_c P(a, B, c)$ - therefore,

    - 0.034 + 0.136 + 0.12 + 0.48, B = 1
    - 0.003 + 0.027 + 0.02 + 0.18, B = 0

    $P(B = 1) = 0.77$

    $P(B = 0) = 0.23$

    $P(A, B) = \sum_c P(A, B, c)$, therefore,

    - 0.003 + 0.027, A = 0, B = 0
    - 0.02 + 0.18, A = 1, B = 0
    - 0.034 + 0.136, A = 0, B = 1
    - 0.12 + 0.48, A = 1, B = 1

$P(A = 0, B = 0) = 0.03$

$P(A = 1, B = 0) = 0.2$

$P(A = 0, B = 1) = 0.17$

$P(A = 1, B = 1) = 0.6$

Testing for marginal independence:

$P(A = 1, B = 1) = 0.6$

$P(A = 1) = 0.8$

$P(B = 1) = 0.77$

$P(A = 1)P(B = 1) = 0.616 \neq P(A = 1, B = 1) = 0.6$

Therefore, **not marginally independent.**

---

b. Are A, C conditionally independent given B?

Question formulated: $P(A, C \mid B)$

If conditionally independent, then $P(A, C \mid B) = P(A \mid B)\ P(C \mid B)$

$P(A \mid B) = P(A, B)/P(B)$, therefore

$P(A = 0 \mid B = 0) = 0.13$

$P(A = 1 \mid B = 0) = 0.87$

$P(A = 0 \mid B = 1) = 0.22$

$P(A = 1 \mid B = 1) = 0.78$

$P(C \mid B) = P(C, B)/P(B)$, therefore

$$P(C=0 \mid B=0) = 0.1$$
$$P(C=1 \mid B=0) = 0.9$$
$$P(C=0 \mid B=1) = 0.2$$
$$P(C=1 \mid B=1) = 0.8$$

Checking B = 0:

- $P(A=1, C=1|B=0) = 0.18/0.23 = 0.78 = P(A|B)P(C|B)$
- $P(A=1, C=0|B=0) = 0.02/0.23 = 0.087 = P(A|B)P(C|B)$
- $P(A=0, C=1|B=0) = 0.027/0.23 = 0.12 = P(A|B)P(C|B)$
- $P(A=0, C=0|B=0) = 0.003/0.23 = 0.013 = P(A|B)P(C|B)$

These sum to 1 - correct!

Checking B = 1:

- $P(A=1, C=1|B=1) = 0.48/0.77 = 0.62 = P(A|B)P(C|B)$
- $P(A=1, C=0|B=1) = 0.12/0.77 = 0.16 = P(A|B)P(C|B)$
- $P(A=0, C=1|B=1) = 0.136/0.77 = 0.18 = P(A|B)P(C|B)$
- $P(A=0, C=0|B=1) = 0.034/0.77 = 0.04 = P(A|B)P(C|B)$

These also sum to 1 - correct!

Since $P(A, C \mid B) = P(A|B)P(C|B)$ holds for all possibilities, A, C **are conditionally independent** given B.

---

3. Provide the corresponding CPFs for the directed graph.

A-node : $P(A) = \sum_b \sum_c P(A, b, c)$

P(A)	0.8

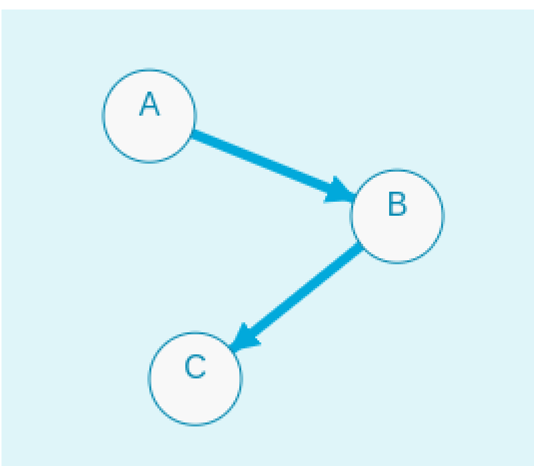
B-node: $P(B \mid A) = P(A, B) \ / \ P(A)$ =

A	P(B | A)
1	0.75
0	0.85

C-node: $P(C \mid A, B) = P(A, B, C) \ / \ P(A, B)$

A	B	P(C | A,B)
1	1	0.8
0	1	0.8
1	0	0.9
0	0	0.9

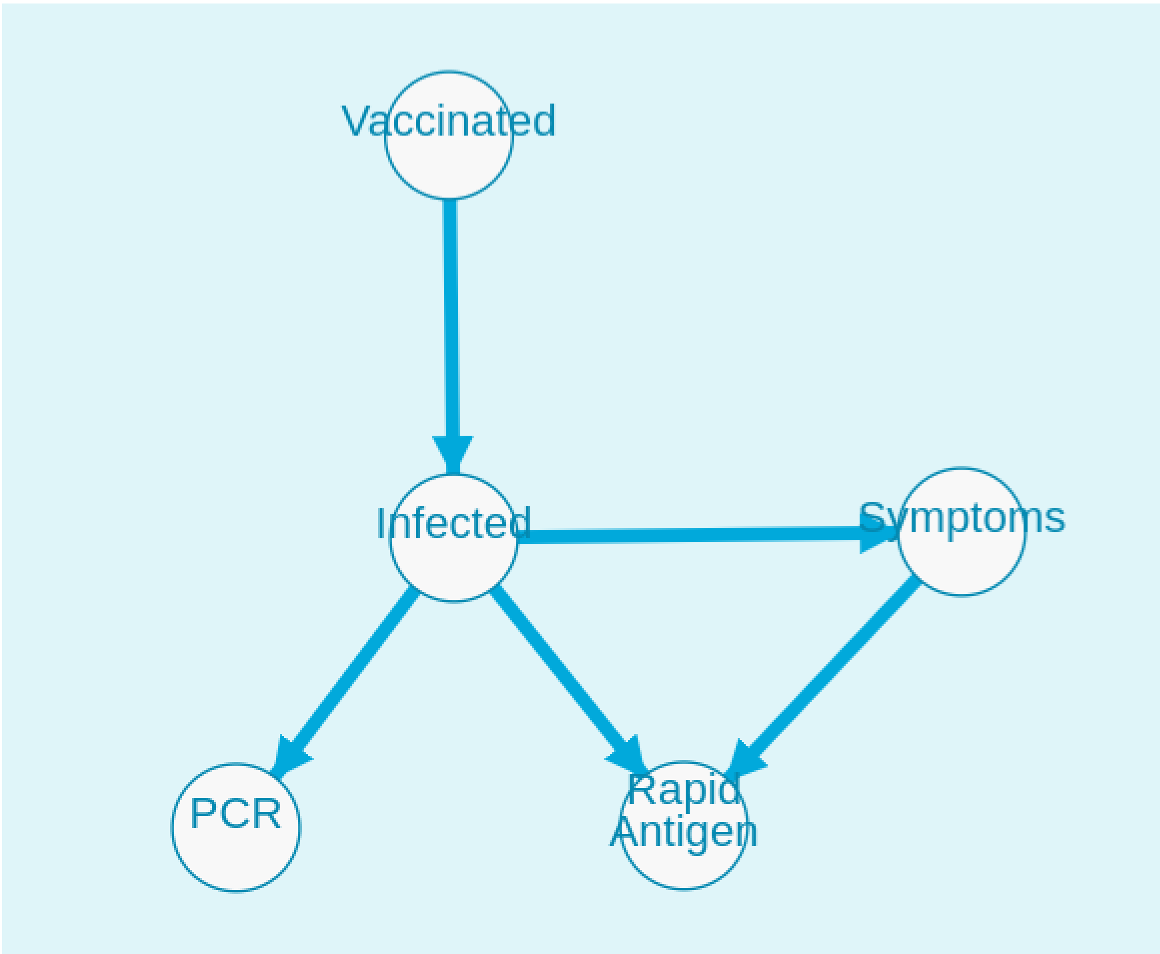
4. Can the Bayesian network be simplified?



Node C depends on Node A and Node B - $P(C) = P(C \mid A, B)$. Since $P(B) = P(B \mid A)$, and we get to Node B via Node A, then the full joint is reachable; $P(B \mid A)P(A) = P(A, B)$ and $P(C \mid A, B)P(A, B) = P(A, B, C)$.

The CPFs do not change.

Question 2



P(Vaccinated=1)	P(Vaccinated=0)
0.9	0.1

Vaccinated	P(Infected=1|Vaccinated)	P(Infected=0|Vaccinated)
1	0.05	0.95
0	0.22	0.78

Infected	P(Symptoms=1|Infected)	P(Symptoms=0|Infected)
1	0.66	0.34
0	0	1

Infected	P(PCR=1|Infected)	P(PCR=0|Infected)
1	0.982	0.018
0	0.008	0.992

Infected	Symptoms	P(Rapid Antigen=1|Infected,Symptoms)	P(Rapid Antigen=0|Infected,Symptoms)
1	1	0.8	0.2
1	0	0.58	0.42
0	1	0.08	0.92
0	0	0.1	0.9

Brian has tested positive with a rapid antigen test, and negative with a PCR test. How likely is it that Brian is infected with Flu-22?

$$P(infected) = P(infected \mid RAT = 1, PCR = 0)$$

$$= \alpha P(infected, RAT = 1, PCR = 0)$$

$$= \sum_v \sum_s \alpha P(v,\ infected,\ s,\ RAT = 1,\ PCR = 1)$$

Probability that Brian is infected:

$v = 0, s = 0$: 0.1 * 0.22 * 0.34 * 0.018 * 0.58

$v = 0, s = 1$: 0.1 * 0.22 * 0.66 * 0.018 * 0.8

$v = 1, s = 0$: 0.9 * 0.05 * 0.34 * 0.018 * 0.58

$v = 1, s = 1$: 0.9 * 0.05 * 0.66 * 0.018 * 0.8

Sum = 0.0008745912

Probability that Brian is not infected:

$v = 0, s = 0$: 0.1 * 0.78 * 1 * 0.992 * 0.1

$v = 0, s = 1$: 0.1 * 0.78 * 0 * … = 0

$v = 1, s = 0$: 0.9 * 0.95 * 1 * 0.992 * 0.1

$v = 1, s = 1$: 0 since no symptoms

Sum = 0.0925536

Therefore $\alpha = 0.0008745912 + 0.0925536 = 0.0934281912$

The probability that Brian is infected is therefore $0.0008745912/0.0934281912 = 0.00936110599$

$$P(infected = 1 \mid RAT = 1, PCR = 0) \approx 0.0094$$

Alex, having been vaccinated, has tested positive with a rapid antigen test. How likely is it that Alex is going to get a positive PCR result if he takes the test?

$P(PCR) = P(PCR \mid vaccinated = 1, RAT = 1)$

$= \alpha P(vaccinated = 1, RAT = 1, PCR)$

$= \sum_s \sum_i \alpha P(vaccinated = 1, i, s, RAT = 1, PCR)$

Probability Alex will get a positive PCR result:

$i = 0, s = 0$: 0.9 * 0.95 * 1 * 0.008 * 0.1

$i = 0, s = 1$: 0.9 * 0.95 * 0 * … = 0

$i = 1, s = 0$: 0.9 * 0.05 * 0.34 * 0.982 * 0.58

$i = 1, s = 1$: 0.9 * 0.05 * 0.66 * 0.982 * 0.8

Sum = 0.032730588

Probability Alex will get a negative PCR result:

$i = 0, s = 0$: 0.9 * 0.95 * 1 * 0.992 * 0.1

$i = 0, s = 1$: 0

$i = 1, s = 0$: 0.9 * 0.05 * 0.34 * 0.018 * 0.58

$i = 1, s = 1$: 0.9 * 0.05 * 0.66 * 0.018 * 0.8

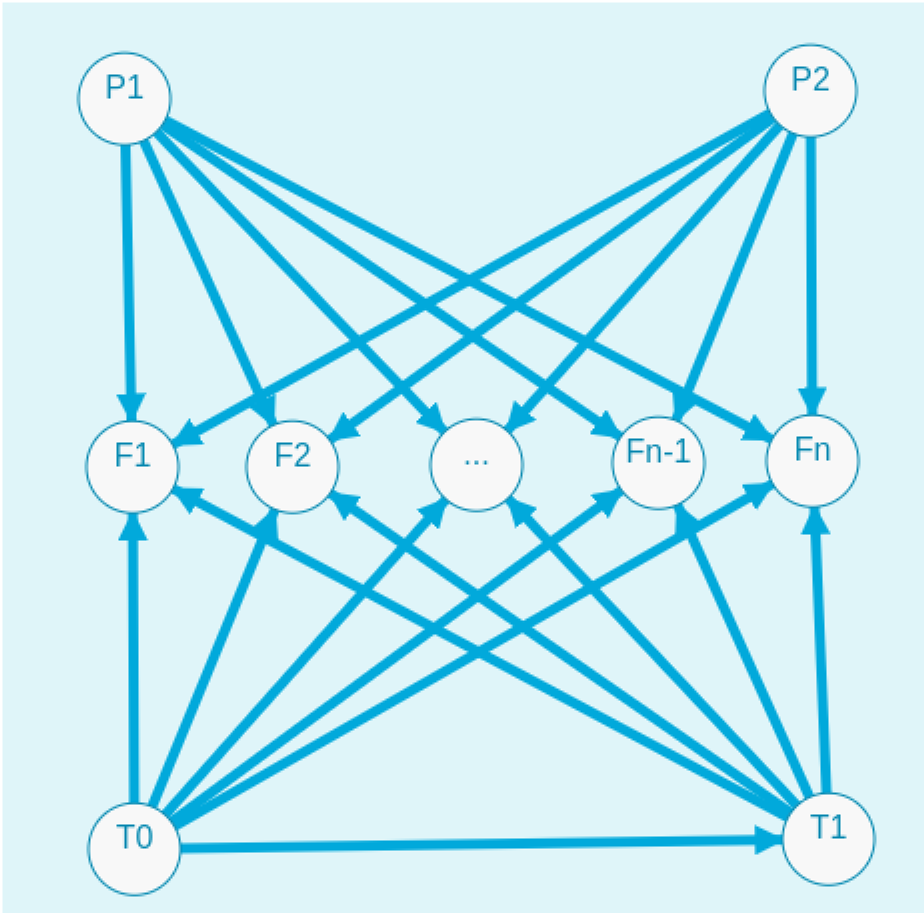
Sum = 0.085403412

Therefore α is $0.032730588 + 0.085403412 = 0.118134$

The probability that Alex will get a positive PCR result is $0.032730588/0.118134 = 0.277063233$

$P(PCR = 1 \mid vaccinated = 1, RAT = 1) \approx 0.28$

Question 3



$P(P1) = 1/101$ if P1 in {0.00, 0.01, …, 1.00}; $P(P1) = 0$ otherwise

$P(P2) = 1/101$ if P2 in {0.00, 0.01, …, 1.00}; $P(P2) = 0$ otherwise

$P(T0) = 1/(n-2)$ if T0 in {1, 2, …, n-2}; $P(T0) = 0$ otherwise

$P(t1) = 1/(n - T0 - 1)$ if T1 in {T0+1, …, n-1}; $P(T1) = 0$ otherwise

$P(Fi \mid T0, T1, P1, P2) =$

- $P1 : Fi = 1, i \leq T0 \ or \ i \ > T1$
- $1 - P1 : Fi = 0, i \leq T0 \ or \ i > \ T1$
- $P2 : Fi = 1, T0 < i \leq T1$
- $1 - P2 : Fi = 0, T0 < i \leq T1$

- 0 : otherwise

Using the exact inference algorithm, $P(P1) = P(P1 \mid T0, T1, F1, ..., Fn)$

The time complexity for this algorithm is 2^n, as with any exact inference, where n is the number of nodes. Since exact inference already becomes unreasonable for networks with the number of nodes greater than twenty-five (referenced from lectures), this algorithm is is scalable for multiple switches up to a point - it would greatly depend on how many coin toss results are already contained within the network.

Outline of the Gibbs Sampling algorithm in pseudo-code below

```
D = data
DL = length of data
burn-in period = DL / 2

turning point 1 = initial guess
turning point 2 = initial guess
coin1 bias = initial guess
coin2 bias = initial guess
possible coin biases = [0.00, 0.01, ..., 0.99, 1.00]

results = empty array

for i in (DL*10 + burn-in):
sample turning point 1 using data, coin1 bias, and coin2 bias
sample turning point 2 using turning point 1, data, coin1 bias, and coin2 bias

sample coin1 bias based off the data in the appropriate regions defined by
  the turning points

sample coin2 bias based off the data in the appropriate regions defined by
  the turning points

if i > burn-in:
add turning point 1, turning point 2, coin1 bias, and coin2 bias to the
  results array
end

end
```

This algorithm is scalable for multiple switches (up to a reasonable number), as you can always sample another turning point which is positioned based off the previous ones.

Advanced

A large portion of the code within my jupyter notebook has translated, studied, and adapted from the tutorial lecture slides on studres. I have provided comments above the functions in my notebook to show understanding of what they do, and why they are there.

After inspecting and understanding the coin toss Julia code from the tutorial into Python, I saw that there would be few alterations required to allow calculation of two switching points. After guessing the initial values for the two swaps and the two biases of the coins, I find the the most likely first switching point based on the data. Once this is found, I limit the data to after the first turning point, and then perform the same process again, only this time, with the probabilities reversed. (i.e., before the coin switch the probability is p2, and after, p1).

When calculating the biases of the coins, I limit the samples from the data to everything before the first switch and after the second for the coin 1; likewise, coin 2's bias is calculated using data between the two swaps.

I found the most likely turning points to be at the 109th toss, and the 215th toss , with the first coin's bias towards heads 0.35, and the second coin's bias 0.78.

To deal with missing data, I replaced the unknown values with either a 1 or a 0, chosen using the relevant probability depending upon where the values lie with respect to the coin switches. For example, if there were to be a missing value before the first coin swap, it would be replaced with either a 1 or a 0, with a probability of p1 that it will be a 1, and a 1-p1 probability that it will be a zero. Of course, initially I cannot use existing probabilities to calculate what the most likely missing values would be, and so I simply replace the missing values with randomly chosen 1s and 0s. I concede this is a sub-optimal solution, and given more time I would calculate the probability of heads and tails respecting the guessed coin switch points while ignoring the missing values, and then replace the missing values using these probabilities initially.

I have demonstrated the efficacy of my implementation in my jupyter notebook - removing 15 concurrent values from the document (5%) only changes the predicted biases of the coins by 0.01, and did not affect the suggested location of the turning points at all!