Introduction to Probabilistic Graphical Models **VU Thi Hai Yen**
Prof. Umut Simsekli haiyen96.hp@gmail.com

Practical Session 1

PROBLEM 1:

In many practical applications, we often need to compute $s = \log \sum_{i=1}^{I} exp(v_i)$, where each $v_i < 0$ and $|v_i|$ is very large. Derive (mathematically) and implement a numerically stable algorithm for computing log(sum(exp(v))), where $v = (v_i)_{i=1}^{I}$ is a vector of numbers. Explain why it should work.
Test your algorithm on $log(sum(exp{-1234}, -1235))$.

SOLUTION:
For an arbitrary a, we have:

$$s = \log \sum_{i=1}^{I} exp(v_i) = \log \left[\exp(-a) \left(\sum_{i=1}^{I} exp(v_i) \right) \right] - \log e^{-a} = \left[\log \sum_{i=1}^{I} exp(v_i - a) \right] + a$$

This means, you can shift the center of the exponential sum.
In order to force the greatest value to be zero and make the other values under control, we will choose:

$$a = \max_i v_i.$$

This is a good choice as even for large values of $|v_i|$, we always shift the biggest exponential component to $\exp(0) = 1$, and the other ones can be computed numerically and relatively stable to this max value (as we are computing the exponential of negative numbers).

PROBLEM 2:

A robot is moving across a circular corridor. We assume that the possible positions of the robot is a discrete set with N locations. The initial position of the robot is unknown and assumed to be uniformly distributed. At each step k, the robot stays where it is with probability ϵ, or moves to the next point in counter-clock direction with probability $1-\epsilon$. At each step k, the robot can observe its true position with probability w. With probability $1-w$, the position sensor fails and gives a measurement that is independent from the true position (uniformly distributed).

1. Choose the appropriate random variables, define their domains, write down the generative model and draw the associated directed graphical model.

SOLUTION:

Fix a point in the discrete set with N locations, we denote it by 0. From the point 0, the next points in counter-clock direction are denoted by 1, 2, ..., N-1 respectively.

Let X_k be the true position of the robot at time step k.

The domain of X_k ($k \in \{1,2,3,...,\}$) is: $\{0,1,2,3,...,N-1\}$.

Let Y_k be the position which the position sensor shows at time step k.

The domain of Y_k ($k \in \{1,2,3,...,\}$) is: $\{0,1,2,3,...,N-1\}$.

The generative model estimates the distribution $\mathbb{P}(X|Y)$ - the probability of observing X.

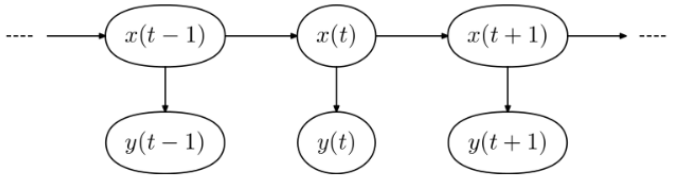

Figure 1: The associated directed graphical model

2. Define the conditional probability tables (i.e., the transition model and the observation model) given the verbal description above.
SOLUTION:
We write down the transition probabilities of the hidden model and the observation as follows:

$$\mathbb{P}(X_{k+1} = j | X_k = i) = \begin{cases} \epsilon & \text{if } j = i \\ 1 - \epsilon & \text{if } j = i + 1 \\ 0 & \text{otherwise} \end{cases},$$

$$\mathbb{P}(Y_k = j | X_k = i) = \begin{cases} w + \dfrac{1 - w}{n} & \text{if } j = i \\ \dfrac{1 - w}{n} & \text{otherwise} \end{cases}.$$

Hence we obtain the matrix for the transition model is

$$T = \epsilon I_{N \times N} + (1 - \epsilon) K_{N \times N}$$

where K is the matrix obtained by permutating all rows from I 1 increment to the right.

The matrix for the observation model is

$$B = w I_{N \times N} + \frac{1 - w}{n} L_{N \times N}$$

where L is the matric whose all coefficients are 1.

For example: $n = 3$ we have

$$I = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, K = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}, L = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}.$$

3. Specify the following verbal statements in terms of posterior quantities using mathematical notation.

SOLUTION:

Let n denote the current position of the robot.

(a) Distribution of the robots current position given the observations so far:

$$\mathbb{P}(X_n|Y_1, Y_2, ..., Y_n),$$

(b) Distribution of the robots position at time step k given all the observations:

$$\mathbb{P}(X_k|Y_1, Y_2, ..., Y_n),$$

(c) Distribution of the robots next position given the observations so far:

$$\mathbb{P}(X_{n+1}|Y_1, Y_2, ..., Y_n),$$

(d) Distribution of the robots next sensor reading given the observations so far:

$$\mathbb{P}(Y_{n+1}|Y_1, Y_2, ..., Y_n),$$

(e) Distribution of the robots initial position given observations so far:

$$\mathbb{P}(X_1|Y_1, Y_2, ..., Y_n),$$

(f) Most likely current position of the robot given the observations so far:

$$argmax_{\{1,2,...,N-1\}}\mathbb{P}(X_n|Y_1, Y_2, ..., Y_n),$$

(g) Most likely trajectory taken by the robot from the start until now given the observations so far:

$$argmax_{X_i \in \{1,2,...,N-1\}}\mathbb{P}((X_1, X_2, ..., X_n)|Y_1, Y_2, ..., Y_n)$$

4. Implement a program that simulates this scenario; i.e., generates realizations from the movements of the robot and the associated sensor readings. You can use the randgen function you wrote earlier. Simulate a scenario for k = 1, . . . , 100 with $N = 50, \epsilon = 0.3, w = 0.8$.

SOLUTION:
See the `generate()` function.

5. Implement the Forward-Backward algorithm for computing the quantities defined in 3-a,b,f.
SOLUTION:
The Forward-Backward algorithm aims to compute the posterior distribution of a hidden Markov variable at time step k given all the observations:

$$\mathbb{P}(X_k|Y_1, Y_2, \dots, Y_n).$$

To compute this value, we first decompose it using Bayes' rule and the fact that $Y_1, \dots, Y_k \coprod Y_{k+1}, \dots, Y_n | X_k$ and obtain:

$$\begin{aligned} \mathbb{P}(X_k|Y_1, Y_2, \dots, Y_n) &= \frac{\mathbb{P}(X_k, Y_1, \dots, Y_n)}{\mathbb{P}(Y_1, \dots, Y_n)} \\ &= \frac{\mathbb{P}(Y_1, \dots, Y_n|X_k)\mathbb{P}(X_k)}{\mathbb{P}(Y_1, \dots, Y_n)} \\ &= \frac{\mathbb{P}(Y_{k+1}, \dots, Y_n|X_k)\mathbb{P}(Y_1, \dots, Y_k|X_k)\mathbb{P}(X_k)}{\mathbb{P}(Y_1, \dots, Y_n)} \\ &= \frac{\mathbb{P}(Y_{k+1}, \dots, Y_n|X_k)\mathbb{P}(Y_1, \dots, Y_k, X_k)}{\mathbb{P}(Y_1, \dots, Y_n)} \end{aligned} \tag{1}$$

We now put:

$$\gamma_k(X_k) = \mathbb{P}(X_k|Y_1, Y_2, \dots, Y_n), \alpha_k(X_k) = \mathbb{P}(Y_1, \dots, Y_k, X_k), \beta_k(X_k) = \mathbb{P}(Y_{k+1}, \dots, Y_n|X_k).$$

To compute $\alpha_k(X_k)$, we derive as:

$$\begin{aligned} \alpha_k(X_k) &= \mathbb{P}(Y_1, \dots, Y_k, X_k) \\ &= \mathbb{P}(Y_1, \dots, Y_k|X_k)\mathbb{P}(X_k) \\ &= \mathbb{P}(Y_1, \dots, Y_{k-1}|X_k)\mathbb{P}(Y_k|X_k)\mathbb{P}(X_k) \quad (\text{as } Y_1, \dots, Y_{k-1} \coprod Y_k|X_k) \\ &= \mathbb{P}(Y_k|X_k)\mathbb{P}(Y_1, \dots, Y_{k-1}, X_k) \\ &= \mathbb{P}(Y_k|X_k) \sum_{X_{k-1}} \mathbb{P}(Y_1, \dots, Y_{k-1}, X_{k-1}, X_k) \\ &= \mathbb{P}(Y_k|X_k) \sum_{X_{k-1}} \mathbb{P}(X_k|Y_1, \dots, Y_{k-1}, X_{k-1})\mathbb{P}(Y_1, \dots, Y_{k-1}, X_{k-1}) \\ &= \mathbb{P}(Y_k|X_k) \sum_{X_{k-1}} \mathbb{P}(X_k|X_{k-1})\alpha_{k-1}(X_{k-1}) \quad (\text{Markov's property}). \end{aligned} \tag{2}$$

If we write $\boldsymbol{\alpha_k}$ as a vector $(\alpha_k(0), \alpha_k(1), \dots, \alpha_k(n-1))^T$, we obtain that

$$\boldsymbol{\alpha_k} = B_{:,Y_k} \odot (T\boldsymbol{\alpha_{k-1}}),$$

where $\odot$ denotes the element-wise product. This is called the **forward** pass.

To compute $\beta_k(X_k)$, we derive as:

$$\begin{aligned}
\beta_k(X_k) &= \mathbb{P}(Y_{k+1}, \dots, Y_n | X_k) \\
&= \sum_{X_{k+1}} \mathbb{P}(Y_{k+1}, \dots, Y_n, X_{k+1} | X_k) \\
&= \sum_{X_{k+1}} \mathbb{P}(Y_{k+1}, \dots, Y_n | X_{k+1}, X_k) \mathbb{P}(X_{k+1} | X_k) \\
&= \sum_{X_{k+1}} \mathbb{P}(Y_{k+1}, \dots, Y_n | X_{k+1}) \mathbb{P}(X_{k+1} | X_k) \quad \text{(Markov's property)} \\
&= \sum_{X_{k+1}} \mathbb{P}(Y_{k+1} | X_{k+1}) \mathbb{P}(Y_{k+2}, \dots, Y_n | X_{k+1}) \mathbb{P}(X_{k+1} | X_k) \quad (\text{as } Y_{k+1} \coprod Y_{k+2}, \dots, Y_n | X_{k+1}) \\
&= \sum_{X_{k+1}} \beta_{k+1}(X_{k+1}) \mathbb{P}(Y_{k+1} | X_{k+1}) \mathbb{P}(X_{k+1} | X_k)
\end{aligned} \tag{3}$$

If we write $\boldsymbol{\beta_k}$ as a vector $(\beta_k(0), \beta_k(1), \dots, \beta_k(n-1))^T$, we obtain that

$$\boldsymbol{\beta_k} = T^T(B_{:,Y_{k+1}} \odot \boldsymbol{\beta_{k+1}}),$$

where $\odot$ denotes the element-wise product. This is called the **backward** pass.
Finally, we can compute the vector $\boldsymbol{\gamma_k} := (\gamma_k(0), \gamma_k(1), \dots, \gamma_k(n-1))^T$ as follows:

$$\boldsymbol{\gamma_k} = normalize(\boldsymbol{\alpha_k} \odot \boldsymbol{\beta_k}),$$

where the normalization comes from the denominator in equation (1).

Thee three functions are implemented respectively in the `forward()`, `backward()` and `foward_backward()` functions.

6. Assume now that at each step the robot can be kidnapped with probability $\mathbf{k}$. If the robot is kidnapped its new position is independent from its previous position and is uniformly distributed. Repeat 4 and 5 for this new model with $\mathbf{k} = 0.1$. Can you reuse your code?

SOLUTION:
We just need to modify the transition probability as follows:

$$\mathbb{P}(X_{k+1} = j | X_k = i) = \begin{cases} (1-\mathbf{k})\epsilon + \dfrac{\mathbf{k}}{N} & \text{if } j = i \\ (1-\mathbf{k})(1-\epsilon) + \dfrac{\mathbf{k}}{N} & \text{if } j = i+1 \\ \dfrac{\mathbf{k}}{N} & \text{otherwise} \end{cases}.$$

Then the new transition matrix is:

$$T = (1-\mathbf{k})\epsilon I_{N\times N} + (1-\mathbf{k})(1-\epsilon)K_{N\times N} + \frac{\mathbf{k}}{n}L_{N\times N}.$$

The generation process is implemented in the `generate_kidnap()` function.