# Course 52935 Spring 2024
# Probabilistic Methods in Artificial Intelligence
# Project 2 - Inference

Prof. Gal Elidan
TA: Ela Fallik

Due: 27/06/2024

## Part II

**Programming Assignment - Inference in HMM**

We look at the same HMM problem from Project 1: Identifying active promoters. This time, we want to use the model in order to answer certain queries in an efficient way. Specifically, we'll focus on the marginal posterior query $p(X_t|o_{1:T})$. We'll use dynamic programming to calculate this posterior, and then compare the exact calculation to the estimation of two sampling procedures: Likelihood Weighting and Gibbs.

**Technical Remarks:**

In this project, you'll need to add methods to the HMM class you wrote in Project 1. The signatures of the methods you're required to implement appear in `HMM_prog2.py`. The questions code (that loads data, calls the different methods and creates the plots) is supplied in `prog2.py`. Additionally, some other functions are given in `utils.py`. We provide one HMM model (set of CPDs) for this project, `HMM1`. For simplicity, we'll look at short sequences $T = 10$.

### 1. Exact inference using dynamic-programming (DP)

1. You are supplied with code for the forward and backward algorithms. Use them to implement the `log_posterior_Xt` method in the HMM class.

2. Exact posteriors: We load the observations from `obs_data.csv`, calculate the marginal posteriors for the $N = 20$ observations in the dataset $p(X_t = 1|o_{1:T}[i])$, and plot the prior distributions $p(X_t = 1)$ versus the mean marginal posteriors $\mu_t = \frac{1}{N}\sum_{i=1}^{N} p(X_t = 1|o_{1:T}[i])$ for each $t$.

   What are the differences and why do they exist? What does it suggest about the distribution from which the observations were sampled?

3. We want to use the marginal posteriors $p(X_t|o_{1:T}[i])$ to assign each location $t$ its status and predict areas of active promoter:

$$\hat{X}_t[i] = \arg\max_x p(X_t = x|o_{1:T}[i])$$

   Implement a `naive_predict_by_posterior` method in the HMM class.

4. Prediction: We use this method to predict hidden sequences for the given observations. We compare them to the ground-truth sequences from `hidden_data.csv`. What is the accuracy of these predictions (use the same definition as Project 1)? What is the difference compared to the prediction we used in Project 1? Is the rule in Eq.4 a good prediction rule?

## 2. Sampling-based inference

Given the complexity of exact inference for most problems, we usually turn to approximate inference methods. We'll use this problem to test two sampling-based methods that estimate the marginal posterior $p(X_t|o_{1:T})$ - Gibbs sampling and Likelihood Weighting.

1. Gibbs sampling: In this algorithm, we'll start with a starting point $X_{1:T}$ sampled from the prior $p(X_{1:T})$, and sample $M$ sequences of hidden states sequentially:

$$X_t^{(m)} \sim p(X_t|X_{-t}^{(m-1)}, o_{1:T}[i]), \quad t = 1, \ldots, T$$

   We then use the samples to estimate the posterior $p(X_t = x|o_{1:T}[i])$ for each $t$.

   (a) For observations $o_{1:T}[i]$, at iteration $m$, we want to use the samples from the previous iteration $\{X_t^{(m-1)}\}_{t=1}^{T}$ to sample $X_t^{(m)}$ from the distribution $p(X_t|X_{-t}^{(m-1)}, o_{1:T}[i])$. How can we sample from this distribution using the CPDs of the network?

   (b) Given $M$ samples $\{X_t^{(m)}\}_{m=1}^{M}$ from the process described above, how can we calculate the posterior $p(X_t = x|o_{1:T}[i])$, while discarding the first $M_{\text{start}}$ samples to allow for a "burn-in" period?

   (c) Implement a method `gibbs_sampling_posterior` in the HMM class to calculate the posterior $p(X_t = x|o_{1:T}[i])$ for each $t$ with $M - M_{\text{start}}$ samples.

   (d) We run the Gibbs algorithm with $M \in [10, 50, 70, 100, 200, 300, 500]$ samples to calculate the marginal posteriors of each observation in the dataset. We run the algorithm 5 times for each value of $M$. We plot the posterior of $X_5$ for the first 10 observations $p(X_5|o_{1:T}[1:10])$ versus $M$ with confidence intervals ($\mu \pm \sigma$), and add to the plot the exact posterior calculated using DP. Does the Gibbs estimation converge? Does it converge to the correct posterior? Explain your results. What can you say about the convergence of this algorithm?

2. (BONUS) Likelihood Weighting (LW): In the second algorithm we consider, we'll use the Forward Sampling method from Project 1 to sample $M$ samples, while fixing the observations $o = o_{1:T}[i]$. We then calculate the likelihood weight of each sample, and use these weights to estimate the posterior $p(X_t = x|o_{1:T}[i])$ for each $t$.

   (a) For observations $o_{1:T}[i]$, we want to Forward sample $M$ samples from the network while keeping the observations fixed. Describe this process in the HMM network. Which variables do we need to sample and how?

   (b) For observations $o_{1:T}[i]$, at iteration $m$, we want to calculate the likelihood weights for the observed variables $E$, $w^{(m)} = \prod_{e \in E} p(e|Pa_e)$. How can we calculate this using the CPDs of the network?

   (c) Implement a method `likelihood_weighting_posterior` in the HMM class to calculate the posterior $p(X_t = x|o_{1:T}[i])$ for each $t$ with $M$ samples.

   (d) We run the LW algorithm with $M \in [10, 50, 70, 100, 200, 300, 500]$ samples to calculate the marginal posteriors of each observation in the dataset. We run the algorithm 5 times for each value of $M$. We plot the same plots as for Gibbs. What can you say about the convergence of this algorithm?

3. We use the estimated posteriors of each algorithm and the `naive_predict_by_posterior` method to assign each location its status (same as for the exact posterior). We plot the accuracy of the algorithms as a function of $M$, and plot on the same graph the accuracy of the exact posterior (as a constant function of $M$). What is the trend?