Lab 4
Markov Random Fields (MRFs)

Ex. 1 (1p) We consider a small Markov Random Field describing the distribution of 5 binary (with values -1 and 1) variables: A_1, A_2, A_3, A_4 and A_5. The relations of conditional independence are given by the following links: A_1 is connected with A_2 and A_3; A_2 is connected with A_4 and A_5, A_3 is connected with A_4; A_4 is connected with A_5.

The local functions corresponding to the cliques of the model are known:

$$\phi_{A_{i_1},A_{i_2},\ldots,A_{i_k}} = e^{i_1 A_{i_1} + i_2 A_{i_2} + \ldots + i_k A_{i_k}}.$$

a) (0.5p) Using pgmpy, visualize the graph of the Markov network; then determine the cliques of the model.
b) (0.5p) Determine the joint distribution of the five variables and infer the states of maximum probabilities (the best configuration) for them.

Ex. 2 (2p) A classic problem that can be solved using Markov Random Fields is **image denoising**, where the objective is to clean a noisy image by estimating the value of each pixel based on its neighbors.

1. *(Image Selection)* First, choose a very small image (around size 5×5) to use as the original image. The image can be created either deterministically or randomly.

2. *(Noise Introduction)* Randomly select a number of pixels (around 10%) to be modified, thereby creating a noisy version of the original image.

We consider a 2D grid where each node (pixel) is connected to its four direct neighbors (North, South, East, West). The total energy $E(x)$ of a grid can be described by two components:

- A component that penalizes discrepancies between observed and latent variables.
- A component that penalizes discrepancies between neighboring pixels, favoring a smoother image.

The energy function can be written as:

$$E(x) = \sum_i \lambda (x_i - y_i)^2 + \sum_{(i,j)} (x_i - x_j)^2$$

where:

- x_i is the clean pixel to be estimated;
- y_i is the observed (noisy) pixel;
- λ is a regularization factor that controls the influence of the noise, chosen by the user;
- (i, j) represents pairs of neighboring pixels.

a) (1p) Using pgmpy, define a Markov network that describes the context above.
b) (1p) Use the **Maximum A Posteriori (MAP)** method to estimate the original image starting from the noisy one. (you can use BeliefPropagation and map_query functions.)