Exercise 2

Deadline: 9.11.2017, 1:00 pm (no lecture or exercise group next week)
Regulations:

- You need $\geq 50\%$ of all points in the weekly exercises and successfully work on a ≥ 2 weeks fulltime project and submit a written report in order to pass this class.
- Hand in your solutions in groups of two or three people.
- Explain and show your answers in a single PDF file. Add references to your code files such that it is clear which file has to be run for each exercise and how the plots and results are generated. **Just handing in code or individual plots is not sufficient.**
- Send a single file (zip, rar, tar, ...) containing your solutions including all graphics, descriptions, source code and your write up to mlcv1718@gmail.com. **Do not send more than a single attachment.**
- The subject line of this email **must** start with **[EX02]** followed by the full names of all group members. **People not mentioned in the subject will not get credits.**
- Make sure your code works with the Anaconda python 3.6 distribution (`https://www.anaconda.com/download`).

Exercise 1. 5 P.

This exercise can be solved entirely without writing any code.

Let $E(x_0, x_1, x_2)$ be an energy-function of 3 binary variables $x_0 \in \{0, 1\}$, $x_1 \in \{0, 1\}$ and $x_2 \in \{0, 1\}$. $E(x_0, x_1, x_2)$ is a sum of factors where each factor depends only on a subset of variables:

$$E(x_0, x_1, x_2) = \phi_0(x_0) + \phi_1(x_1) + \phi_2(x_1) + \phi_p(x_0, x_1) + \phi_p(x_0, x_2) + \phi_p(x_1, x_2)$$

The unary factors are defined as:

$$\phi_0(x_0) = \begin{cases} 0.1 & \text{if } x_0 = 0 \\ 0.9 & \text{if } x_0 = 1 \end{cases} \qquad \phi_1(x_1) = \begin{cases} 0.8 & \text{if } x_1 = 0 \\ 0.1 & \text{if } x_1 = 1 \end{cases} \qquad \phi_2(x_2) = \begin{cases} 0.9 & \text{if } x_2 = 0 \\ 0.1 & \text{if } x_2 = 1 \end{cases}$$

And the pairwise factor is a Potts function:

$$\phi_p(x_i, x_j) = \begin{cases} 0 & \text{if } x_i = x_j \\ 1 & \text{if } x_i \neq x_j \end{cases}$$

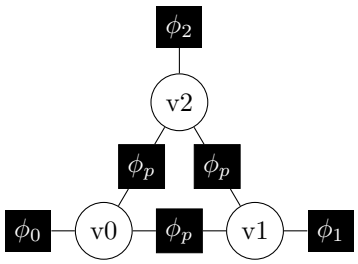

Figure 1: A graphical representation of $E(x_0, x_1, x_2)$ as factor graph. Each black box corresponds to a factor, each circle corresponds to a variable.

a) Evaluate $E(x_0, x_1, x_2)$ by hand for all possible configuration of x_0, x_1 and x_2

b) Which configuration of x_0, x_1 and x_2 minimizes $E(x_0, x_1, x_2)$? Comment on the result.

Exercise 2. 5 P.

This exercise can be solved entirely without writing any code.

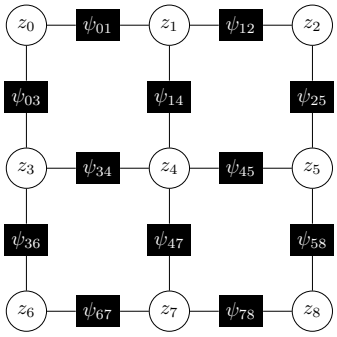

Figure 2: A graphical representation of a $N \times N$ graphical model with only binary terms as a factor graph. Each black box corresponds to a factor, each circle corresponds to a variable.

Consider a graphical model without unary terms (i.e. without a likelihood term) on a grid graph as in the picture with the size $N \times N$ where each variable z_i can take only two states: $z_i \in \{0, 1\}$. Assume

$$\psi_{ij}(z_i, z_j) = \begin{cases} \alpha & \text{if } z_i = z_j \\ \beta & \text{if } z_i \neq z_j \end{cases}$$

as the prior. The energy of this model is given as the following sum over all neighboring nodes:

$$E(Z) = \sum_{i \sim j} \psi_{ij}(z_i, z_j)$$

Show all global minima of E for

a) $\alpha = 0$ and $\beta = 1$ b) $\alpha = 1$ and $\beta = 0$

(Hints: You do not need to prove that you found all optima. Showing an example for a single $N \geq 3$ with instructions for how to construct the solutions for larger N is enough. There are two different configurations for (a) and (b) each.)

★ **Bonus Exercise 3.** (+2 P.)

Consider a model with binary variables and without unary terms again. This time the structure of the graphical model is arbitrary and not necessarily a grid. The regularizer is again given by:

$$\psi_{ij}(z_i, z_j) = \begin{cases} \alpha & \text{if } z_i = z_j \\ \beta & \text{if } z_i \neq z_j \end{cases}$$

a) Is it possible to find Z such that $E(Z) = 0$ for $\alpha = 0$ and $\beta = 1$ for all such models with binary variables regardless of the structure / topology of the graphical model? Describe how.

b) Is it possible to find Z such that $E(Z) = 0$ for $\alpha = 1$ and $\beta = 0$ for all such models with binary variables regardless of the structure / topology of the graphical model? If not, which properties must be fulfilled to be able to find Z such that $E(Z) = 0$.

Exercise 4. 5 P.

The goal of this exercise is to use Gibbs sampling to sample from two different priors:

a) $\psi_{ij}(z_i, z_j) = \begin{cases} 0 & \text{if } z_i = z_j \\ \alpha & \text{if } z_i \neq z_j \end{cases}$

b) $\psi_{ij}(z_i, z_j) = \alpha |z_i - z_j|$

Assume a $N \times M$ grid where each variable can take S different states. Start with a random image and use the algorithm discussed in the lecture. You can chose suitable values for N, M, $\alpha \geq 1$ and $S > 3$ yourself.

Hints:

- Use `numpy.random.choice` with the optional argument `p` to sample from a non-uniform distribution.
- If your sampler does not converge you can increase α. I used $\alpha = 2$, $S = 5$, $N = 40$ and $M = 60$

Run your sampler for both regularization terms and plot different samples. Comment on your results.

You can use the following template if you want to (also available on the website):

```
import numpy as np
import matplotlib.pyplot as plt

def prior_l0(n_states):
    prior = np.zeros((n_states, n_states))
    # TODO: fill matrix such that it represents the first prior
    return prior

def prior_l1(n_states):
    prior = np.zeros((n_states, n_states))
    # TODO: fill matrix such that it represents the second prior
    return prior

def calc_proba(lattice, x, y, prior):
    states = []
    # TODO: fill states with probabilities for all possible n_states
    return states

def gibbs_update(lattice, x, y, prior):
    # TODO: chose new state
    lattice[x,y] = new_state

def sweep_scanlines(lattice, prior):
    # TODO: itererate over lattice and update all nodes

n_iter = 100
n_states = 5
n_x = 40
n_y = 40

# TODO: setup lattice to random integers between 0 and n_states with shape (n_x, n_y)
```

```
# TODO: call function to setup prior to be a (n_states,n_states) array

for _ in range(n_iter):
    sweep_scanlines(lattice, prior)

f = plt.figure()
ax = f.add_subplot(1, 1, 1)
ax.imshow(lattice, cmap='gray', vmax=n_states)
f.savefig("sample.png")
```

★ **Bonus Exercise 5.** (+4 P.)

Use a graphical model with only pairwise interactions on a grid. Set the unaries to the class probabilities from the first exercise sheet and choose a suitable prior. Modify your Gibbs sampler to sample from the posterior distribution.

a) Comment on the prior you selected.

b) Plot different samples from the posterior.

c) Plot the mean taken over many samples. (Make sure the samples you use are not correlated.)

d) Plot the distribution of the different classes for each pixel (i.e. plot an image for each class s that shows how often each pixel was assigned to the class)