

# Programming Assignment 2

CMPT 727  
Spring 2020

In this assignment, you will implement Gibbs sampling for the image restoration problem defined in Problem 5 of homework 4.

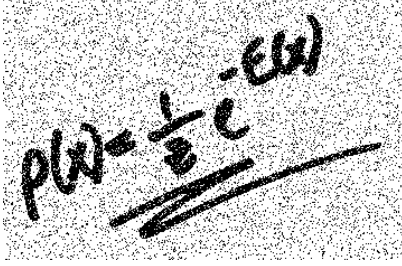


Figure 1: Noisy Image

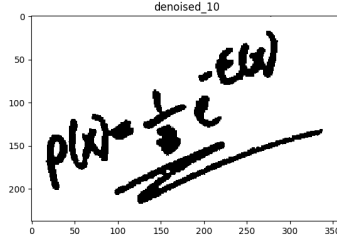
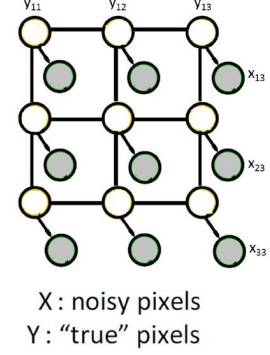


Figure 2: Denoised Image



Graphical model structure, for  $N = M = 3$ .

Figure 3: Graphical model

As a reminder, we use  $x = \{x_{ij}\}$  to denote the observed image, with  $x_{ij} \in \{-1, +1\}$  representing the pixel at row  $i$  and column  $j$ . Assume a black-and-white image, with  $-1$  corresponding to white and  $+1$  to black. The image has dimensions  $N \times M$ , so that  $1 \leq i \leq N$  and  $1 \leq j \leq M$ . Assume a set of (unobserved) variables  $\mathbf{y} = \{y_{ij}\}$  representing the true (unknown) image, with  $y_{ij} \in \{-1, +1\}$  indicating the value of  $x_{ij}$  before noise was added. Each (internal)  $y_{ij}$  is linked with four immediate neighbors,  $y_{i-1,j}$ ,  $y_{i+1,j}$ ,  $y_{i,j-1}$ , and  $y_{i,j+1}$  which together are denoted  $y_{Nbr(i,j)}$ . Pixels at the borders of the image (with  $i \in \{1, N\}$  or  $j \in \{1, M\}$ ) also have neighbors denoted  $y_{Nbr(i,j)}$ , but these sets are reduced in the obvious way. We denote  $E$  the corresponding set of edges.

We have derived the conditional probability of the  $y_{ij}$  is black given its Markov Blanket, where we use the logistic function  $\sigma(x) = \frac{e^x}{1+e^x}$ :

$$p(y_{ij} = 1 \mid y_{Nbr(i,j)}, x_{ij}) = \sigma(2\beta \left[ \sum_{y_{x,y} \in y_{Nbr(i,j)}} y_{x,y} \right] + 2\eta x_{ij})$$

For example:

$$\begin{aligned} p(y_{11} = 1 \mid y_{Nbr(1,1)}, x_{11}) &= p(y_{11} = 1 \mid y_{21}, y_{12}, x_{11}) \\ &= \sigma(2\beta[y_{21} + y_{12}] + 2\eta x_{11}) \end{aligned}$$

You will apply your implementation to two small, black-and-white images that have been made available with the problem set. These two noisy images—and the original, undistorted image from which they derive—are available both in PNG format and in a simple text format that lists each coordinate pair  $(i, j)$  and the corresponding value of  $x_{ij}$ . You may find it useful to convert between this text representation and a viewable image format.

We have provided some helper code for this assignment. You can download it [here](#).

## What to submit

Please submit the following two files to CourSYS:

- gibbs.py – Your completed implementation.
- report.pdf – A pdf file answering all the questions in this assignment.

## (4 points) Question 1

Outline a Gibbs sampling algorithm (in pseudocode) that iterates over the pixels in the image and samples each  $y_{ij}$  given its Markov blanket. Use the simple approach of sweeping across the image in row-major fashion on every iteration of the algorithm. Thus, an “iteration” will generate a complete new sample of  $y$ . Allow for a burn-in of  $B$  iterations, followed by draws of  $S$  samples. You may assume  $\eta$  and  $\beta$  are fixed constants.

## (10 points) Question 2

Implement your algorithm and apply it to the noise image A (a\_noise10.png.txt). Use values of  $\eta = 1$ ,  $\beta = 1$ ,  $B = 50$ , and  $S = 200$ . On each iteration of your algorithm, compute the energy  $E(\mathbf{y}, \mathbf{x})$  using the formula below for the current sample of  $y$  and output it to a log file, keeping track of which values correspond to the burn-in. Run your algorithm with three different initializations - one in which each  $y_{ij}$  is initialized to  $x_{ij}$ , one in which each  $y_{ij}$  is initialized to  $-x_{ij}$  and one in which the  $y_{ij}$  are set to 1 or +1 at random. Plot the energy of the model as a function of the iteration number for all three chains and visually inspect these traces for signs of convergence.

$$E(\mathbf{y}, \mathbf{x}) = -\eta \sum_{i=1}^N \sum_{j=1}^M y_{ij} x_{ij} - \beta \sum_{((i,j),(i',j')) \in E} y_{ij} y_{i'j'}$$

(Notice in particular that each pair of neighbors,  $y_{ij}$  and  $y_{i'j'}$ , factors into the formula only once, despite that each variable is a neighbor of the other. Failing to account for this will lead to double counting of  $\beta$  values.)

- Do all three seem to be converging to the same general region of the posterior, or are some obviously suboptimal?

- Does the burn-in seem to be adequate in length?
- Is there substantial fluctuation from iteration to iteration, indicating that the chain is mixing well, or does it become stuck at particular energies for several iterations at a time?

## (4 points) Question 3

Have your program output a restored image after completing its sampling iterations, by thresholding the estimated posterior probabilities for the  $y_{ij}$  variables at 0.5 — i.e., by estimating the “true” color of each pixel  $(i, j)$  as:

$$\hat{y}_{ij} = \begin{cases} +1, & \text{if } p(y_{ij} = 1|\mathbf{x}) > 0.5 \\ -1, & \text{otherwise} \end{cases}$$

To estimate the required posterior probabilities, store a running count  $c_{ij}$  of the number of (retained) samples for which each  $y_{ij} = 1$ , and then use the Monte Carlo estimate:

$$p(y_{ij} = 1|\mathbf{x}) \approx \frac{1}{S} \sum_t 1(y_{ij}^{(t)} = 1) = \frac{c_{ij}}{S}$$

where  $y_{ij}^{(t)}$  represents the  $t^{\text{th}}$  sample of  $y_{ij}$ . Restore both the noise images in this way, using the same values of  $\eta$ ,  $\beta$ ,  $B$ , and  $S$  as above and  $y_{ij}$  initialized to  $x_{ij}$ . Evaluate the quality of the restoration by computing the fraction of all pixels that differ between the restored images (a\_noise10.png.txt and b\_noise10.png.txt) and the original image.

- Prepare a figure for each the the two images, showing the original, the noisy version, and the restoration side by side.
- Report the restoration error for each image.

*If you have implemented your algorithm correctly, your restored images should be quite close to the original.*

## Question 4

- a) Give one short piece of feedback about the course so far. What have you found most interesting? Is there a topic that you had trouble understanding? Are there any changes that could improve the value of the course to you?
- b) How many hours did you spend on this assignment?

# Programming Assignment 2

---

CMPT 727  
Spring 2020

Please provide your answers in your report.

*Thanks to Stephano Ermon for giving us permission to adapt this assignment from Stanford CS 228.*