HOMEWORK 2: PROGRAMMING BELIEF PROPAGATION AND MCMC[1]

10-708 PROBABILISTIC GRAPHICAL MODELS (SPRING 2022)
https://andrejristeski.github.io/10708-22/

OUT: Feb 16th
DUE: Feb 28th at 11:59 PM
TAs: Kai-Ling Lo, Aaron Rumack

START HERE: Instructions

- **Collaboration policy:** The purpose of student collaboration is to facilitate learning, not to circumvent it. Studying the material in groups is strongly encouraged. It is also allowed to seek help from other students in understanding the material needed to solve a particular homework problem, provided no written notes (including code) are shared, or are taken at that time, and provided learning is facilitated, not circumvented. The actual solution must be done by each student alone. The presence or absence of any form of help or collaboration, whether given or received, must be explicitly stated and disclosed in full by all involved. See the Academic Integrity Section on the course site for more information: https://andrejristeski.github.io/10708-22/#:~:text=Academic%20Integrity%20Policies
- **Late Submission Policy:** See the late submission policy here: https://andrejristeski.github.io/10708-22/#:~:text=Grace%20Day/Late%20Homework%20Policy
- **Submitting your work to Gradescope:** We use Gradescope (https://www.gradescope.com/courses/349316/assignments) to collect PDF submissions of open-ended questions on the homework (e.g. mathematical derivations, plots, short answers). The course staff will manually grade your submission, and you'll receive personalized feedback explaining your final marks. The homework template must be used and can be completed in Latex or by hand. Handwritten submissions must be legible otherwise we will not be able to give credit to your solutions. No changes should be made to the template, boxes and choices **MUST** remain the same size and in the same locations between the template and your completed submission, the document has 13 pages so your submission must contain no more and no less than 13 pages. Submit your PDF solution to Homework 2: Programming
- **Programming Code:** You will also submit your code for the programming questions on the homework to Gradescope, specifically the 'Homework 2: Code' submission slot. All code written must be submitted in order for you to get any credit for the written components of the programming section.
- For **multiple choice** or **select all that apply questions**, shade in the box or circle in the template document corresponding to the correct answer(s) for each of the questions. For LaTeX users, replace `\choice` with `\CorrectChoice` to obtain a shaded box/circle, and don't change anything else.

[1] Compiled on Wednesday 16th February, 2022 at 15:28

B Programming [30 pts]

Your goal in this assignment is to implement several Markov Chains for sampling for an Ising model. The graph for the Ising model will be a 2-D grid, where each node is connected only to the nodes directly above, below, left, and right. Your solution will be implemented in a Jupyter notebook, in which we have provided a template for the Ising model, and some visualization tools for you to use (you're welcome to modify them based on your own implementation).

The general Ising model was defined in Homework 1. In the specific case of a 2-D grid, we will modify the notation as follows. Consider the random variables $\mathbf{X} = \{\mathbf{X}_{ij} \in \{-1, 1\} : i \in [n], j \in [n]\}$ following the distribution of an Ising model G with parameters $\mathbf{J}$. Precisely, the joint distribution of the random variables is expressed as:

$$p(\mathbf{X} = \mathbf{x}) = \frac{1}{Z_G} \exp\Big(\sum_{(i,j)\in[n]\times[n]} \mathbf{J}_{ij}\mathbf{x}_{ij} + \sum_{(k,l)\neq(i,j)\in[n]\times[n]} \mathbf{J}_{ij,kl}\mathbf{x}_{ij}\mathbf{x}_{kl}\Big), \tag{B.1}$$

in which Z_G is the partition function. For brevity, we will denote the set of parameters $\{\mathbf{J}_{ij}\}$ as $\mathbf{J}_s$, and the set $\{\mathbf{J}_{ij,kl}\}$ as $\mathbf{J}_{st}$. Moreover, in the questions below, we will always set all the values in $\mathbf{J}_{st}$ or $\mathbf{J}_s$ to the same number — so e.g., if we say, set $\mathbf{J}_{st} = 0.1$, what we mean is set all $\mathbf{J}_{ij,kl}$ to 0.1.

B.1 Sampling algorithms

In this section, you will implement Gibbs sampling, as well as Gibbs sampling with tempering.

(a) (2 points) This requires calculating conditional probabilities without accessing the partition function. Write the conditional probability:

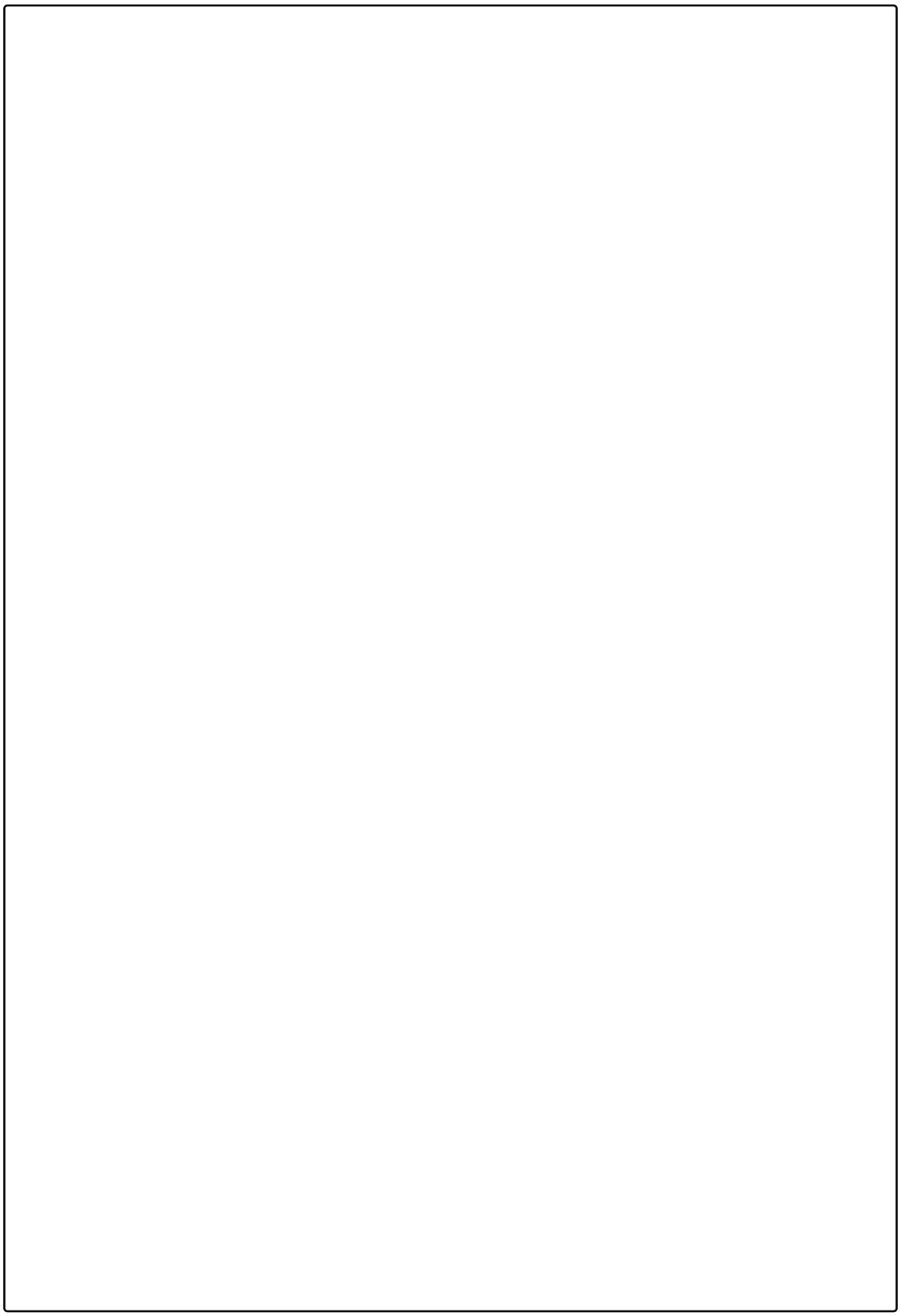
$$p(\mathbf{X}_{ij} = \mathbf{x}_{ij} | \{\mathbf{x}_{kl} : (k, l) \neq (i, j)\}),$$

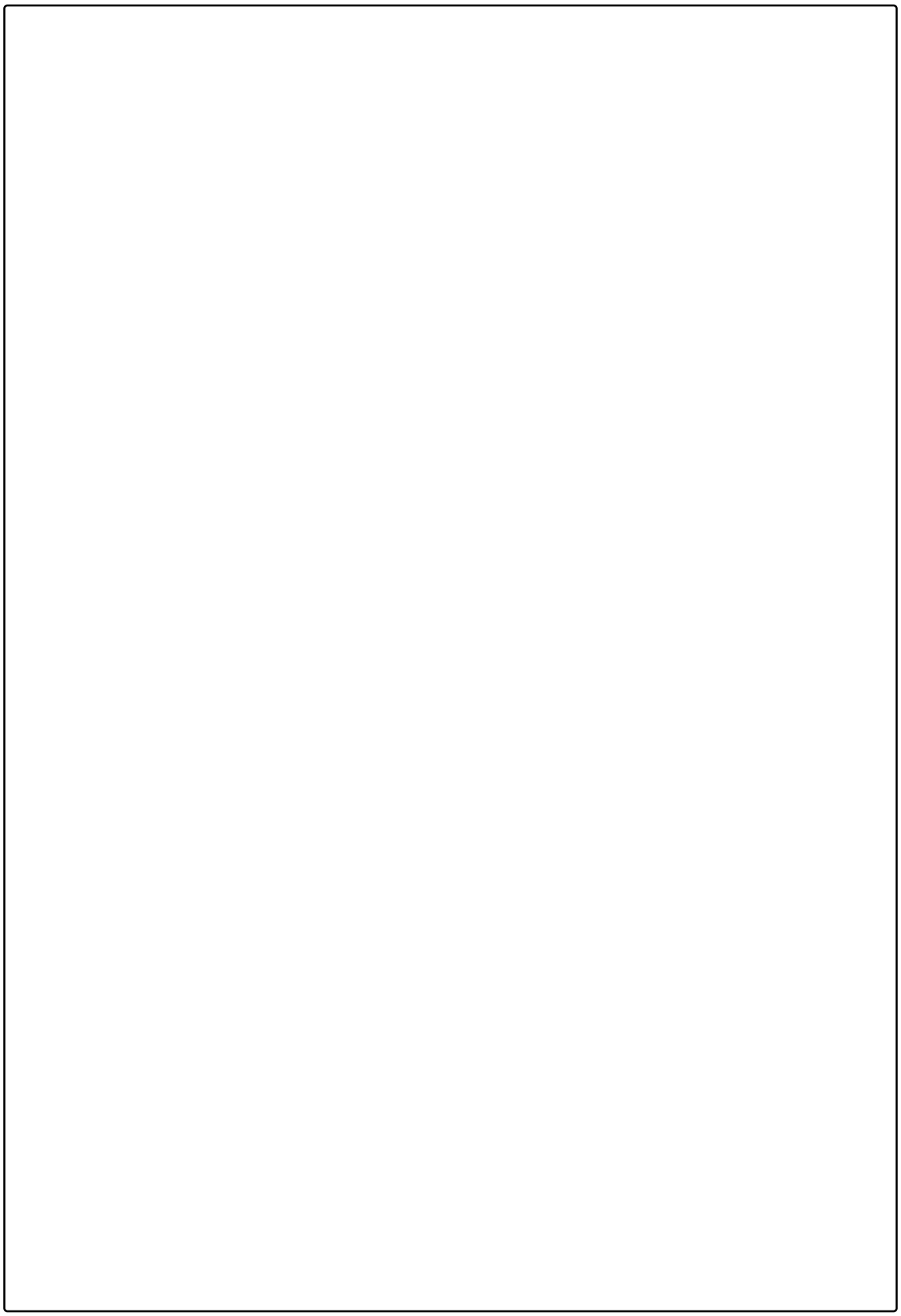
using at most 4 values of (k, l).

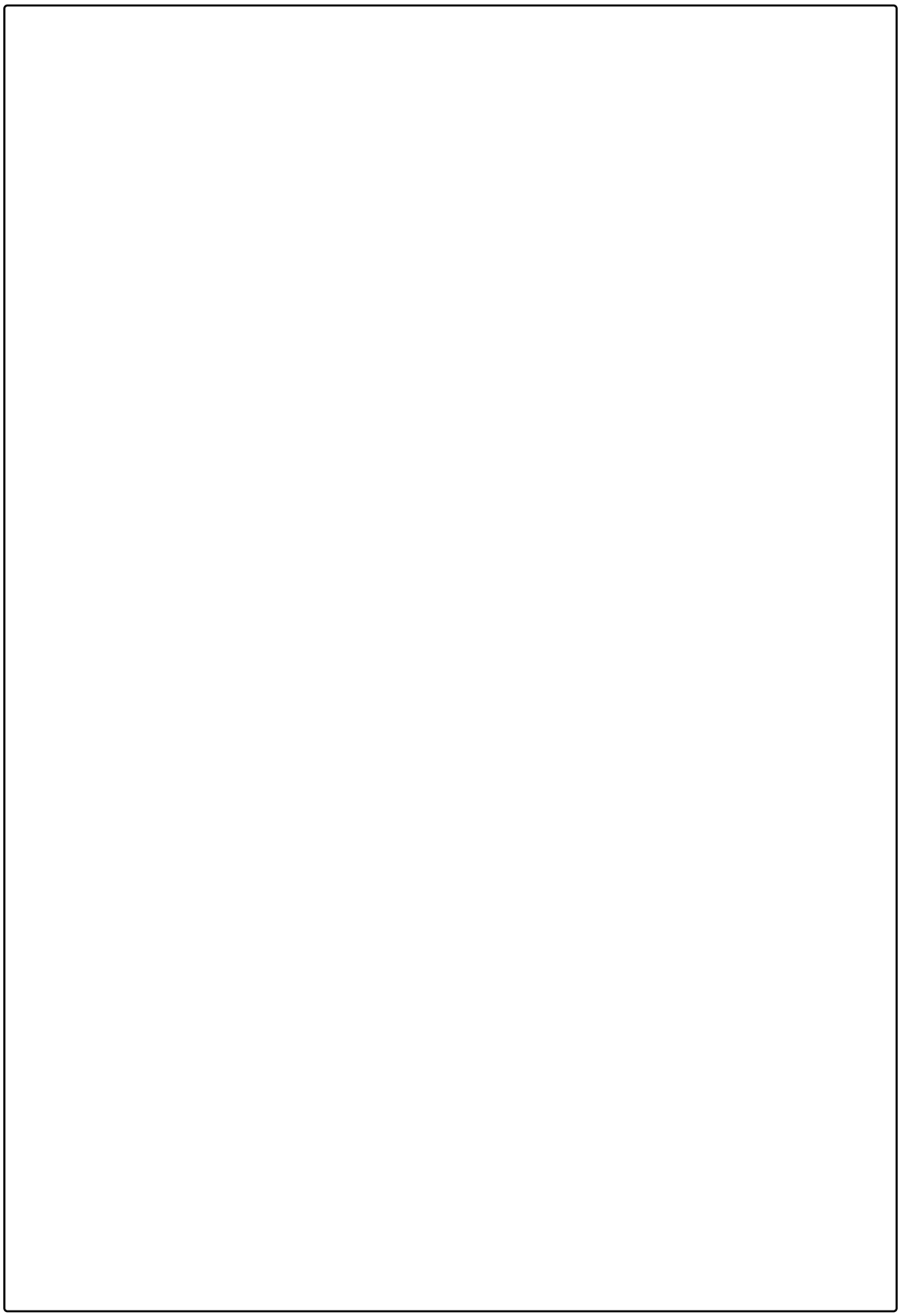
(b) (10 points) Implement Gibbs sampling in the provided Jupyter notebook. To be specific, you can follow the sketch of the algorithm below:

procedure GIBBSSAMPLING($p(\mathbf{X}), \mathbf{x}, T$)
    $\hat{\mathbf{x}} \leftarrow []$
    for $t \in (1 \dots T)$ **do**
        for $(i, j) \in$ shuffle($\{(1, 1), \dots, (N, N)\}$) **do**
            $\mathbf{x}_{ij} \sim p(\mathbf{X}_{ij} = \mathbf{x}_{ij} \mid \{\mathbf{x}_{kl} : (k, l) \neq (i, j)\}$
        $\hat{\mathbf{x}} \leftarrow \hat{x} + [\mathbf{x}]$ ▷ Append new sample to the list
    Return $\hat{\mathbf{x}}$

Run the sampler on a 100×100 grid for 400 iterations with same $\mathbf{J}_s = 0$ and different $\mathbf{J}_{st} = 0.25, 0.5, 1, 10$. For each experiment, plot the initial state and the state snapshot at every 100 iterations. There should be in total 20 plots.







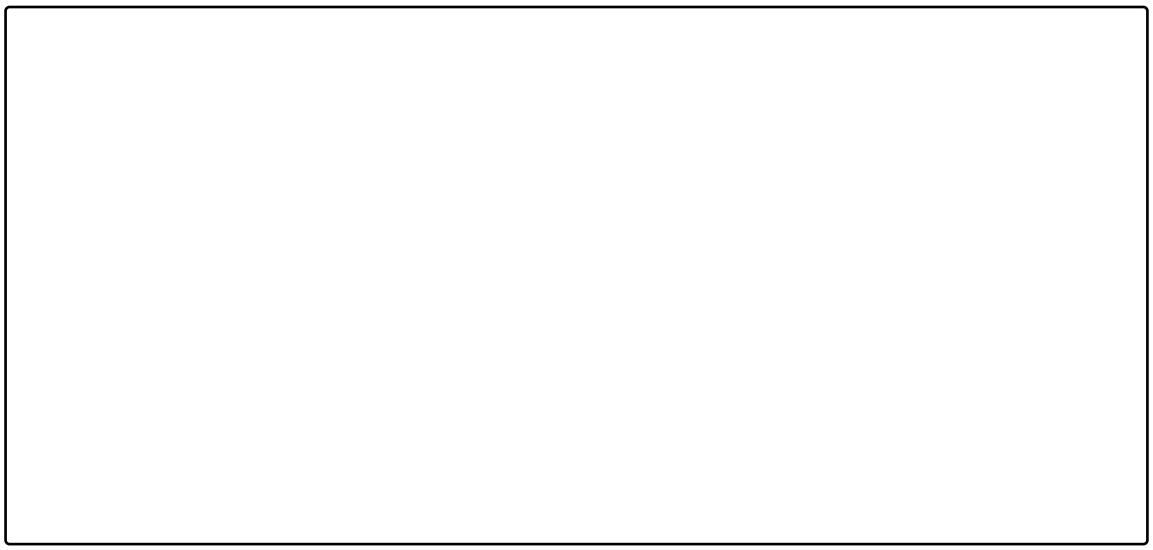
(c) (8 points) If instead of sampling, we are interested in calculating the partition function, we can do so through annealed importance sampling (AIS). To achieve this, consider p_0 as some distribution easy to sampled from, and p_1 as the target distribution that we would like to estimate the partition function for. With a sequence of $0 = \beta_0, \beta_1, \ldots, \beta_K = 1$, we can estimate (the log of) the partition function of p_1 as below:
(Note: $\hat{p}$ stands for the un-normalized potential function of p, and Z_0 is the partition function of p_0)

procedure AIS($p_0(\mathbf{X}), p_1(\mathbf{X}), \beta, T$)
  $\hat{Z} \leftarrow []$
  for $i \in (1 \ldots M)$ **do**
    Sample $\mathbf{x}_1$ from p_0
    for $j \in (1, \ldots, K-1)$ **do**
      Sample $\mathbf{x}_{j+1}$ given $\mathbf{x}_j$ using Gibbs sampling from model $p_{\beta_j} \propto p_0^{1-\beta_j} p_1^{\beta_j}$
      (i.e. $p_{\beta_j}(\mathbf{x}) \propto p_0(\mathbf{x})^{1-\beta_j} p_1(\mathbf{x})^{\beta_j}$)
    Calculate $\hat{z} = \log\left(\frac{\hat{p_1}(\mathbf{x}_1)}{\hat{p_0}(\mathbf{x}_1)} \frac{\hat{p_2}(\mathbf{x}_2)}{\hat{p_1}(\mathbf{x}_2)} \cdots \frac{\hat{p_K}(\mathbf{x}_K)}{\hat{p_{K-1}}(\mathbf{x}_K)}\right) + \log(Z_0)$
    $\hat{Z} \leftarrow \hat{Z} + [\hat{z}]$ ▷ Append new sample to the list
  Return $\frac{1}{M} \sum \hat{Z}[i]$

Consider a 10×10 grid, set

- p_0 as a uniform distribution over all possible configurations (i.e. a model with $\mathbf{J}_s = \mathbf{J}_{st} = 0$),
- p_1 as the Ising model with $\mathbf{J}_s = 0, \mathbf{J}_{st} = 1$ and
- $\beta_0, \beta_1, \ldots, \beta_K = 0, 0.001, \ldots 1 (K = 1001)$,

Run the experiments for $M = 1000$ rounds. In addition, conduct importance sampling (IS) to estimate the partition function with p_0 as the proposal distribution. You should sample 500000 samples from p_0 for the estimation. Report your estimation and the variance of your samples from both AIS and IS, also compare the difference between them.



(d) (10 points) Implement Gibbs sampling with tempering in the provided Jupyter notebook. For tempering you should modify your algorithm to follow the steps below for each iteration (not for each random variable):
Note: again, $\hat{p}$ stands for the un-normalized potential function, and you should apply the AIS to estimate the partition function $\hat{Z}_{T[i]}$ for each temperature.

- The state space is $\mathbb{R}^{n\times n} \times [L]$, where $[L]$ is the number of temperatures. Denote the temperatures as an array $\mathbf{T}$.
- Let $M_{\mathbf{T}[i]}$ be the Markov chain (i.e. the Gibbs chain) corresponding to temperature $\mathbf{T}[i]$ and let the current state be $(\mathbf{x}, \mathbf{T}[i])$.
- With probability 0.5, evolve $\mathbf{x}$ according to $M_{\mathbf{T}[i]}$ to new point $\mathbf{x}'$. Set next point to $(\mathbf{x}', \mathbf{T}[i])$.
- With probability 0.5, pick a neighbor of the current temperature $\mathbf{T}[i]$: i.e. $\mathbf{T}[i-1]$ or $\mathbf{T}[i+1]$, with probability 0.5 for each. If the chosen neighbor is $T[j]$, set the next point to $(\mathbf{x}, \mathbf{T}[j])$ with probability $\min\left(\frac{\hat{p}_{\mathbf{T}[j](\mathbf{x})/\hat{Z}_{T[j]}}}{\hat{p}_{\mathbf{T}[i]}(\mathbf{x})/\hat{Z}_{T[i]}}, 1\right)$.

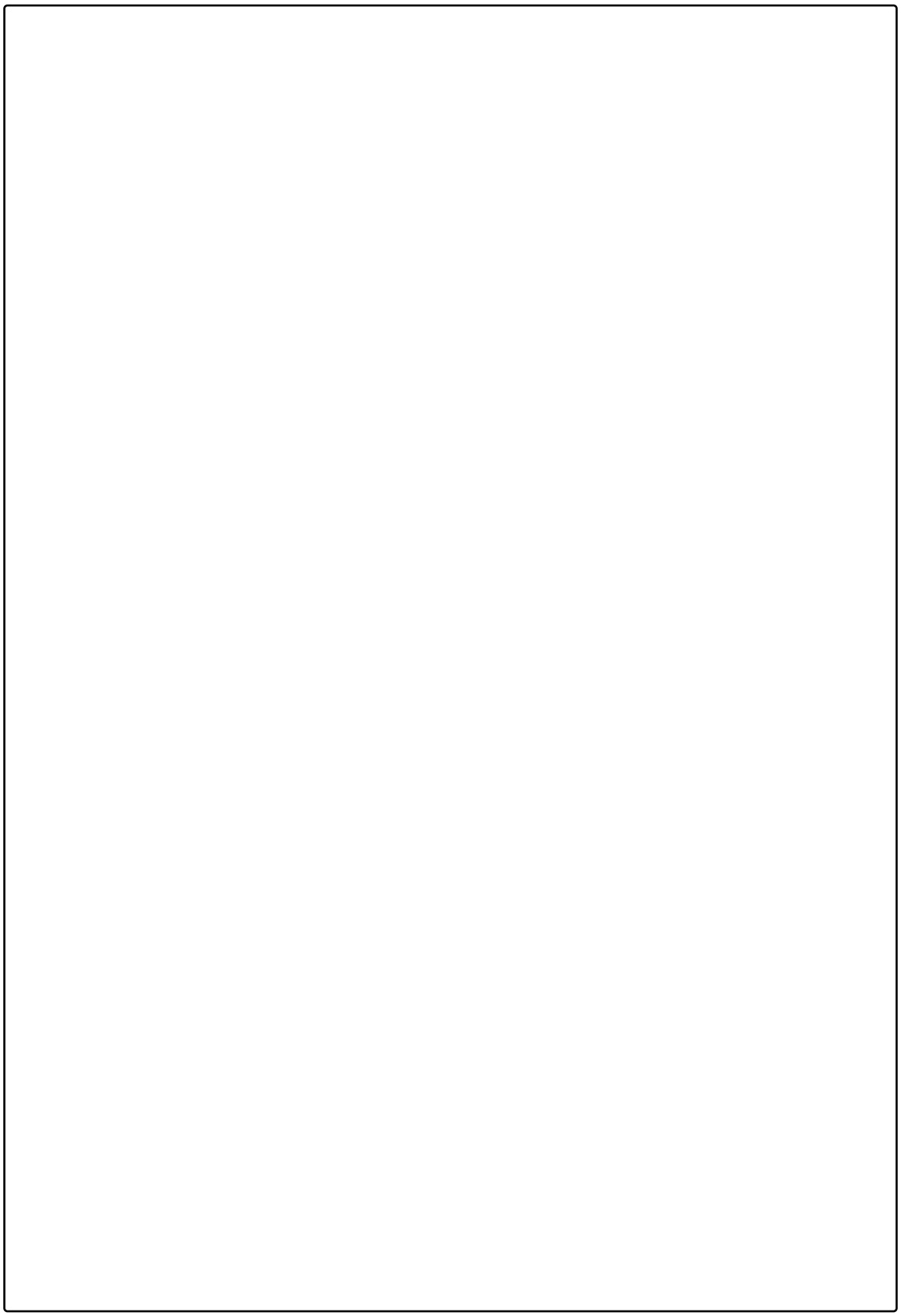
Consider a 5×5 grid, set

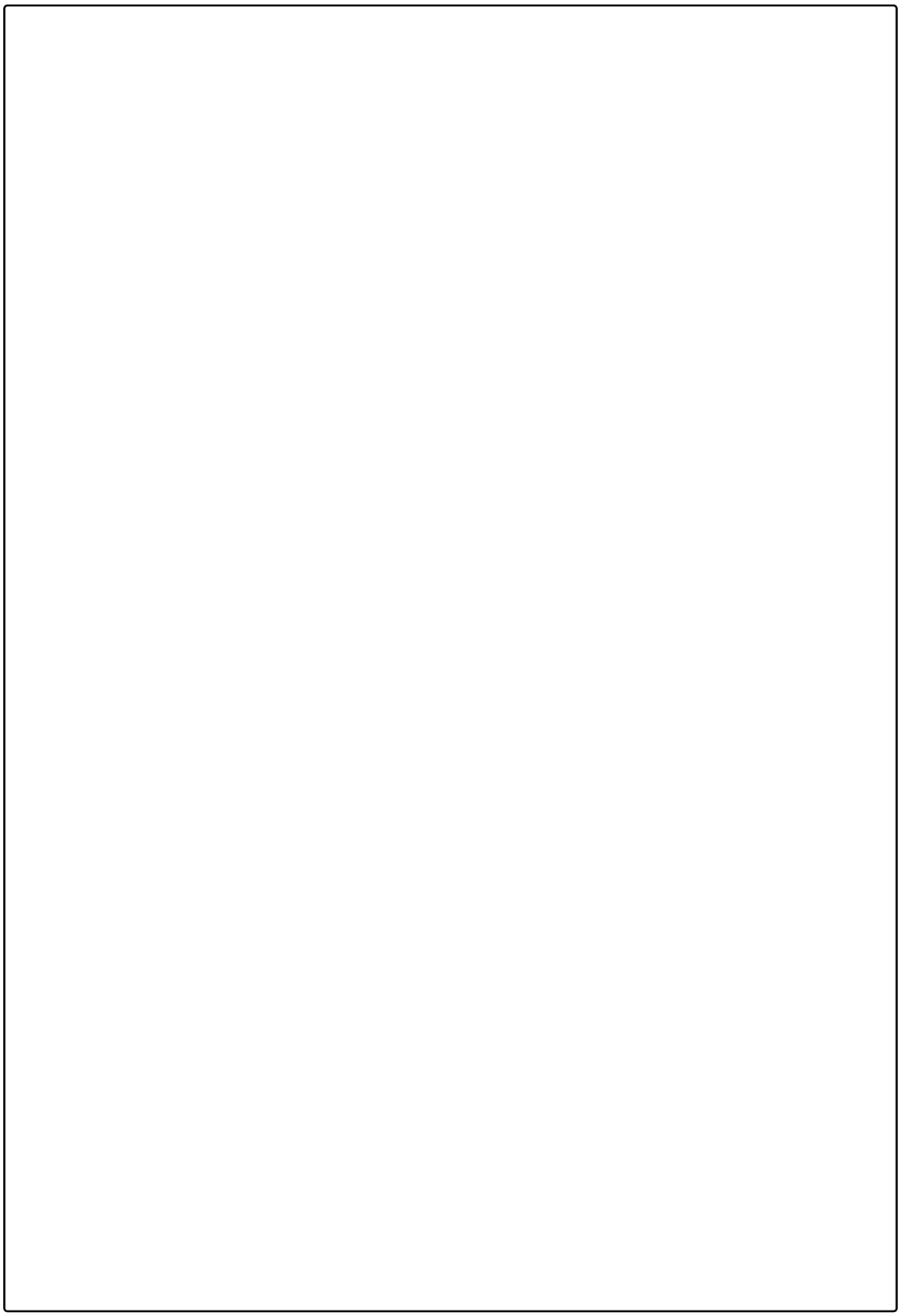
- $\mathbf{J}_s = 0$, $\mathbf{J}_{st} = 1.0, 1.2, 1.5, 2.0$, and
- $\mathbf{T} =$ `np.linspace(0.5, 2.0, 31)`, and
- for each AIS estimation, $K = M = 50$.

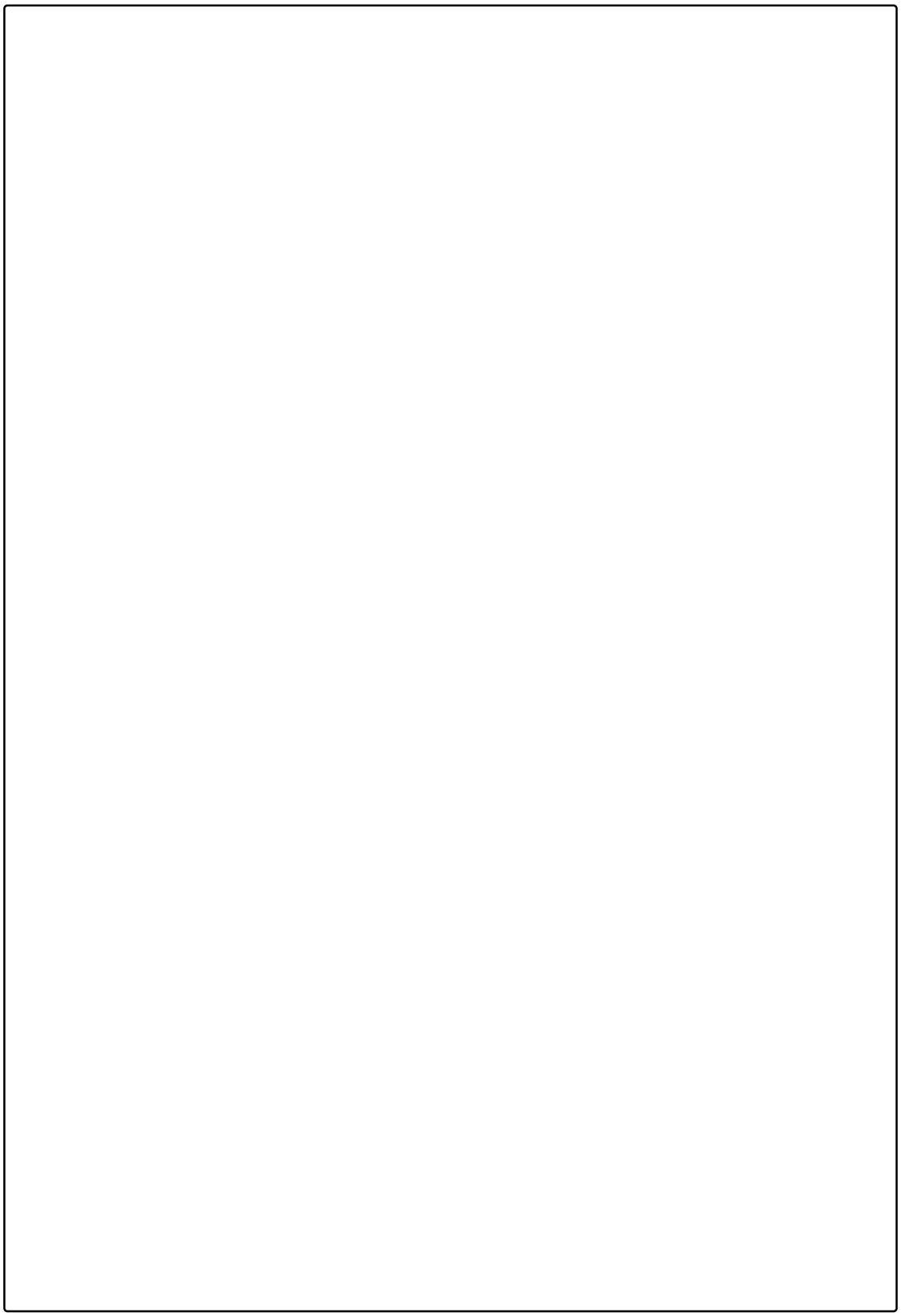
Run both vanilla Gibbs sampling and tempering sampling on the model for 100000 iterations. Provide the plots of

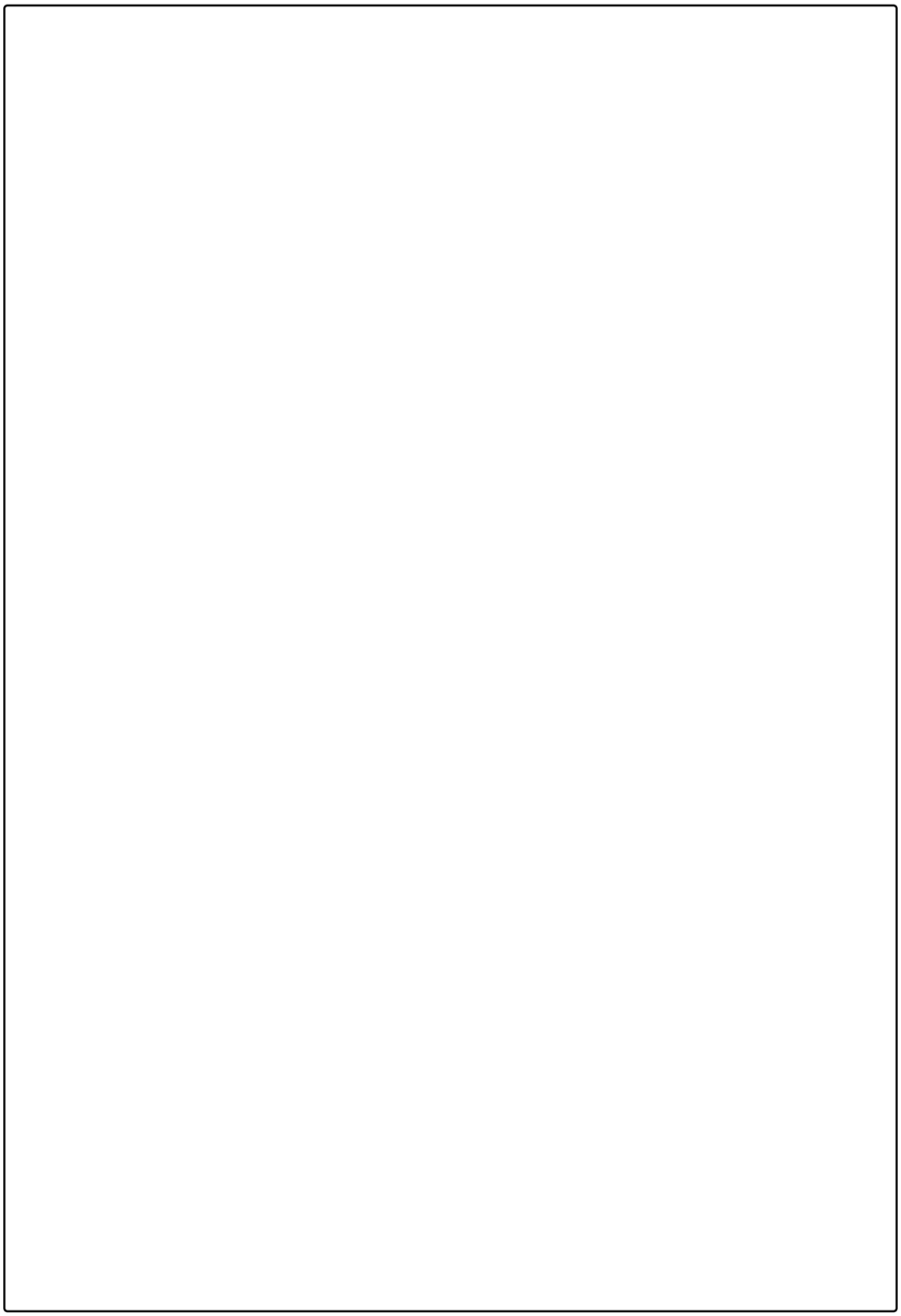
- the temperature of each iteration (for Gibbs sampling you can simply plot a horizontal line), and
- the summation of the variable assignments (e.g. $+25$ for all $+1$, and -25 for all -1)

versus the number of iterations. In addition, after running the tempering sampling, collect the samples under temperature $T = 1$, and plot the samples. There should be $4 \times (4 + 1) = 20$ plots in total.









B.2 Code Submission [0 pts]

You must submit all of your code to the "Homework 2 Code" slot on Gradescope. If you have multiple `.py` or `.ipynb` files, you must upload all of them. Your code will **not** be autograded on Gradescope. Instead, we will grade the programming part first by examining the written deliverable; when your result is far from what we expect, we'll look into your code to decide the partial credit. As such, please carefully identify major sections of the code via comments.