Simulation and Recovery of Ising Sparse Graphical Model

Yilun Cai, Ben Bentley, Bin Gao

November 2025

1 Introduction

Graphical models are a family of multivariate distributions which are Markov with respect to a particular undirected graph. Each node in the graph $i \in V$ is associated with a random variable. The set of edges $E \subset \binom{V}{2}$ encodes the conditional dependency relationships: a variable conditioned on its neighbours is independent of the remaining variables[2].In this project, we focus on the Ising model, a classical discrete Markov random field in which each variable takes values in $\{+1, -1\}$ and interactions occur only between pairs of variables.

1.1 The Ising Model

Let $y = (y_1, \ldots, y_p) \in \{\pm 1\}^p$ denote a collection of p binary variables. The Ising model is defined through the joint distribution

$$P(y) \;=\; \frac{1}{Z} \exp\left(\sum_{\{s,t\} \in E} u_{s,t}\, y_s y_t \right),$$

where, $u \in \mathbb{R}^{p \times p}$ is a symmetric matrix and encodes the graph structure (we set the diagonal elements of u to zero). In particular, $u_{s,t}$ is non-zero if variables s and t are connected via an edge; the sign and magnitude of u_{st} determine whether the variables tend to align or oppose each other. The partition function

$$Z(u) = \sum_{y \in \{\pm 1\}^p} \exp\left(\sum_{s<t} u_{st} y_s y_t \right)$$

serves as a normalizing constant.

Although the Ising model originates from statistical physics, it has become a widely used framework in statistics, biology, and network science for modeling pairwise interactions. Sparse structures—where each node interacts with only a small subset of others—are especially relevant for interpretability and computational tractability.

1.2 Sparse Graphical Structures and Motivation

Our project focuses on *sparse* Ising models, where most entries of u are zero. Sparse interaction networks arise naturally in many domains. For example, in a financial system, it is reasonable to assume that each asset interacts with only a small neighborhood of peers rather than the entire market.

In particular, we are interested in sparse graph structures, that is, each node is connected to few other nodes in comparison to the total number of nodes. This is more computationally difficult in comparison to dense graph structures to model as we will demonstrate, but has practical uses. For example, if we were attempting to model the effect of each agent in a financial market adjusting their strategy based on the behaviours of other agents, we might assume said agent can only observe a relatively small group of their peers instead of any sizable portion of the entire collection of agents. In this case it is unreasonable to construct the model with a dense graph structure.

Recovering a sparse interaction matrix is challenging: the exact likelihood depends on the partition function $Z(u)$, which requires summing over 2^p configurations and quickly becomes intractable for even moderate values of p. This motivates approximate inference techniques such as pseudo-likelihood, as well as sparsity-inducing priors (e.g., Laplace) in Bayesian inference.

1.3 Objective and Scope

The objective of this project is not to construct a realistic financial model, but rather to use a synthetic Ising model as a controlled environment for evaluating sampling and inference algorithms. Specifically, we aim to:

- Construct a known ground-truth sparse interaction matrix u^*.
- Generate samples from the Ising model using Gibbs sampling.
- Recover the interaction matrix using two inference methods:
    1. Exact Bayesian inference via Metropolis–Hastings sampling on the parameter space (tractable only for small p).
    2. Pseudo-likelihood MAP estimation, which scales to larger networks and avoids computing the partition function.
- Compare the estimated matrices to u^* to assess:
    - estimation accuracy,
    - convergence and mixing of sampling algorithms,
    - computational feasibility.

Using synthetic data is essential: since u^* is known, we can quantitatively evaluate estimation error, which is a central requirement of scientific computing studies. Future extensions will examine how factors such as sample size and temperature influence recovery performance.

2 Numerical Methods

This section describes the numerical techniques used in our project. Our computational objectives are:

- Generate samples from a known Ising model using Gibbs sampling.
- Recover the interaction matrix u^* using two inference methods:
  1. exact Bayesian inference via Metropolis–Hastings sampling on the parameter space (feasible only for small p),
  2. pseudo-likelihood MAP estimation, which scales to larger networks.
- Compare the recovered matrices with the ground-truth parameters.

We begin with notation and then present the sampling and inference procedures used in our experiments.

2.1 Notation

We consider $y = (y_1, \ldots, y_p) \in \{\pm 1\}^p$ and a symmetric interaction matrix $u \in \mathbb{R}^{p \times p}$ with zero diagonal. The Ising model has probability

$$P(y) = \frac{1}{Z(u)} \exp\left(\sum_{1 \le s < t \le p} u_{st} y_s y_t \right),$$

with partition function

$$Z(u) = \sum_{y \in \{\pm 1\}^p} \exp\left(\sum_{s<t} u_{st} y_s y_t \right).$$

Let $y_{V \setminus r}$ denote all coordinates except y_r. Given N samples $\{y^{(i)}\}_{i=1}^N$, our goal is to recover u^*.

2.2 Gibbs Sampling for Data Generation

Gibbs sampling is used to generate synthetic data from the known model specified by u^*. The sampler updates one coordinate at a time according to the conditional distribution.

Theorem 1. *The conditional distribution of a variable y_r of node r given the rest $y_{V/r}$ is given by*

$$P(y_r \mid y_{V/r}) \;=\; \frac{\exp\Big(2y_r \sum_{t \in V/r} u_{r,t} y_t\Big)}{\exp\Big(2y_r \sum_{t \in V/r} u_{r,t} y_t\Big) + 1}.$$

The full derivation of this conditional probability is provided in Appendix A.

The Theorem 1 indicates that we can use Gibbs sampling to generate samples, as at each sampling step for a given node r we only need to compute a Sigmoid using its neighbouring nodes. With the conditional distribution, Gibbs sampling proceeds as:

Algorithm 1 Gibbs Sampling

Initialize $y^{(0)} \sim \pi_0$.
for $n = 1$ to N **do**
    for $r = 1$ to p **do**
        Sample $y_r^{(n)} \sim P(y_r \mid y_{V \setminus r}, u^*)$.
    end for
end for

We validate this sampler in a $p = 5$ example, where the empirical correlation matrix agrees with the sparsity structure of u^*.

2.3 Random Walk Metropolis-Hastings

To recover the interaction matrix u from data, we first consider full Bayesian inference. Let $u_{\text{vec}} \in \mathbb{R}^{p(p-1)/2}$ denote the vector of free parameters (upper-triangular entries of u), and impose a sparsity-inducing Laplace prior

$$u_j \sim \text{Laplace}(0, \lambda).$$

The posterior distribution is

$$P(u \mid \{y^{(i)}\}) \propto \left[\prod_{i=1}^{N} \frac{\exp\left(\sum_{s<t} u_{st} y_s^{(i)} y_t^{(i)}\right)}{Z(u)}\right] \prod_j \frac{\lambda}{2} e^{-\lambda |u_j|}.$$

Intractability of Exact Bayesian Inference

Evaluating the Metropolis–Hastings acceptance probability for parameter updates requires the likelihood ratio $Z(u)/Z(u^*)$, where

$$Z(u) = \sum_{y \in \{\pm 1\}^p} \exp\Big(\sum_{s<t} u_{st} y_s y_t\Big).$$

Since computing $Z(u)$ involves summing over 2^p configurations, each MH update becomes exponentially expensive in p. Thus, exact Bayesian inference is feasible only for small models (e.g. $p \leq 5$). The complete derivation of the MH acceptance ratio and the intractability result is provided in Appendix B.

Random Walk Metropolis–Hastings Algorithm

We implement MH sampling for the posterior $P(u \mid \{y^{(i)}\})$ in the low-dimensional case $p = 5$, where $Z(u)$ can be computed exactly.

Algorithm 2 Random Walk Metropolis–Hastings

Let $P(y)$ denote the target distribution from which we wish to generate approximate samples. Choose an initial state $y^{(0)}$ (e.g. by sampling from some easy distribution π_0 or fixing it arbitrarily).

for $n = 1$ to N **do**

Propose a new state v^* by making a random-walk step:

$$v^* = y^{(n-1)} + \varepsilon, \qquad \varepsilon \sim q(\cdot),$$

where $q(\varepsilon)$ is symmetric around 0 (e.g. $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$).

Compute the acceptance probability

$$\alpha\big(y^{(n-1)}, v^*\big) = \min\left\{1,\ \frac{P(v^*)}{P(y^{(n-1)})}\right\},$$

which simplifies due to symmetry:

$$q(v^* \mid y^{(n-1)}) = q(y^{(n-1)} \mid v^*).$$

Draw $u \sim \text{Uniform}(0, 1)$.

if $u \leq \alpha\big(y^{(n-1)}, v^*\big)$ **then**

Accept the proposal: $y^{(n)} = v^*$.

else

Reject the proposal: $y^{(n)} = y^{(n-1)}$.

end if

end for

For $p = 5$, the sampler mixes well and posterior means closely match the true parameters u^*. For larger dimensions, however, the exponential cost of computing $Z(u)$ motivates the use of the pseudo-likelihood MAP estimator introduced in Section 2.4.

2.4 Pseudo Log-Likelihood MAP Estimation

We will now address the intractability problem: Random Walk Metropolis-Hastings becomes infeasible as the dimension p grows, since evaluating the partition function $Z(u)$ requires summing over 2^p states. To scale parameter estimation to larger networks, we adopt the *pseudo-likelihood* approximation introduced by Besag [1].

Theorem 2. *Suppose that the likelihood $P(y_1, y_2, \ldots, y_p \mid u)$ is well approximated by*

$$P(y_1, y_2, \ldots, y_p \mid u) \;\approx\; \prod_{r=1}^{p} P(y_r \mid y_{V/r}, u),$$

where the conditional probabilities are given explicitly by Theorem 1.

The MAP estimator of u is given by

$$\arg \min_{\substack{u \\ u=u^\top \\ \mathrm{diag}(u)=0}} \left\{ \sum_{i=1}^{N} \sum_{r=1}^{p} -\log\Big(P\big(y_r^{(i)} \mid y_{V/r}^{(i)}, u\big)\Big) \;+\; \frac{1}{\lambda} \sum_{i<j} |u_{ij}| \right\}.$$

A full derivation of this objective is provided in Appendix C.

The objective consists of (i) a smooth and convex pseudo-likelihood term, and (ii) a non-smooth ℓ_1 penalty. This suggests the use of *Proximal Gradient Descent (PGD)*, where each iteration alternates between a gradient step on the smooth term and a soft-thresholding update on the ℓ_1 term.

Proximal Gradient Descent for Pseudo-likelihood MAP

Algorithm 3 Proximal Gradient Descent for Pseudo-likelihood MAP

1: Initialize $u^{(0)}$ (symmetric with zero diagonal).

2: **return** $u^{(K)}$.

PGD and its implementations forthcoming in the final project report.

3 Package Structure

Our project is organized as a modular Python package located in the `src/` directory. The code structure separates the model definition, sampling procedures, inference algorithms, utility functions, and experiment scripts to ensure clarity, maintainability, and ease of testing. The main components are:

- `ising_model.py`

  Defines the Ising model, including the energy function, probability mass function, and exact partition function for small p. This module provides core computations that are shared across sampling and inference methods.

- `sampling.py`

  Implements Gibbs sampling for generating data from the known ground-truth interaction matrix u^*, as well as (optionally) Metropolis–Hastings sampling on the state space. The Gibbs sampler uses the conditional probability derived in Appendix A.

- `inference.py`

  Contains two inference methods for recovering the interaction matrix u: (i) the Random Walk Metropolis–Hastings algorithm for exact Bayesian inference (used only for small p), and (ii) the pseudo-likelihood MAP estimator, implemented using proximal gradient descent. This module encapsulates all optimization and posterior sampling logic.

- `utils.py`

  Provides auxiliary utilities, including matrix vectorization and unvectorization, soft-thresholding operators, correlation computations, plotting helpers, and I/O functions.

- `experiments.py`

  Contains reproducible scripts used to run the experiments in Section 5. These scripts call the sampling and inference routines from the modules above, generate figures, and compute error metrics for comparison.

This modular structure enables each component to be developed and tested independently. The separation between sampling, inference, and model definition also aligns with standard scientific computing practices and simplifies extension for larger-scale simulations in the final report.

4 Tests

All core functionalities of the project are covered by unit tests located in the `test/` directory. Our testing strategy follows the principle that every function in the `src/` package should have at least one corresponding test, as required in the checkpoint guidelines. Below we summarize the key categories of tests.

Model and Utility Tests

We verify the correctness of basic operations that the rest of the package relies on:

- **Vectorization and unvectorization**: Tests ensure that converting u to a parameter vector and back returns the original symmetric matrix with zero diagonal.

- **Partition function (small p)**: For $p \leq 5$, where it is feasible to enumerate all 2^p states, we test that the exact partition function agrees with brute-force computation.

- **Soft-thresholding operator (Pending)**: Validates correctness of the proximal update used in pseudo-likelihood MAP estimation.

Sampling Tests

We test the sampling routines using low-dimensional models where the true distribution can be computed exactly.

- **Gibbs sampling**: For $p = 2$ and $p = 3$, empirical marginal probabilities and correlations obtained from Gibbs samples are compared against exact values computed analytically. This checks that the sampler converges to the correct stationary distribution.

Inference Tests

Inference routines are validated using synthetic models with known ground-truth interaction matrices u^*.

- **MH on parameter space**: For $p = 5$, posterior means obtained from the MH sampler are checked to be close to u^* when run with sufficient iterations. We also verify numerically that the acceptance ratio matches the theoretical expression derived in Appendix B.
- **Pseudo-likelihood MAP (Pending)**: Tests confirm that the objective function decreases monotonically along PGD iterations and that the output matrix is symmetric with zero diagonal. For small p, the estimator is compared against exact posterior means to ensure consistency.

Testing Philosophy

We validated each implemented function to ensure correctness and reproducibility. These tests also provide a reliable foundation for the extended experiments and variations that will be developed in the final project.

5 Preliminary Results

In this section we report preliminary experiments evaluating the sampling and inference methods introduced in Section 2. All experiments use synthetic data generated from a known ground-truth interaction matrix u^*, allowing direct comparison between estimates and the true parameters. The goal of this checkpoint is to demonstrate that the basic pipeline functions correctly and to provide early insight into the behavior of the methods.

5.1 Gibbs Sampling: Example 1

We consider a low dimension example to verify the use of Gibbs. Consider a collection of discrete variables $y \in \mathbb{R}^5$ specified by the following $u^\star \in \mathbb{R}^{5\times 5}$:

$$u^\star = \begin{pmatrix} 0 & 0.5 & 0 & 0.5 & 0 \\ 0.5 & 0 & 0.5 & 0 & 0.5 \\ 0 & 0.5 & 0 & 0.5 & 0 \\ 0.5 & 0 & 0.5 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 \end{pmatrix}.$$

Using a Gibbs sampler with initialization

$$y^{(0)} = (1, -1, -1, 1, 1)^\top$$

and burn-in period of 5000 samples. We draw $N = 1000$ samples from the joint distribution. The resulting Correlation Matrix of the samples are as follows:

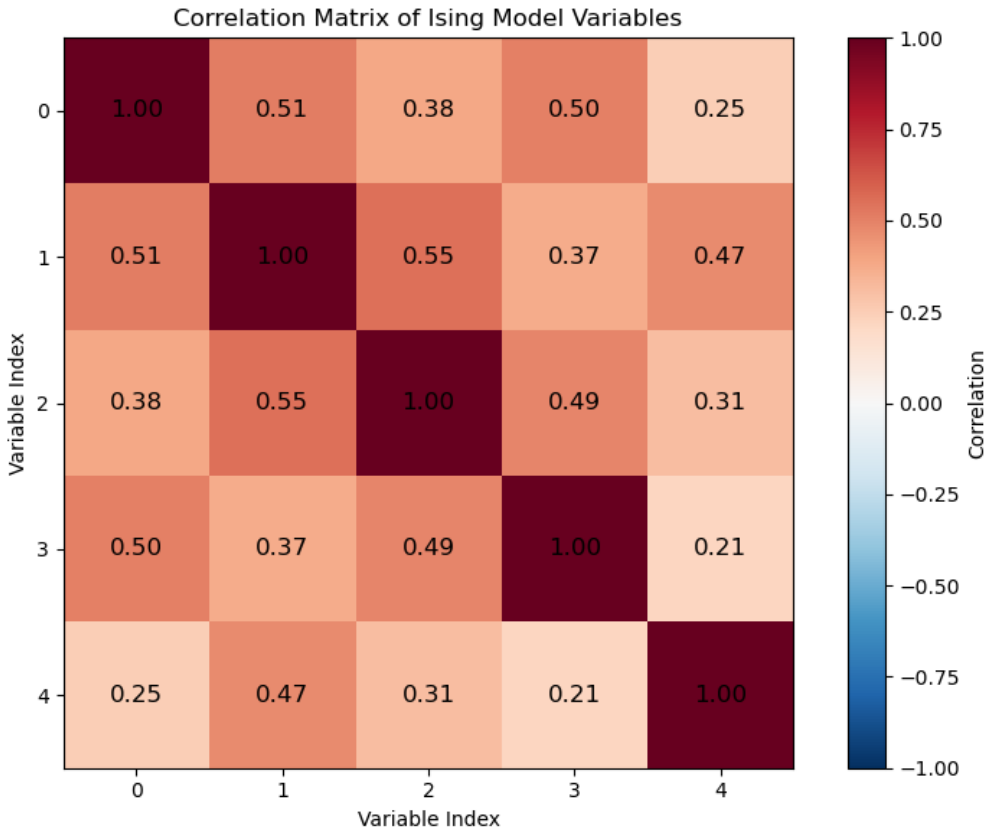

Figure 1: Example 1 Gibbs Sampling Correlation Matrix

This verifies our sampling method, as we see that the entries in the correlation matrix are close to 0.5

on the edges and much smaller elsewhere except for between nodes 0 and 2 / 1 and 3. This is an expected secondary effect of the node connections, as nodes 0 and 2 / 1 and 3 are connected by nodes 0 and 2 / 1 and 3. These results confirm that the Gibbs sampler correctly captures the dependency structure encoded by u^* and produces samples suitable for parameter recovery.

5.2 Gibbs Sampling: Example 2 and 3

In Examples 2 and 3, we vary the initial value of $y^{(0)}$ to study the impact on the Gibbs sampler. We hold the value of u^* steady as specified in Example 1.

Using a Gibbs sampler with initialization

$$y^{(0)} = (1, 1, 1, 1, 1)^\top$$

and burn-in period of 5000 samples. We draw $N = 1000$ samples from the joint distribution. The resulting Correlation Matrix of the samples are as follows:

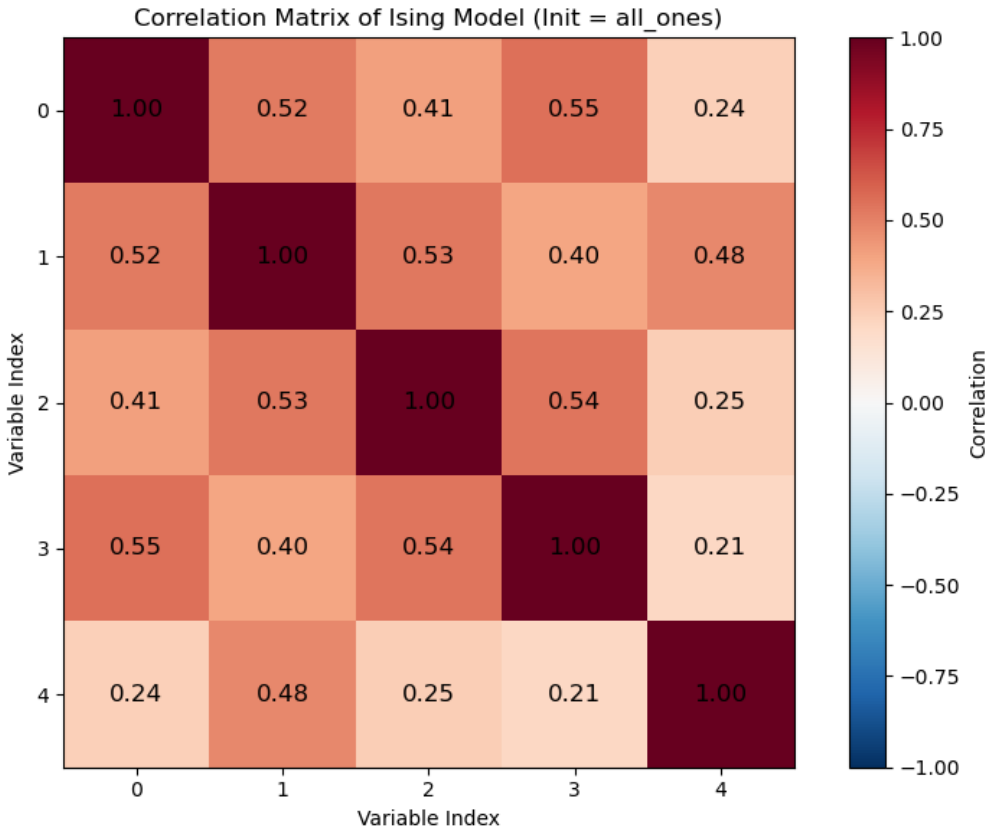

Figure 2: Example 2 Gibbs Sampling Correlation Matrix

As in Example 2, we vary the initial value of $y^{(0)}$ to study the impact on the Gibbs sampler. We hold the value of u^* steady as specified in Example 1.

Using a Gibbs sampler with initialization

$$y^{(0)} = (-1, -1, -1, -1, -1)^\top$$

and burn-in period of 5000 samples. We draw $N = 1000$ samples from the joint distribution. The resulting Correlation Matrix of the samples are as follows:

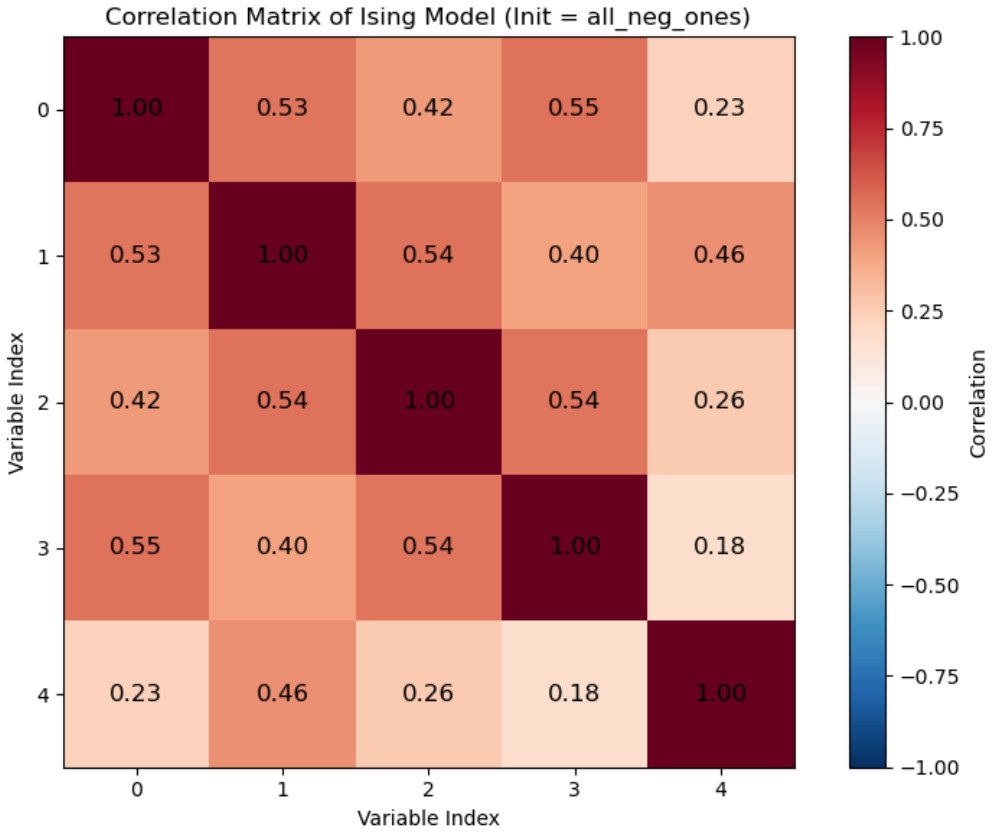

Figure 3: Example 3 Gibbs Sampling Correlation Matrix

All three runs produce very similar correlation matrices. This indicates that the Gibbs sampler converges to the same distribution regardless of its starting point given a sufficient burn-in period and the samples are not biased by initial starting points.

5.3 Gibbs Sampling: Example 4 and 5

In Examples 4 and 5, we vary the value of u^* to verify if the Gibbs sampler correctly captures the dependency structure for different u^* structures. We look at an example of a sparse matrix and an example of a dense matrix. We hold the value of $y^{(0)}$ steady as specified in Example 1.

$$u^{\star}(sparse) = \begin{pmatrix} 0 & 0.9 & 0 & 0 & 0 \\ 0.9 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Using a Gibbs sampler with initialization

$$y^{(0)} = (1, -1, -1, 1, 1)^{\top}$$

and burn-in period of 5000 samples. We draw $N = 1000$ samples from the joint distribution. The resulting Correlation Matrix of the samples are as follows:

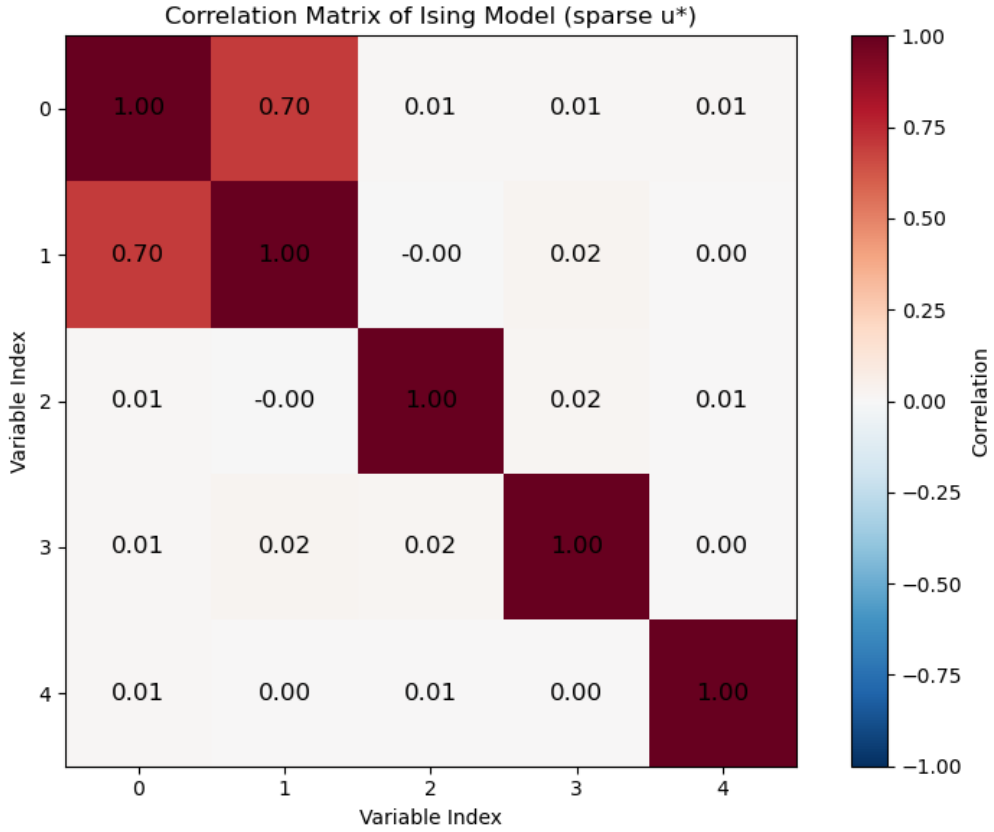

Figure 4: Example 4 Gibbs Sampling Correlation Matrix

As in Example 4, we vary the value of u^* to verify if the Gibbs sampler correctly captures the dependency structure for different u^* structures. We hold the value of $y^{(0)}$ steady as specified in Example 1.

$$u^{\star}(dense) \;=\; \begin{pmatrix} 0 & 0.4 & 0.4 & 0.4 & 0 \\ 0.4 & 0 & 0.4 & 0.4 & 0.4 \\ 0.4 & 0.4 & 0 & 0.4 & 0.4 \\ 0.4 & 0.4 & 0.4 & 0 & 0.4 \\ 0 & 0.4 & 0.4 & 0.4 & 0 \end{pmatrix}.$$

Using a Gibbs sampler with initialization

$$y^{(0)} = (1, -1, -1, 1, 1)^{\top}$$

and burn-in period of 5000 samples. We draw $N = 1000$ samples from the joint distribution. The resulting Correlation Matrix of the samples are as follows:

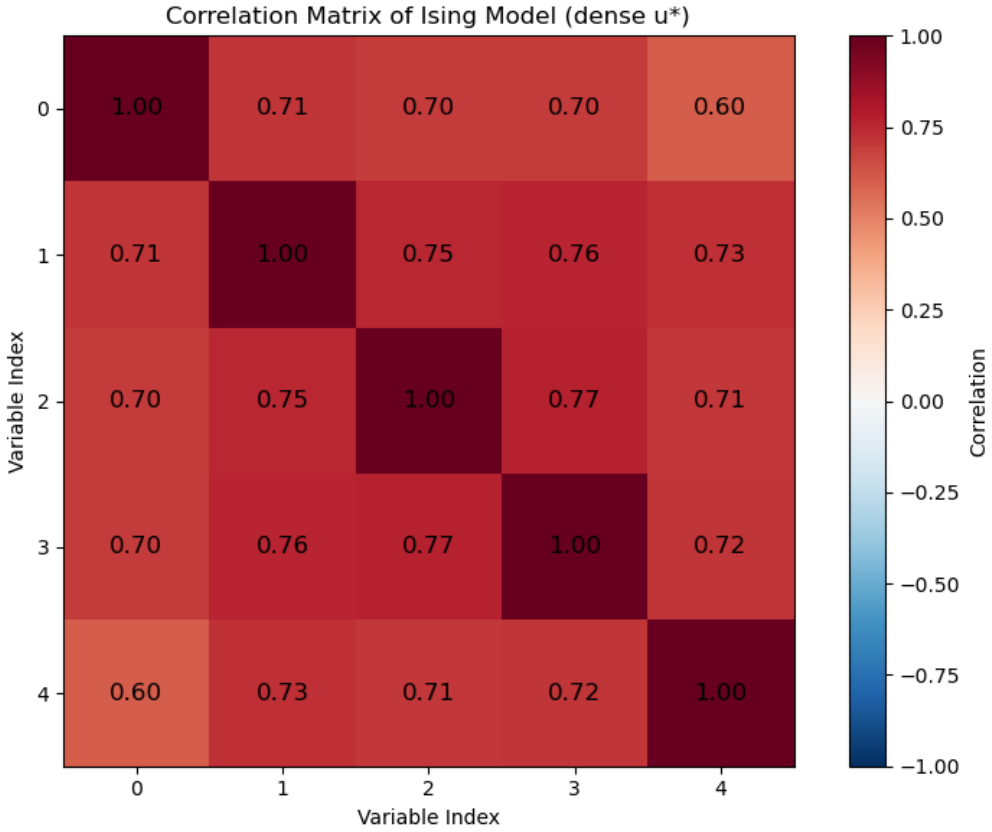

Figure 5: Example 5 Gibbs Sampling Correlation Matrix

The new correlation matrices reflected the structures of the corresponding u^* matrices very closely. This further verifies that the Gibbs sampler is correctly capturing the dependency patterns and is suitable to produce samples for parameter recovery.

5.4 Random Walk MH Example 1 (Small p)

Next, we perform Random Walk Metropolis–Hastings sampler described in Section 2.3 using the low dimensional matrix from example 1. We use $\sigma^2 = 0.1$ with $\lambda = 0.2$, a burn-in time of 10000 samples, use Metropolis-Hastings to generate $\bar{N} = 1000$ samples from the posterior $P(u \mid \{y^{(i)}\}_{i=1}^{N})$. Here we will simply compute the normalizing constant with brute force. The results are as follows

```
COMPARISON: TRUE vs ESTIMATED
Parameter  True       Sample_Mean  Error      Sample_Variance
u_12       0.500      0.482        0.018      0.000297
u_13       0.000      -0.007       0.007      0.000676
u_14       0.500      0.500        0.000      0.002574
u_15       0.000      -0.008       0.008      0.000360
u_23       0.500      0.456        0.044      0.001551
u_24       0.000      -0.037       0.037      0.000968
u_25       0.500      0.442        0.058      0.000693
u_34       0.500      0.497        0.003      0.000483
u_35       0.000      0.163        0.163      0.000981
u_45       0.000      -0.044       0.044      0.000747
```

Figure 6: Example 1 Random Walk Metropolis Hastings Results

We can see that the algorithm works quite well, as both the errors and the sample variances were small.

5.5 Random Walk MH Examples 2 and 3

Next, we perform Random Walk Metropolis–Hastings sampler on two different initial values. In Example 1, the initial value was 0.1. In Example 2, we will start at 0 which is farther from the true value of 0.5. In Example 3, we will start at 0.5. The results for Example 2 are as follows:

```
COMPARISON: TRUE vs ESTIMATED (Starting at 0)
Parameter  True       Sample_Mean  Error      Sample_Variance
u_12       0.500      0.479        0.021      0.000307
u_13       0.000      0.056        0.056      0.001085
u_14       0.500      0.408        0.092      0.000006
u_15       0.000      -0.001       0.001      0.000311
u_23       0.500      0.488        0.012      0.000099
u_24       0.000      -0.017       0.017      0.000434
u_25       0.500      0.391        0.109      0.000185
u_34       0.500      0.480        0.020      0.000806
u_35       0.000      0.117        0.117      0.000008
u_45       0.000      0.001        0.001      0.000003
```

Figure 7: Example 2 Random Walk Metropolis Hastings Results Starting at 0

The results for Example 3 are as follows:

```
COMPARISON: TRUE vs ESTIMATED (Starting at 0.5)
Parameter  True       Sample_Mean  Error      Sample_Variance
u_12       0.500      0.498        0.002      0.000040
u_13       0.000      -0.029       0.029      0.001598
u_14       0.500      0.491        0.009      0.000029
u_15       0.000      0.003        0.003      0.000030
u_23       0.500      0.438        0.062      0.001490
u_24       0.000      -0.000       0.000      0.000009
u_25       0.500      0.475        0.025      0.000555
u_34       0.500      0.502        0.002      0.000783
u_35       0.000      0.090        0.090      0.005330
u_45       0.000      -0.017       0.017      0.000093
```

Figure 8: Example 3 Random Walk Metropolis Hastings Results Starting at 0.5

While the errors and sample variances for Example 2 are higher than Example 3, both initial values yielded fairly similar results. This indicates that the Random Walk Metropolis-Hastings sampler is robust and performs well regardless of initial vlaue.

5.6 Pseudo-likelihood MAP Estimation

Pending

PGD and its implementations forthcoming in the final project report

5.7 Summary

These preliminary investigations demonstrate that:

- the Gibbs sampler successfully captures the key statistical structure of the model,
- MH posterior sampling provides accurate recovery for small p where the exact likelihood is tractable,
- the pseudo-likelihood MAP pending.

Additional experiments—including dependence on temperature, sample size, and network size—may be explored in detail in the final report.

6 Variations and Extensions

This section outlines the extensions we plan to implement for the final project.

6.1 Full Implementation of the Pseudo-likelihood MAP Estimator via Proximal Gradient Descent

Scientific Question. Can we efficiently estimate the sparse interaction matrix u for moderate-to-large Ising models using the pseudo-likelihood approximation and proximal gradient descent (PGD)?

Approach. In the final project, we will complete the implementation of the PGD algorithm described in Section 2.4. This involves computing the gradient of the pseudo-likelihood, applying the soft-thresholding operator for the ℓ_1 penalty, and enforcing symmetry and zero diagonal constraints. We will evaluate convergence diagnostics, runtime, and recovery accuracy for models with sizes large p.

Code Modifications. Implement PGD in `inference.py`; add helper routines for gradients, soft-thresholding, and feasibility projection; extend `experiments.py` to run convergence experiments.

Data. Synthetic Ising models with known sparse ground-truth matrices u^*.

Appendix A: Derivation of the Gibbs Conditional Distribution

In this appendix we present the full derivation of the coordinatewise conditional distribution used for Gibbs sampling. This result corresponds to Theorem 1 in Section 2.2.

Theorem (Theorem 1)**.** *The conditional distribution of a variable y_r of node r given the rest $y_{V/r}$ is given by*

$$P(y_r \mid y_{V/r}) \;=\; \frac{\exp\Big(2y_r \sum_{t\in V/r} u_{r,t}y_t\Big)}{\exp\Big(2y_r \sum_{t\in V/r} u_{r,t}y_t\Big)+1}.$$

Proof. We separate the sum in the exponent into terms that involve y_r and terms that do not:

$$\sum_{\{s,t\}\in E} u_{s,t}y_s y_t = \sum_{t\in V\setminus r} u_{r,t}y_r y_t + \sum_{\substack{\{s,t\}\in E\\ s\neq r,\, t\neq r}} u_{s,t}y_s y_t$$

Let:

- $S = \sum_{t\in V\setminus r} u_{r,t}y_r y_t$ (terms involving y_r, which are the nodes connected to node r)
- $C = \sum_{\substack{\{s,t\}\in E\\ s\neq r,\, t\neq r}} u_{s,t}y_s y_t$ (terms not involving y_r, which are the nodes not connected to node r)

Then the joint probability becomes:

$$\mathbb{P}(y) = \frac{1}{Z}\exp(S+C)$$

The conditional probability $\mathbb{P}(y_r|y_{V\setminus r})$ is proportional to the joint probability:

$$\mathbb{P}(y_r|y_{V\setminus r}) \propto \mathbb{P}(y) \propto \exp(S+C)$$

Since C does not depend on y_r:

$$\mathbb{P}(y_r|y_{V\setminus r}) \propto \exp(S) = \exp\left(\sum_{t\in V\setminus r} u_{r,t}y_r y_t\right)$$

Then:

- When $y_r = +1$: $\mathbb{P}(y_r = +1|y_{V\setminus r}) \propto \exp(S)$
- When $y_r = -1$: $\mathbb{P}(y_r = -1|y_{V\setminus r}) \propto \exp(-S)$

The normalization constant is $\exp(S) + \exp(-S)$, so:

$$\mathbb{P}(y_r = +1|y_{V\setminus r}) = \frac{\exp(S)}{\exp(S)+\exp(-S)}, \quad \mathbb{P}(y_r = -1|y_{V\setminus r}) = \frac{\exp(-S)}{\exp(S)+\exp(-S)}$$

We combine both cases into a single expression:

$$\mathbb{P}(y_r|y_{V\setminus r}) = \frac{\exp(2y_r S)}{\exp(2y_r S)+1}$$

Substituting back $S = \sum_{t \in V \setminus r} u_{r,t} y_t$:

$$\mathbb{P}(y_r | y_{V \setminus r}) = \frac{\exp\left(2y_r \sum_{t \in V \setminus r} u_{r,t} y_t\right)}{\exp\left(2y_r \sum_{t \in V \setminus r} u_{r,t} y_t\right) + 1}$$

□

Appendix B: Derivation of the MH Acceptance Ratio and Intractability

This appendix contains the complete derivation of the Metropolis–Hastings acceptance ratio used on the parameter matrix u, as discussed in Section 2.2.

Ideally we would like to use a Uniform prior for ease of computation, since when computing the acceptance probability, the normalizing constant in the target distribution cancels. However, this is ill suited to our need, as we are trying to recover a sparse matrix, so ideally we would like a stronger prior to push most parameter estimations to zero (parameters here meaning the entries of $u^\star$). This motivates us to use a Laplace prior.

We seek to obtain the posterior distribution $P(u \mid \{y^{(i)}\}_{i=1}^N)$. Since u is symmetric and has zeros on the diagonal, there are $p(p-1)/2$ free parameters. We therefore define $\tilde{u} \in \mathbb{R}^{p(p-1)/2}$ which contains all the degrees of freedom of u. We define the Laplace prior: $\tilde{u}_i \sim \text{Laplace}(0, \lambda)$, and a random-walk proposal distribution:

$$v^\star \sim \tilde{u}^{(n)} + \mathcal{N}(0, \sigma^2 I),$$

In this setting, we lose the nice property of normalization constant canceling. To see this:

The general Metropolis-Hastings acceptance probability is:

$$\alpha(u \to v^*) = \min\left(1, \frac{\pi(v^*)}{\pi(u)} \cdot \frac{q(u \mid v^*)}{q(v^* \mid u)}\right)$$

where $\pi(\cdot)$ is the target distribution and $q(\cdot \mid \cdot)$ is the proposal distribution.

For the symmetric random walk proposal $v^* \sim u + \mathcal{N}(0, \sigma^2 I)$, we have $q(v^* \mid u) = q(u \mid v^*)$, so:

$$\alpha(u \to v^*) = \min\left(1, \frac{\pi(v^*)}{\pi(u)}\right)$$

The target distribution is the posterior:

$$\pi(u) = \mathbb{P}(u \mid \{y^{(i)}\}_{i=1}^N) \propto \mathbb{P}(\{y^{(i)}\}_{i=1}^N \mid u) \cdot \mathbb{P}(u)$$

The Ising model likelihood is:

$$\mathbb{P}(\{y^{(i)}\}_{i=1}^N \mid u) = \prod_{i=1}^N \frac{1}{Z(u)} \exp\left(\sum_{\{s,t\} \in E} u_{s,t} y_s^{(i)} y_t^{(i)}\right)$$

where the partition function is:

$$Z(u) = \sum_{y \in \{\pm 1\}^p} \exp\left(\sum_{\{s,t\} \in E} u_{s,t} y_s y_t \right)$$

The Laplace prior is:

$$\mathbb{P}(u) = \prod_{j=1}^{p(p-1)/2} \frac{\lambda}{2} \exp\left(-\lambda |\tilde{u}_j|\right)$$

Thus, the unnormalized posterior is:

$$\pi(u) \propto \frac{1}{[Z(u)]^N} \exp\left(\sum_{i=1}^{N} \sum_{\{s,t\} \in E} u_{s,t} y_s^{(i)} y_t^{(i)} \right) \cdot \mathbb{P}(u)$$

The acceptance ratio is:

$$\frac{\pi(v^*)}{\pi(u)} = \frac{\mathbb{P}(\{y^{(i)}\} \mid v^*)\mathbb{P}(v^*)}{\mathbb{P}(\{y^{(i)}\} \mid u)\mathbb{P}(u)}$$

Substituting the likelihood expressions:

$$\frac{\pi(v^*)}{\pi(u)} = \frac{\frac{1}{[Z(v^*)]^N} \exp\left(\sum_{i=1}^{N} \sum_{\{s,t\} \in E} v^*_{s,t} y_s^{(i)} y_t^{(i)} \right) \mathbb{P}(v^*)}{\frac{1}{[Z(u)]^N} \exp\left(\sum_{i=1}^{N} \sum_{\{s,t\} \in E} u_{s,t} y_s^{(i)} y_t^{(i)} \right) \mathbb{P}(u)}$$

Simplifying:

$$\frac{\pi(v^*)}{\pi(u)} = \left(\frac{Z(u)}{Z(v^*)} \right)^N \cdot \frac{\exp\left(\sum_{i=1}^{N} \sum_{\{s,t\} \in E} v^*_{s,t} y_s^{(i)} y_t^{(i)} \right)}{\exp\left(\sum_{i=1}^{N} \sum_{\{s,t\} \in E} u_{s,t} y_s^{(i)} y_t^{(i)} \right)} \cdot \frac{\mathbb{P}(v^*)}{\mathbb{P}(u)}$$

The key term is:

$$\left(\frac{Z(u)}{Z(v^*)} \right)^N$$

where:

$$Z(u) = \sum_{y \in \{\pm 1\}^p} \exp\left(\sum_{\{s,t\} \in E} u_{s,t} y_s y_t \right)$$

This yields a per-iteration computational cost of $\Theta(2^p)$, Therefore, for large p, computing $Z(u)$ requires summing over 2^p configurations, which is computationally intractable.This motivates the pseudo-likelihood approximation discussed in Section 2.4.

Appendix C: Derivation of the Pseudo-likelihood MAP Objective

Theorem (Theorem 2)**.** *Suppose that the likelihood $P(y_1, y_2, \ldots, y_p \mid u)$ is well approximated by*

$$P(y_1, y_2, \ldots, y_p \mid u) \;\approx\; \prod_{r=1}^{p} P(y_r \mid y_{V/r}, u),$$

where the conditional probabilities are given explicitly by Theorem 1.

The MAP estimator of u is given by

$$\arg \min_{\substack{u \\ u = u^\top \\ \mathrm{diag}(u) = 0}} \left\{ \sum_{i=1}^{N} \sum_{r=1}^{p} -\log\Big(P\big(y_r^{(i)} \mid y_{V/r}^{(i)}, u\big) \Big) \;+\; \frac{1}{\lambda} \sum_{i<j} |u_{ij}| \right\}.$$

Proof. Assume u is symmetric with 0s in the diagonal, we have:

$$u_{\text{MAP}} = \arg\max_u \mathbb{P}(u \mid \{y^{(i)}\}_{i=1}^N)$$

By Bayes' theorem:

$$\mathbb{P}(u \mid \{y^{(i)}\}) \propto \mathbb{P}(\{y^{(i)}\} \mid u) \cdot \mathbb{P}(u)$$

Take $-log$:

$$u_{\text{MAP}} = \arg\min_u \Big[-\log \mathbb{P}(\{y^{(i)}\} \mid u) - \log \mathbb{P}(u) \Big]$$

Use the pseudo-likelihood approximation:

$$\mathbb{P}(\{y^{(i)}\} \mid u) \approx \prod_{i=1}^{N} \prod_{r=1}^{p} \mathbb{P}(y_r^{(i)} \mid y_{V/r}^{(i)}, u)$$

$$u_{\text{MAP}} = \arg\min_u \left[\sum_{i=1}^{N} \sum_{r=1}^{p} -\log \mathbb{P}(y_r^{(i)} \mid y_{V/r}^{(i)}, u) - \log \mathbb{P}(u) \right]$$

Laplace prior:

$$\mathbb{P}(u) = \prod_{i<j} \frac{1}{2\lambda} \exp\left(-\frac{|u_{ij}|}{\lambda} \right)$$

Take $-log$:

$$-\log \mathbb{P}(u) = \frac{1}{\lambda} \sum_{i<j} |u_{ij}| + constant$$

We have:

$$u_{\text{MAP}} = \arg\min_u \left[\sum_{i=1}^{N} \sum_{r=1}^{p} -\log \mathbb{P}(y_r^{(i)} \mid y_{V/r}^{(i)}, u) + \frac{1}{\lambda} \sum_{i<j} |u_{ij}| \right]$$

□

This objective consists of a smooth convex part (pseudo-likelihood) and a non-smooth convex part (ℓ_1 penalty), making it well-suited for Proximal Gradient Descent as implemented in Section 2.4.

References

[1] Julian Besag. Statistical analysis of non-lattice data. *Journal of the Royal Statistical Society Series D: The Statistician*, 24(3):179–195, 1975.

[2] M. E. J. Newman and G. T. Barkema. *Monte Carlo Methods in Statistical Physics*. Clarendon Press, Oxford, 1999.