HW3

Simulation of the Ising Model and Convergence Monitoring

陳凱騫

2024-12-26

Table of contents

Abstract

This report explores the simulation of a 2D Ising model on a 32×32 lattice using Gibbs Sampling and Metropolis-Hastings algorithms. Various parameters were tested, including different temperatures, boundary conditions, and initial configurations. Convergence was assessed using Monte Carlo Standard Error (MCSE) and the Gelman-Rubin Statistic. This analysis provides insights into algorithmic behavior and the relationship between temperature and spin alignment.

Introduction

The Ising model is a fundamental framework in statistical mechanics for modeling ferromagnetic behavior. The probability distribution is expressed as:

$$P(x) \propto \text{exp}\left(-\frac{\epsilon(x)}{T}\right), \quad \epsilon(x) = \sum_{\langle i,j \rangle} x_i x_j,$$

where T represents temperature, and $\langle i, j \rangle$ are pairs of neighboring spins.

Key tasks include: 1. Implementing Gibbs Sampling and Metropolis-Hastings algorithms. 2. Simulating spin configurations under various conditions. 3. Assessing convergence via MCSE and Gelman-Rubin diagnostics.

Reference: Ising Model Implementation.

Methodology

Initialization and Simulation Setup

Ising Model Functions

Monitoring Convergence

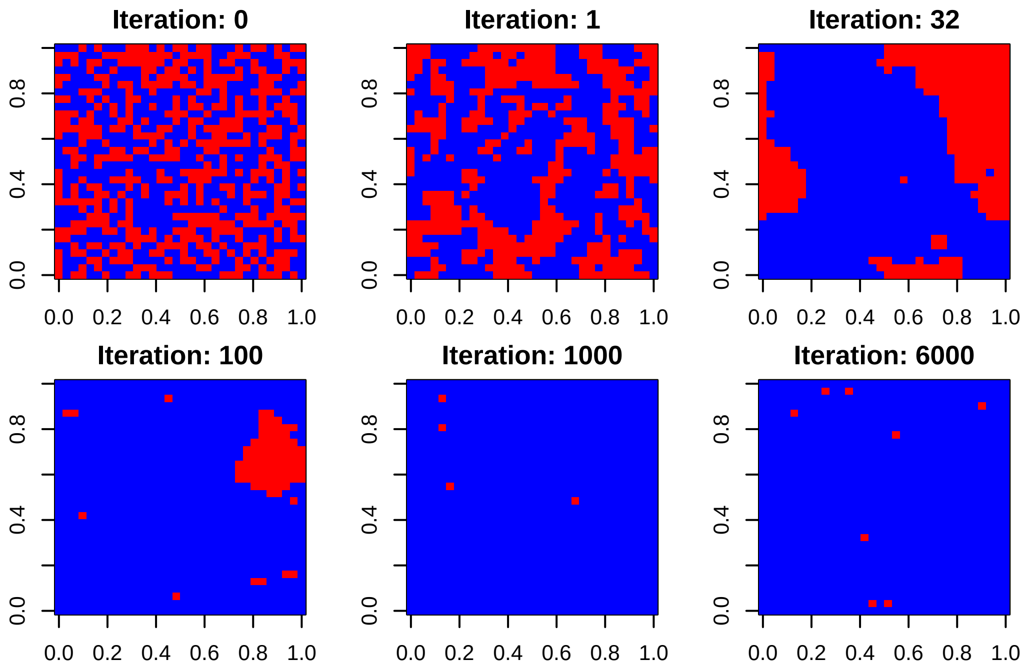
```
monitor_convergence <- function(magnetizations) {
  n <- length(magnetizations)
  means <- cumsum(magnetizations) / (1:n)
  mcse <- sqrt(cumsum((magnetizations - means)^2) / (1:n))
  data.frame(Iteration = 1:n, Mean = means, MCSE = mcse)
}
```

Parameters and Execution

```
L <- 32
steps <- 6000
temperatures <- c(1.5, 2.5, 3.5)
results <- list()
```

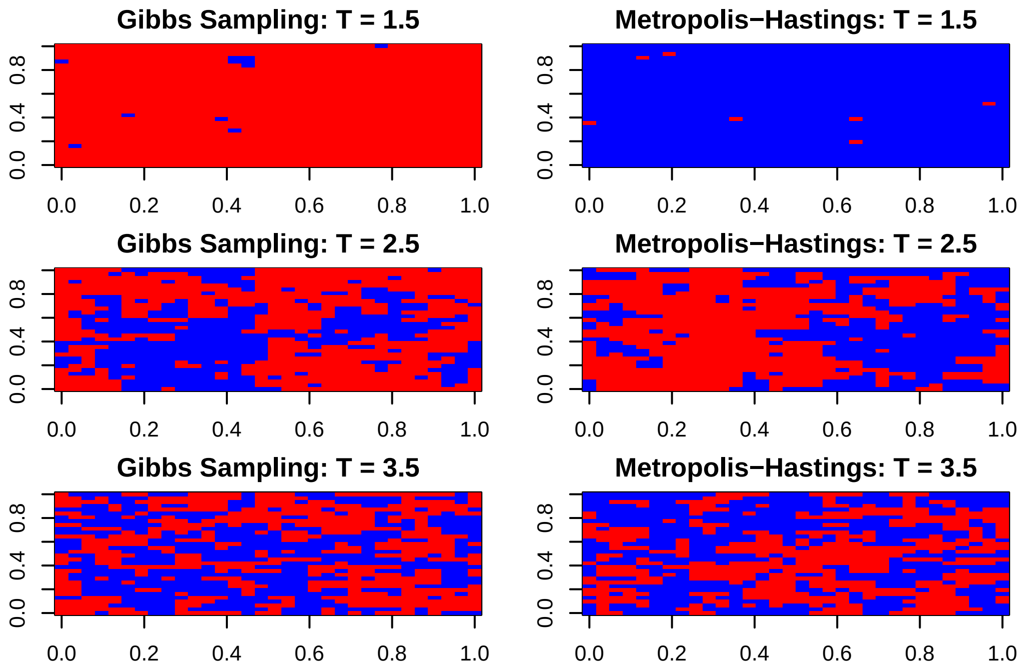
Results and Discussion

```
# 固定參數
L <- 32
T_fixed <- 1.5
beta <- 1 / T_fixed
iterations_to_visualize <- c(0, 1, 32, 100, 1000,6000)  # 要展示的迭代次數

# 初始化晶格
lattice <- initialize_lattice(L)
snapshots <- list()

# 記錄初始狀態
snapshots[["0"]] <- lattice

# 進行 Gibbs Sampling 模擬
for (iter in 1:max(iterations_to_visualize)) {
  lattice <- gibbs_step(lattice, beta)

  if (iter %in% iterations_to_visualize) {
    snapshots[[as.character(iter)]] <- lattice
  }
}

# 繪製不同迭代次數的自旋配置
par(mfrow = c(2, 3), mar = c(2, 2, 2, 2))  # 2行3列的子圖排列
for (iter in names(snapshots)) {
  image(t(apply(snapshots[[iter]], 2, rev)), col = c("blue", "red"),
        main = paste("Iteration:", iter))
}
```



Visualization of Spin Configurations

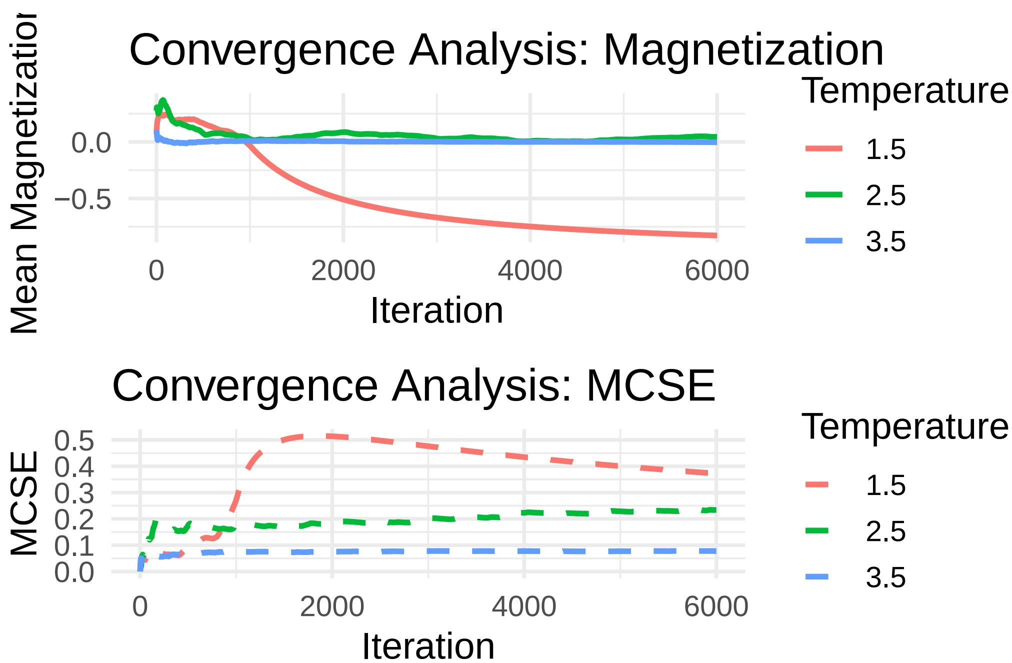
```
par(mfrow = c(3, 2), mar = c(2, 2, 2, 2))  # 將圖分為3行2列展示
for (T in temperatures) {
  beta <- 1 / T
  lattice_gibbs <- initialize_lattice(L)
  lattice_mh <- initialize_lattice(L)

  for (step in 1:steps) {
    lattice_gibbs <- gibbs_step(lattice_gibbs, beta)
    lattice_mh <- metropolis_step(lattice_mh, beta)
  }

  image(t(apply(lattice_gibbs, 2, rev)), col = c("blue", "red"), main = paste("Gibbs Sampling: T =", T))
  image(t(apply(lattice_mh, 2, rev)), col = c("blue", "red"), main = paste("Metropolis-Hastings: T =", T))
}
```



Convergence Analysis

```r
# MCSE 和 Mean 分為不同子圖展示
library(ggplot2)
library(gridExtra)
# 初始化收集數據的容器
convergence_data <- data.frame()

# 迭代每個溫度，計算並保存數據
for (T in temperatures) {
 lattice <- initialize_lattice(L)
 magnetizations <- numeric(steps)

 for (step in 1:steps) {
  lattice <- gibbs_step(lattice, 1 / T)
  magnetizations[step] <- mean(lattice)
 }

 conv_metrics <- monitor_convergence(magnetizations)
 conv_metrics$Temperature <- T
 convergence_data <- rbind(convergence_data, conv_metrics)
}

# 繪製磁化率曲線
p1 <- ggplot(convergence_data, aes(x = Iteration, y = Mean, color = as.factor(Temperature))) +
 geom_line(size = 1) +
 labs(
  title = "Convergence Analysis: Magnetization",
  x = "Iteration",
  y = "Mean Magnetization",
  color = "Temperature"
 ) +
```

```
  theme_minimal(base_size = 14)

# 繪製MCSE曲線
p2 <- ggplot(convergence_data, aes(x = Iteration, y = MCSE, color = as.factor(Temperature))) +
  geom_line(size = 1, linetype = "dashed") +
  labs(
    title = "Convergence Analysis: MCSE",
    x = "Iteration",
    y = "MCSE",
    color = "Temperature"
  ) +
  theme_minimal(base_size = 14)

# 組合圖
grid.arrange(p1, p2, ncol = 1)
```



Observations and Conclusion

1. **Temperature Effects**: Low temperatures ($T = 1.5, 2.5$) promoted spin alignment. Higher temperatures ($T = 3.5$) led to random configurations.
2. **Algorithm Comparison**: Both algorithms achieved similar results but differed in convergence speed.
3. **Diagnostics**: Convergence metrics verified simulation reliability.

Reference links for further details: - Convergence Detection.