Monte Carlo simulation of hyper-parameters in hidden Markov random fields

Xin Luo

Master of Science in Mathematics (for international students)
Submission date: June 2014
Supervisor: Håkon Tjelmeland, MATH

Norwegian University of Science and Technology
Department of Mathematical Sciences

Preface

This thesis is the final part of my Master of Science degree in Mathematics at the Norwegian University of Science and Technology. The author shows his sincere gratitude for the careful supervision and the excellent help from Professor Håkon Tjelmeland, and also thank my parents and friends for their emotional support over past one year. The technical aid of firmware is very important for the scientific computing as well, so the author is grateful to technical staffs working in the Department of Mathematical Sciences.

Xin Luo

Trondheim

May 2014

Abstract

Bayesian inference is an important branch in statistical sciences. The subject of this thesis is about Bayesian inference in models for a vast number of dependent objects. The fundamental scheme used for the inference in this thesis is Markov chain Monte Carlo (MCMC) methods. However, conventional MCMC approaches are not feasible for resolving issues resulting from intractable normalizing constants. Several methods, such as the single auxiliary variable method and the exchange algorithm, can avoid computer-intensive calculations of intractable normalizing constants, but they are merely applicable to the case without latent variables. Based on Markov random fields (MRFs), we propose two different strategies to overcome the difficulty as the result of noisy observations. The first strategy named simulation strategy I combines the exchange algorithm with MCMC techniques to simulate from the distribution of interest. For the second strategy called simulation strategy II, we use more auxiliary variables with block updates, and adopt partially ordered Markov models (POMMs) to approximate the proposal distributions in order to get rid of high dimensional calculations. We empirically compare these two strategies according to the diagnostics of convergence, mixing and CPU time requirements coming from our simulation examples. For the mixing assessment, we use the Effective Sample Size (ESS). Simulation strategy II using block updates can generate stationary Markov chains having better mixing within fewer number of iterations (≤ 200), and maximum ratio of ESS in simulation strategy II to ESS in simulation strategy I is 6.27. Simulation strategy I uses less CPU time per iteration irrespective of convergence to stationarity or simulations of equilibrium samples, but simulation strategy I needs more iterations (> 200), and for large absolute values of parameters it produces poor mixing compared with simulation strategy II.

Contents

1 Introduction

In general, Bayesian inference is a crucial branch of statistical theories, known as Bayesian statistics. The subject of this thesis is about Bayesian inference in models for a vast number of dependent objects, for instance, in the areas of uncertainty in reservoir evaluation, image restoration and genetic analysis. Monte Carlo methods are a set of useful tools applied to Bayesian analysis based on sophisticated models. The art of designing Monte Carlo algorithms chiefly resides in the adoption of an adequate trade-off between the simplicity of implementation and the difficult incorporation of important characteristics of the target distribution. With respect to the type of model, we focus on Markov random fields (MRFs) (Hammersley and Clifford, 1971), specifically, binary Markov random fields, which is one of undirected graphical models most prevailing for spatial data. A common scenario is that observations of MRFs are completely given, but in reality, we usually only have noisy observations while the values of latent variables are unknown. As a result, the estimation of parameters of the model we choose implies a significant computational challenge in terms of either a Bayesian or a frequentist perspective.

Suppose $y \in \mathcal{Y}$ indicates noisy observations and $x \in \mathcal{X}$ hidden variables of discrete type. Our target is to estimate the parameters of interest $\theta \in \Theta$. Given only the observations y, using the Bayesian approach we obtain the joint posterior distribution of interest

$$\pi(\theta, x|y) \propto \pi(x, y, \theta) = \pi(\theta)\pi(x|\theta)\pi(y|x, \theta),$$

where $\pi(\theta)$ is the prior distribution of θ, $\pi(x|\theta)$ represents the likelihood of a Markov random field with respect to x, and there is a relation between random variables x and observations y according to $\pi(y|x, \theta)$.

After obtaining $\pi(\theta, x|y)$, it is in principle possible to calculate the marginal distribution $\pi(\theta|y) = \sum_{x \in \mathcal{X}} \pi(\theta, x|y)$, but the direct manipulation (summation) is often difficult especially when x is multi-dimensional, so putting forward an alternative method is necessary. Monte Carlo methods are useful for such problems, but one problem with applying Monte Carlo integration is obtaining samples from a complex probability distribution. Attempts to solve this problem are the roots of Markov Chain Monte Carlo (MCMC) methods. In particular, The MCMC approach is our preference since using this method can tremendously increase the possibility of practical computation based on computer simulations. One of the most important factors concerning how to design Monte Carlo algorithms to sample from $\pi(\theta|y)$ relies on the appropriate choice of proposal distributions. As a rule of thumb, the proposal distributions resulting in efficient algorithms should consist of some properties such as capturing the major characteristics (e.g. scales or dependence structures) of $\pi(\theta|y)$ and being easy to sample from.

In this thesis, we adopt MCMC methods which are a subset of Monte Carlo methods. Under ordinary configurations of MCMC, normalizing constants vanish when computing acceptance probabilities, but in some models, such as the Ising model (Ising, 1925) defined on an MRF, their normalizing constants do not cancel since they are functions of parameters of interest, and calculating such constants involves high dimensional summation or integration. Hence, it is computationally difficult to utilize directly the scheme of Metropolis–Hastings (Hastings, 1970) or Gibbs sampling (Geman and Geman, 1984). Maximum likelihood estimation or Bayesian posterior distributions for parameters of $\pi(\theta|x)$ becomes consequently infeasible by straightforward approaches. However, several strategic and ingenious techniques have been proposed in the literature. There are systematically several perspectives from which we should take into account the methods as follows. One is to calculate approximations instead of exact values. To be more specific, Besag (1974) proposed the definition of pseudo-likelihood functions and later Heikkinen and Hogmander (1994) and Huang and Ogata (2002) defined different versions of such functions. For small lattices, Friel and Rue (2007) came up with approximations using forward-backward algorithm (Reeves and Pettitt, 2004) in lieu of the exact computations. In addition, Møller et al. (2006) proposed an efficient MCMC approach with auxiliary variables so that normalizing constants cancel from Metropolis–Hastings ratios. Murray et al. (2006) provided a generalization of Møller et al. (2006) that obtains better acceptance probabilities and removes the need to estimate model parameters before sampling starts. This method is usually called the exchange algorithm. Furthermore, a wide range of MCMC estimation techniques for normalizing constants were proposed by Chen and Shao (1999) and path sampling by Gelman and Meng (1998). Green and Richardson (2001) and Berthelsen and Møller (2003) applied these estimation techniques in practical applications. All they have done are only related to non-hidden variables without noisy observations, i.e. $\pi(\theta|x) \propto \pi(\theta)\pi(x|\theta)$.

In the present report, given noisy data, we concentrate on how to simulate from $\pi(\theta, x|y)$ with good mixing and fast convergence based on the MCMC technique. Due to the difficulty in computing normalizing constants of the autologistic model (Besag, 1974), we incorporate the exchange algorithm into our algorithm and use partially ordered Markov models (Cressie and Davidson, 1998) to approximate the proposal distributions (Tjelmeland and Austad, 2012) so as to avoid high dimensional calculations. Afterwards, we compare the proposed algorithm with blocking with the alternative without blocking on the basis of the quantities relating to total CPU times and the Effective Sample Size (ESS) (Kass et al., 1998).

The remainder of this thesis is organized as follows. In Section 2, we introduce background theories including MCMC, MRFs, perfect simulation, the exchange algorithm and POMMs that are used for our methodologies. In section 3, we specify and discuss two different simulation strategies. In Section 4, we present simulation examples, and discuss the results. Relevant conclusions and remarks are given in Section 5.

2 Background theory

In this section, we present some background theories for MCMC, MRFs, perfect simulation, the exchange algorithm and POMMs, and briefly describe how these knowledge can be used for our purpose.

2.1 Markov chain Monte Carlo method

In statistics, Markov chain Monte Carlo (MCMC) methods are a class of algorithms using local information for sampling from probability distributions based on devising a Markov chain that has the desired distribution as its equilibrium distribution. In this section we briefly discuss how MCMC is used to sample from a given distribution $\pi(x), x \in \mathcal{X}$. For more thorough introductions on the topic, one can read other literatures: Hastings (1970), Geman and Geman (1984), Carlin and Chib (1995) and Berg (2004).

For simplicity assume x to be a vector of discrete random variables which is the setting carried out in the following sections, but MCMC can also be used to sample continuous random variables. MCMC generates samples from $\pi(x)$ by simulating a Markov chain $\{x^b\}_{b=0}^{\infty}, x^b \in \mathcal{X}$ with limiting distribution identical to $\pi(x)$, so after a sufficiently large number of Markov chain steps the generated x^b is essentially from the distribution $\pi(x)$. There are different approaches regarding how to construct the transition matrix of a Markov chain, but clearly the most frequently used strategy results in the so-called Metropolis–Hastings (MH) algorithm, and in the following we constrain our attention to this setup.

Let $Q(x'|x) = \Pr(x \to x')$ for $x, x' \in \mathcal{X}$ be the transition matrix of a Markov chain. Sufficient conditions for which $\pi(x)$ is the limiting distribution of the Markov chain are that the Markov chain is irreducible and aperiodic, and that $Q(x'|x)$ fulfills the detailed balance condition

$$\pi(x)Q(x'|x) = \pi(x')Q(x|x') \text{ for all } x, x' \in \mathcal{X}. \tag{1}$$

In the MH algorithm, $Q(x'|x)$ is defined as

$$Q(x'|x) = q(x'|x)\alpha(x'|x) \text{ for all } x \neq x',$$

where $q(x'|x)$ is a proposal probability from x to x', and $\alpha(x'|x)$ is an acceptance probability given by

$$\alpha(x'|x) = \min\left\{\frac{\pi(x')q(x|x')}{\pi(x)q(x'|x)}, 1\right\}. \tag{2}$$

It is easy to show that with this expression for $\alpha(x'|x)$, the detailed balance condition in (1) is fulfilled, so that $\pi(x)$ is the limiting distribution of this Markov chain.

Thus, provided an initial value x^0, in each iteration of the MH algorithm, first a candidate value x' is proposed for the next sample dependent on the current one x^b by the transition probability function $q(x'|x^b)$. Then, with the probability $\alpha(x'|x^b)$, the candidate is accepted, i.e. $x^{b+1} = x'$, and otherwise rejected, i.e. $x^{b+1} = x^b$. Following an adequately long burn-in period, the chain has approached its stationary distribution and samples x^b are essentially generated from $\pi(x)$.

Different strategies can be used to construct the proposal distribution $q(x'|x)$, and often a combination of proposal distributions is used within one MH algorithm. This means that several proposal distributions $q_k(x'|x), k = 1, 2, \ldots, K$ are defined and used within the same

MH algorithm. For example, $q_1(x'|x)$ is used in the first iteration, $q_2(x'|x)$ in the second iteration and so on until $q_K(x'|x)$ is used in the Kth iteration. In iteration $K+1$ one adopts $q_1(x'|x)$ again, in iteration $K+2$ one adopts $q_2(x'|x)$ and so on. In the following we discuss some strategies for constructing proposal distributions that we use in MH algorithms in subsequent sections of this thesis.

Assume x to be n-dimensional and write $x = (x_1, \ldots, x_n)$. A single-site Gibbs update is proposing a change for only one of the components of x. Which component to propose a new value for can be drawn at random or be decided deterministically. Now assume that the component to be changed or updated is numbered i, x_i. The single-site Gibbs update defines the potential new state $x' = (x'_1, \ldots, x'_n)$ by setting $x'_j = x_j$ for all $i \neq j$ and sample x'_i from the full conditional $\pi(x_i|x_j, j \neq i)$. Inserting this proposal distribution into the general expression for the acceptance probability $\alpha(x'|x)$ given in (2), it is simple to demonstrate that for single-site Gibbs updates one always has $\alpha(x'|x) = 1$. Hence, with single-site Gibbs updates the potential new states are always accepted.

As the generalization of the single-site Gibbs update, we turn to single-site updates in general and introduce the basics of such updates. Based on the conditions and notations used in single-site Gibbs updates above, we take on a distribution $q(x'_i|x)$ from which x'_i can be conveniently sampled instead of using the full conditional $\pi(x_i|x_j, j \neq i)$ that perhaps leads to difficulties in analytical derivations or practical implementations, and set $x'_j = x_j$ for all $j \neq i$. Consequently, the acceptance probability given in (2) is

$$\begin{aligned}\alpha(x'|x) &= \min\left\{\frac{\pi(x_1, \ldots, x_{i-1}, x'_i, x_{i+1}, \ldots, x_n)q(x_i|x_1, \ldots, x_{i-1}, x'_i, x_{i+1}, \ldots, x_n)}{\pi(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n)q(x'_i|x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_n)}, 1\right\} \\ &\neq 1.\end{aligned}$$

In particular, by setting $q(x_i|x) = \pi(x_i|x_j, j \neq i)$ one gets the single-site Gibbs update discussed above.

In contrast to single-site updates, block updates including block Gibbs updates are widely applicable in many scenarios. If the dimension of a random vector is large, and some or even all of the components of this random vector are potentially highly correlated, using single-site updates makes a Markov chain converge to its stationarity slowly. Thus, we usually utilize blocking schemes which may reduce the time consumption until the chain reaches the

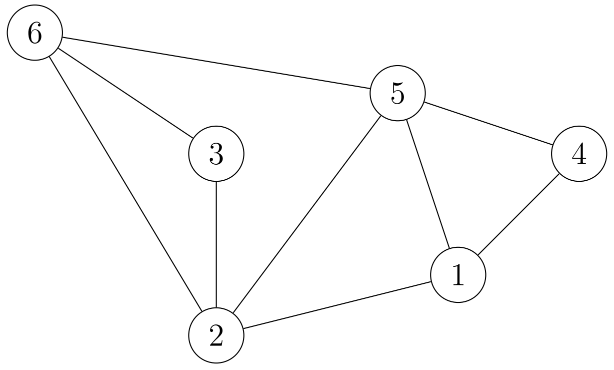


Figure 1: A sample of undirected graph with six nodes and nine edges

equilibrium. For some $A \subseteq \{1, 2, \dots, n\}$, split x into two disjoint parts $x_A = (x_i; i \in A)$ and $x_{-A} = (x_i; i \in \{1, \dots, n\} \backslash A)$. If A has only one component, e.g. $A = \{i\}$ for some $i \in \{1, \dots, n\}$, then we write $x_i = x_A$ and $x_{-i} = x_{-A}$ to simplify the notations. Assume the components of x_A are dependent, then these components can be updated simultaneously when using block updates. That is to say we first propose x'_A from a multivariate proposal distribution $q(x'_A|x)$, and then accept all components of x'_A at the same time with the probability

$$\alpha(x'|x) = \min\left\{\frac{\pi(x'_A, x_{-A})q(x_A|x'_A, x_{-A})}{\pi(x)q(x'_A|x)}, 1\right\} \not\equiv 1.$$

Analogously, the scheme of block Gibbs updates gives a block update with $\alpha(x'|x) = 1$ if one generates x'_A from $\pi(x_A|x_{-A})$ just as for single-site Gibbs updates.

2.2 Markov random fields

Markov random fields (MRFs) are the fundamental model component in this thesis, and in this section we introduce basic definitions and statistical features of MRFs. For more profound details, see relevant literatures such as Hammersley and Clifford (1971) and Kindermann et al. (1980).

Assume that an undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ is defined as a collection of vertices $\mathcal{V} = \{1, 2, \dots, n\}$ and edges $\mathcal{E} \subseteq \{(i, j)|i, j \in \mathcal{V}, i \neq j\}$. A simple example graph is illustrated in Figure 1. Here the vertices are represented as circles and the edges as lines that connect two vertices. In this example, we have the vertices $\mathcal{V} = \{1, 2, 3, 4, 5, 6\}$ and the edges

$\mathcal{E} = \{(1,2),(1,4),(1,5),(2,3),(2,5),(2,6),(3,6)\}$. In general, the neighbors of a vertex $i \in \mathcal{V}$ is $\partial(i) = \{k : (i,k) \in \mathcal{E}\}$, meaning all vertices that directly connect to i. For the graph in Figure 1, for example, the neighbors of vertex 6 is $\partial(6) = \{2,3,5\}$. A graph is complete if $\mathcal{E} = \{(i,j)|i,j \in \mathcal{V}, i \neq j\}$, saying any pair of vertices are neighbors. The neighbors of any vertex in a complete graph thus has the property that $\partial(i) = \{1,2,\ldots,i-1,i+1,\ldots,n\}$ $\forall i \in \mathcal{V}$. According to the definition of being complete, the graph in Figure 1 is not complete since some nodes (e.g. 3 and 5) are not neighbors. The set of vertices in a complete subgraph of $\mathcal{G}$ is called a clique. A maximal clique cl is a clique that it is not a subset of another clique, and we use $\mathcal{CL}$ to denote the set of all maximal cliques of the graph $\mathcal{G}$. For instance, the subgraph $\{1,2,5\}$ is a maximal clique in the graph in Figure 1.

Given an undirected graph $\mathcal{G} = \{\mathcal{V},\mathcal{E}\}$ and a vector of parameters θ, a vector of discrete random variables $x = (x_1,x_2,\ldots,x_n) \in \mathcal{X}$ with a joint probability distribution $\pi(x|\theta)$ is said to be an MRF with respect to $\mathcal{G}$ if it fulfills the positivity condition $\pi(x|\theta) > 0$ for all $x \in \mathcal{X}$ and the local Markov property

$$\pi(x_i|x_{-i},\theta) = \pi\left(x_i|x_{\partial(i)},\theta\right) \quad \text{for } i \in \{1,2,\ldots,n\}. \tag{3}$$

The Hammersley-Clifford theorem given by Hammersley and Clifford (1971) states that a discrete random vector x having a joint probability distribution $\pi(x|\theta)$ is an MRF with respect to a graph $\mathcal{G}$ if and only if it can be factorized over the maximal cliques of $\mathcal{G}$, so it can be expressed as

$$\pi(x|\theta) = \frac{\nu(x|\theta)}{C(\theta)}, \text{ where } \nu(x|\theta) = \exp\left\{-\textstyle\sum_{cl\in\mathcal{CL}} \Phi_{cl}(x_{cl};\theta)\right\}$$

is an unnormalized version of $\pi(x|\theta)$, $C(\theta)$ is the normalizing constant and $\Phi_{cl}(x_{cl};\theta)$ is a clique potential function for a maximal clique cl. In a word, this theorem is equivalent to a sufficient and necessary condition under which a positive probability distribution can be represented by a Markov random field.

Defining the energy function $U(x;\theta)$ by $U(x;\theta) = \sum_{cl\in\mathcal{CL}} \Phi_{cl}(x_{cl};\theta)$, the normalizing constant $C(\theta)$ can be written as

$$C(\theta) = \sum_{z\in\mathcal{X}} \exp\{-U(z;\theta)\}.$$

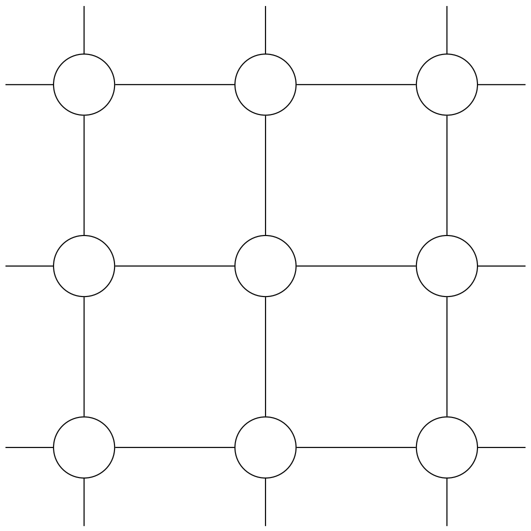


Figure 2: A small section of a rectangular lattice where circles and lines represent vertices and edges in $\mathcal{G}$, respectively.

Note that the normalizing constant $C(\theta)$ is typically not available in closed form. Moreover, the value of $C(\theta)$ cannot be practically computed. For instance, if each component of x has two possible values, 0 and 1, then $C(\theta)$ is calculated based on totally 2^n different combinations of x, so we confirm that computing the accurate value of $C(\theta)$ is greatly computer-intensive even if the number of vertices n is not large (e.g. for a lattice with $n = 100$, $2^{100} > 10^{30}$). As an example of MRFs, we next consider the Ising model.

The Ising model (Ising, 1925) is a particular MRF defining a distribution over a random vector of binary variables. The Ising model and its generalization Potts model (Ashkin and Teller, 1943; Potts, 1952) where random variables might have more than two states are frequently used in the analysis of spatial statistics. Suppose that $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ is defined from a rectangular lattice. Letting n denote the number of vertices in the lattice, we number the vertices from 1 to n lexicographically. Let x_i be the discrete variable related to vertex i, and write $x = (x_1, \ldots, x_n)$. A small part of the resulting graph is shown in Figure 2. Assume that the neighborhood of a vertex is only its nearest horizontal and vertical vertices. Then a maximal clique cl is a set of any pair of two vertices that are horizontally or vertically closest to each other. For the Ising model x is required to be a binary random vector, i.e.

$x \in \{0,1\}^n$, and the potential function for a maximal clique $cl = \{i, j\}$ is given by

$$\Phi_{cl}(x_{cl}; \theta) = -\theta \cdot I(x_i = x_j),$$

where θ determines the intensity of interaction between x_i and its neighbors, and $I(\cdot)$ is an indicator function, i.e. $I(A) = 1$ if A is true and $I(A) = 0$ otherwise. Letting $i \sim j$ denote that vertices i and j are neighbors as defined by $\mathcal{G}$, the energy function of the Ising model can be written as

$$U(x; \theta) = \sum_{cl \in \mathcal{CL}} \Phi_{cl}(x_i, x_j; \theta) = -\theta \sum_{i \sim j} I(x_i = x_j).$$

Thereby, the distribution function of the Ising model is

$$\pi(x|\theta) = \frac{1}{C(\theta)} \exp\left\{\theta \sum_{i \sim j} I(x_i = x_j)\right\}. \tag{4}$$

Now that the distribution of the Ising model (4) is factorized over the cliques of $\mathcal{G}$, we also can easily obtain an expression for the full conditional in (3)

$$\pi(x_i|x_{-i}, \theta) = \pi(x_i|x_{\partial(i)}, \theta) = \frac{1}{C_i\left(x_{\partial(i)}, \theta\right)} \exp\left\{\theta \sum_{j \in \partial(i)} I(x_j = x_i)\right\},$$

where $C_i\left(x_{\partial(i)}, \theta\right)$ is the normalizing constant of $\pi(x_i|x_{\partial(i)}, \theta)$. In addition, the constraint $\Pr(x_i = 1|x_{\partial(i)}, \theta) + \Pr(x_i = 0|x_{\partial(i)}, \theta) = 1$ implies that the normalizing constant is given by

$$C_i\left(x_{\partial(i)}, \theta\right) = \exp\left\{\theta \sum_{j \in \partial(i)} I(x_j = 1)\right\} + \exp\left\{\theta \sum_{j \in \partial(i)} I(x_j = 0)\right\}.$$

2.3 Perfect simulation

To cope with some intrinsic deficiencies of MCMC, we introduce perfect simulation and how to sample by it in this section. In the following we limit our attention to "Coupling From The Past (CFTP)" which is a specific approach in the field of perfect simulation. CFTP is not a generic roadmap for general MRFs, only a few cases including the autologistic model (Besag, 1974) can be perfectly sampled, and simulating from continuous distributions using CFTP is more arduous and such samples are beyond our need. See Propp and Wilson (1996) for more details.

As we know, MCMC requires infinite number of iterations to produce the target distribution, that is, from initial time $t = 0$, a perfect sample can be generated from the target distribution by MCMC at time $t = \infty$. Alternatively, if the Markov chain starts at time $t = -\infty$, then it must converge to its stationary states at time $t = 0$. However, it is impossible to simulate over a infinitely long period of time, and actually we only care about the stationary states at time $t = 0$ instead of the process in which a Markov chain runs from $t = -\infty$ to $t = 0$.

Define a Markov chain $\{x_t\}_{t=-T}^{0}$ from time $t = -T < 0$ to $t = 0$, where $x_t \in \mathcal{X}$ and x_t is n-dimensional for all $t \in \{-T, \ldots, 0\}$. Assume that $\pi(x)$ is the limiting distribution of the Markov chain $\{x_t\}_{t=-T}^{0}$, and that the state space is equipped with a natural partial ordering property (Cressie and Davidson, 1998) "$\preccurlyeq$". Each iteration of the MH scheme uses a number of random variables to generate the potential new state, and decide if it should be accepted. Let v_t denote the vector composed of these random variables at time t. The process in which the state x_{t+1} is simulated based on its last state x_t then can be expressed by a deterministic function, denoted by $\phi(\cdot, \cdot)$, i.e. $x_{t+1} = \phi(x_t, v_t)$. In the following, we also assume that $\phi(\cdot, \cdot)$ owns the property that $\mathcal{X} \ni x \preccurlyeq x' \in \mathcal{X}$ implies $\phi(x, v_0) \preccurlyeq \phi(x', v_0)$ almost surely. The stationary states at time $t = 0$ a Markov chain converges to are independent of its initial state at time $t = -\infty$, so if Markov chains, which start from all states in $\mathcal{X}$ at time $t = -T < 0$ and are updated through $\phi(\cdot, \cdot)$ with the same sequence of random vectors $\{v_t\}_{t=-T}^{0}$ in each iteration, coalesce at time $t \leq 0$, then the state at $t = 0$ for a Markov chain simulated from $-\infty$ is also equal to the same state and thus is a perfect sample.

Furthermore, we assume that there exist an upper bound state $\hat{\mathsf{1}}$ and a lower bound state $\hat{\mathsf{0}}$ such that $\hat{\mathsf{0}} \preccurlyeq x \preccurlyeq \hat{\mathsf{1}}$ for all $x \in \mathcal{X}$. Define two Markov chains $\{\hat{\mathsf{1}}_t\}_{t=-T}^{1}$ and $\{\hat{\mathsf{0}}_t\}_{t=-T}^{0}$ that start from $\hat{\mathsf{1}}$ and $\hat{\mathsf{0}}$, respectively, at time $t = -T$, i.e. $\hat{\mathsf{1}}_{-T} = \hat{\mathsf{1}}$ and $\hat{\mathsf{0}}_{-T} = \hat{\mathsf{0}}$, and are updated according to $\phi(\cdot, \cdot)$ with the same sequence of random vectors $\{v_t\}_{t=-T}^{0}$. Hence, for any $x_t \in \mathcal{X}$ at any time $t \in \{-T, \ldots, 0\}$, given $\hat{\mathsf{0}}_{-T} \preccurlyeq x_{-T} \preccurlyeq \hat{\mathsf{1}}_{-T}$ it is easy to obtain $\phi(\hat{\mathsf{0}}_{-T}, v_{-T}) \preccurlyeq \phi(x_{-T}, v_{-T}) \preccurlyeq \phi(\hat{\mathsf{1}}_{-T}, v_{-T})$, i.e. $\hat{\mathsf{0}}_{-T+1} \preccurlyeq x_{-T+1} \preccurlyeq \hat{\mathsf{1}}_{-T+1}$, then iteratively $\hat{\mathsf{0}}_{-T+2} \preccurlyeq x_{-T+2} \preccurlyeq \hat{\mathsf{1}}_{-T+2}$ and so on until $\hat{\mathsf{0}}_0 = x_0 = \hat{\mathsf{1}}_0$. Therefore, if $\{\hat{\mathsf{1}}_t\}_{t=-T}^{0}$ and $\{\hat{\mathsf{0}}_t\}_{t=-T}^{0}$ coalesce at time $t \leq 0$, all chains starting between $\hat{\mathsf{1}}$ and $\hat{\mathsf{0}}$ must also coalesce with $\hat{\mathsf{1}}_t$ and $\hat{\mathsf{0}}_t$ before time 0. Thereby, we are able to obtain a perfect sample from the target distribution

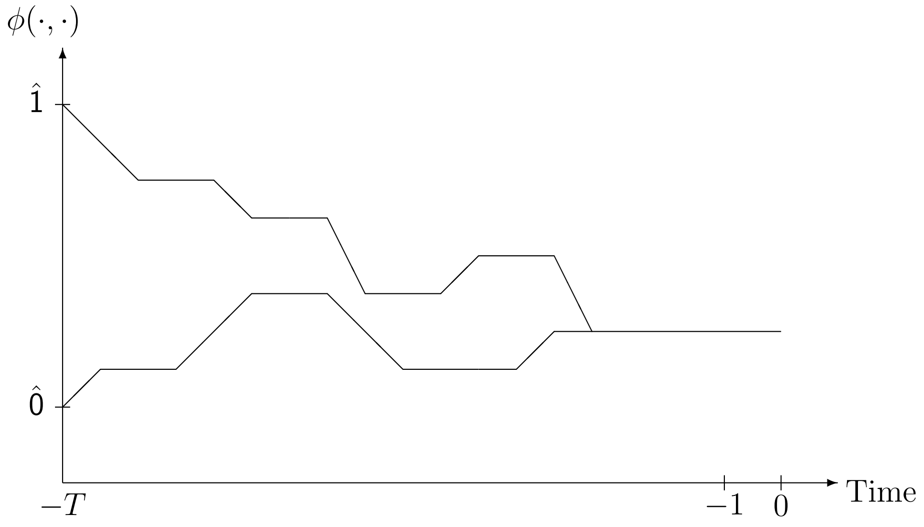


Figure 3: The coalescence of coupled chains

by simulating merely two Markov chains $\{\hat{1}_t\}_{t=-T}^{0}$ and $\{\hat{0}_t\}_{t=-T}^{0}$. This is the essential idea of CFTP. Moreover, CFTP can also determine the smallest value of T, denoted by T_*, at which $\{\hat{1}_t\}_{t=-T}^{0}$ and $\{\hat{0}_t\}_{t=-T}^{0}$ coalesce before time 0. If for some value T, $\{\hat{1}_t\}_{t=-T}^{0}$ and $\{\hat{0}_t\}_{t=-T}^{0}$ have not coalesced at time 0, we can doubly increase the value of T, i.e. set $T = 2T$, and if time $2T$ is a coupling time, it is not difficult to prove that $\Pr\{T < T_* \leq 2T\} = 1$. Figure 3 roughly illustrates how CFTP works for the exact sampling.

Let us consider how to sample exactly from a generalized version of the Ising model (4), namely the autologistic model (Besag, 1974). Assume a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is defined from a rectangular lattice via the same construction procedure as in Section 2.2. Therefore, any notion or term relating to $\mathcal{G}$ is the same as that in Section 2.2. In this section, we introduce the autologistic model by first giving the joint probability distribution of x given a parameter vector $\theta = (\alpha_1, \ldots, \alpha_n, \beta)$

$$\pi(x|\theta) = \frac{1}{C(\theta)} \exp\left\{\sum_{i=1}^{n} \alpha_i x_i + \sum_{i\sim j} \beta \cdot I(x_i = x_j)\right\}, \tag{5}$$

where $C(\theta)$ is the normalizing constant, $x_i \in \{0, 1\}$ and $i, j \in \{1, 2, \ldots, n\}$. This distribution function can be rewritten as

$$\pi(x|\theta) = \frac{1}{C(\theta)} \exp\left\{\sum_{i\sim j} \left[\frac{\alpha_i x_i}{|\partial(i)|} + \frac{\alpha_j x_j}{|\partial(j)|} + \beta \cdot I(x_i = x_j)\right]\right\},$$

where $|\partial(i)|$ represents the number of neighbors of vertex i. Thus, for a maximal clique $cl = \{i, j\} \in \mathcal{CL}$, the potential function is

$$\Phi_{cl}\left(x_{cl};\theta\right) = -\left(\frac{\alpha_i x_i}{|\partial(i)|} + \frac{\alpha_j x_j}{|\partial(j)|} + \beta \cdot I(x_i = x_j)\right).$$

Since $\pi(x|\theta)$ can be factorized over the maximal cliques of $\mathcal{G}$, the random vector x is an MRF with respect to $\mathcal{G}$ by the Hammersley-Clifford theorem. Thereby, the full conditional of (5) is

$$\pi(x_i|x_{-i},\theta) = \frac{1}{C_i(x_{-i},\theta)} \exp\left\{\alpha_i x_i + \sum_{j\in\partial(i)} \beta \cdot I(x_i = x_j)\right\}, \tag{6}$$

where $C_i(x_{-i},\theta)$ is the normalizing constant of $\pi(x_i|x_{-i},\theta)$. Based on the fact $\Pr(x_i = 1|x_{-i},\theta) + \Pr(x_i = 0|x_{-i},\theta) = 1$, the normalizing constant $C_i(x_{-i},\theta)$ in the full conditional is known, that is,

$$C_i(x_{-i},\theta) = \exp\left\{\alpha_i \cdot 1 + \beta \sum_{j\in\partial(i)} I(x_j = 1)\right\} + \exp\left\{\alpha_i \cdot 0 + \beta \sum_{j\in\partial(i)} I(x_j = 0)\right\}.$$

Inserting the expression of $C_i(x_{-i},\theta)$ into (6) and rearranging it, we obtain

$$\Pr(x_i = 1|x_{-i},\theta) = \frac{1}{1 + \exp\left\{\beta \sum_{j\in\partial(i)} \left[I(x_j = 0) - I(x_j = 1)\right] - \alpha_i\right\}}, \tag{7}$$

and $\Pr(x_i = 0|x_{-i},\theta) = 1 - \Pr(x_i = 1|x_{-i},\theta)$.

In addition to the settings for the autologistic model above, we define the partial ordering "$\preccurlyeq$". Assume $x = (x_1 \ldots, x_n) \in \mathcal{X}$ and $x' = (x'_1, \ldots, x'_n) \in \mathcal{X}$, the partially ordering $x \preccurlyeq x'$ holds if and only if $x_i \leq x'_i$ for all $i \in \{1, \ldots, n\}$. Thus, the upper bound state is $\hat{1} = (1, 1, \ldots, 1)$ with n ones, and the lower bound state is $\hat{0} = (0, 0, \ldots, 0)$ with n zeros. Suppose for $i \in \{1, 2, \ldots, n\}$ at time t, the ith component of $\hat{1}_t$ is $\hat{1}_t^i$, and v_t is made up of two random numbers (u_t, k_t) sampled from a real uniform distribution $u_t \sim \text{Unif}(0,1)$ and an integer uniform distribution $k_t \sim \text{Unif}[1,n]$, respectively. Then use the Gibbs update only for the component $\hat{1}_t^{k_t}$ at time t via the full conditional probability of the form (7), i.e.

$$\hat{1}_{t+1}^{k_t} = I\left(u_t \leq \pi\left(\hat{1}_t^{k_t} = 1 \middle| \partial(k_t)\right)\right),$$

where $I(\cdot)$ is an indicator function, and for any $i \neq k_t$, $\hat{1}_{t+1}^i = \hat{1}_t^i$. As a consequence, this procedure can be described by $\hat{1}_{t+1} = \phi(\hat{1}_t, v_t)$. In the same way, we have $\hat{0}_{t+1} = \phi(\hat{0}_t, v_t)$.

```
• $T = 1$
• repeat
    – upper = $(1, 1, \ldots, 1)$, lower = $(0, 0, \ldots, 0)$
    – for $t = -T$ to 0
        * generate an integer $k_t \sim \text{Unif}[1, n]$ and a real number $u_t \sim \text{Unif}(0, 1)$, and set $v_t = (u_t, k_t)$
        * upper = $\phi(\text{upper}, v_t)$ and lower = $\phi(\text{lower}, v_t)$
    – $T = 2T$
• until upper = lower
• return upper
```

Figure 4: Algorithm of CFTP for generating the exact samples from the autologistic model of the form (5).

Figure 4 demonstrates the main principles about exactly simulating from the autologistic model (5).

Figure 5 shows six simulations from the autologstic model (5) with different sets of parameters on a 100×100 lattice. The three scenes on the left in this figure are generated with $\alpha_i = 0.05$ for all i, and the three scenes on the right are based on $\alpha_i = 0.3$ for all i. The parameter β is set to 0.1, 0.5 and 0.9 for the two upper, middle and lower scenes, respectively. Black and white pixels stand for numbers 1 and 0, respectively. By vertically observing subfigures from top to bottom on the left-hand (or right-hand) panel, the distribution of black and white pixels is less and less scattering, and it shows a significant clustering pattern in the bottom plots. Thus, β determines the interaction effect between x_i and its neighbors. Comparing two plots horizontally in the upper, middle and lower positions, the number of black pixels on the right-hand panel is greater than that on the left-hand panel, so we know that it is likely to get more 1's as α_i increases.

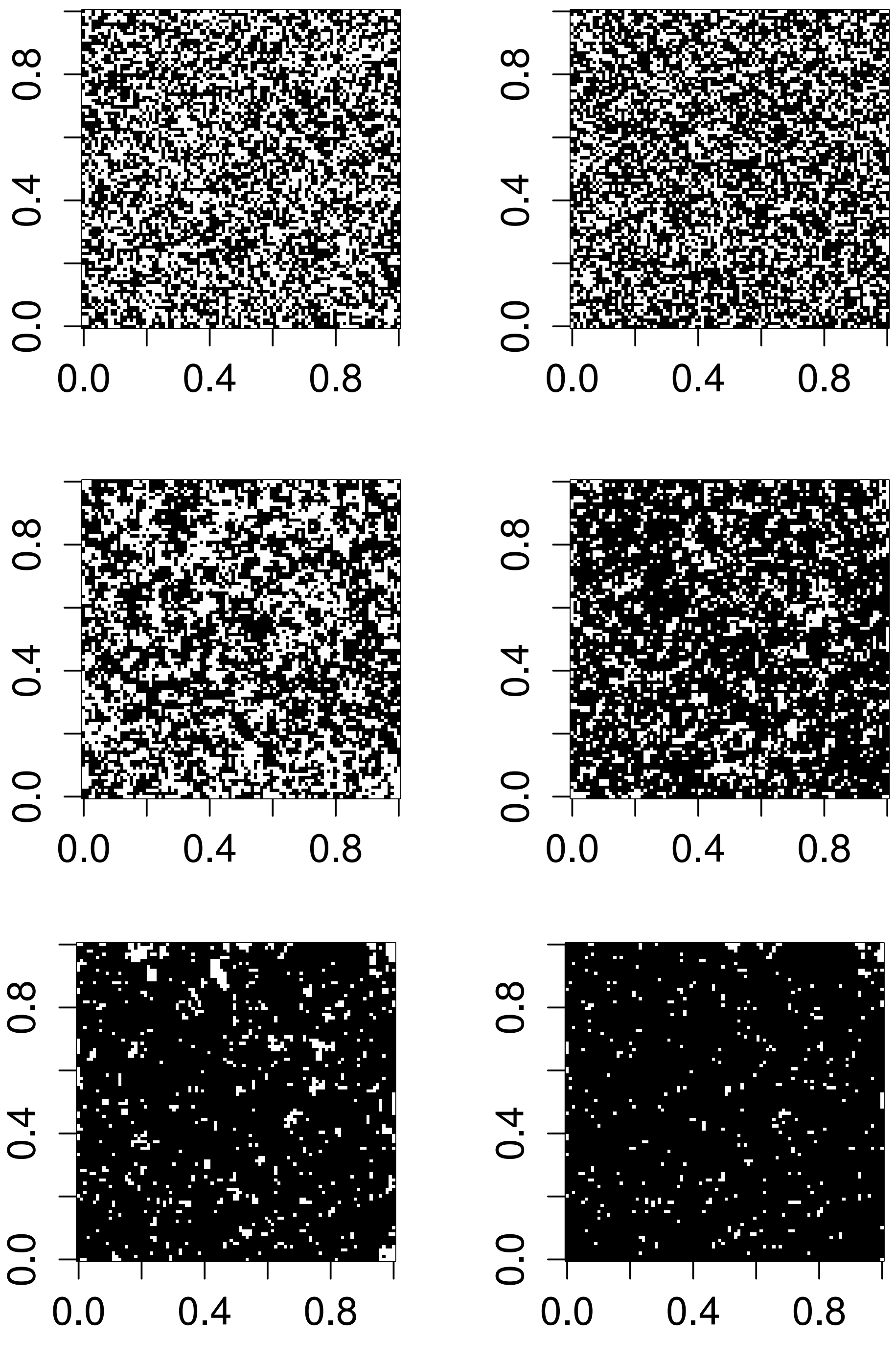
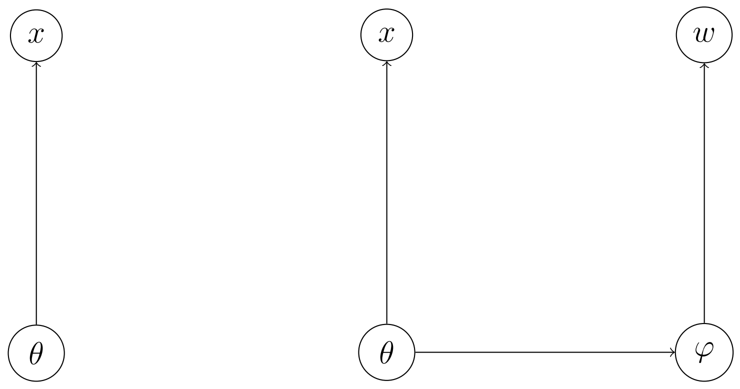


Figure 5: Simulations of the autologistic model of the form (5) with different α_i ($i = 1, 2, \ldots, 100$) and β on a 100×100 lattice. The three scenes on the left in this figure are generated with $\alpha_i = 0.05$ for all i, and the three scenes on the right are based on $\alpha_i = 0.3$ for all i. The parameter β is set to 0.1, 0.5 and 0.9 for the two upper, middle and lower scenes, respectively.



(a) The original model (b) The extended model

Figure 6: Two graphical models: (a) is the original model of $\pi(x|\theta)$ without auxiliary variables, (b) is the extended model used for the exchange algorithm with auxiliary variables φ and w.

2.4 The exchange algorithm

The exchange algorithm helps us get rid of the computational difficulties and other potential issues as the result of normalizing constants when using naïve MCMC schemes, so in this section we briefly discuss this algorithm and it is the core strategy we adopt in the following parts. One can read Murray et al. (2006) for more details.

Suppose we have observed $x \in \mathcal{X}$ assumed to be a sample from an MRF with distribution $\pi(x|\theta)$, where θ is a parameter vector. Our interest is in θ and adopting the Bayesian approach, we assume a prior $\pi(\theta)$ for it, and focus on the resulting posterior $\pi(\theta|x) \propto \pi(\theta)\pi(x|\theta)$. The corresponding graphical model is illustrated in Figure 6(a). Adopting the same notations as in Section 2.2, the posterior distribution of θ can also be expressed as

$$\pi(\theta|x) \propto \frac{1}{C(\theta)}\pi(\theta)\nu(x|\theta). \tag{8}$$

Apparently, simulating from $\pi(\theta|x)$ by MCMC leads to a fraction of two different normalizing constants that is tough to compute in practice. Hence, we choose to use the exchange algorithm (Murray et al., 2006). On the basis of the original graphical model shown in Figure 6(a), we need to construct two auxiliary variables w and φ, and the new graphical model is depicted in Figure 6(b). Given θ we assume it is available to use CFTP to sample x from the distribution function $\pi(x|\theta)$ whose normalizing constant is a function of θ, and the distribution $\pi(w|\varphi)$ is of the same type of distribution as $\pi(x|\theta)$. It is required that $w \in \mathcal{X}$

and $\varphi \in \Theta$ share the same state spaces as $x \in \mathcal{X}$ and $\theta \in \Theta$, respectively. Moreover, let $q(\varphi|\theta)$ be a conditional distribution for φ given θ.

Since only x is observed, we first focus on the joint distribution $\pi(\theta, \varphi, w|x)$, and then by integrating out auxiliary variables φ and w, it is easy to obtain

$$\pi(\theta|x) = \int_{\varphi \in \Theta} \left[\sum_{w \in \mathcal{X}} \pi(\theta, \varphi, w|x) \right] d\varphi.$$

Therefore, the joint distribution $\pi(\theta, \varphi, w|x)$ is of interest, and

$$\begin{aligned} \pi(\theta, \varphi, w|x) &\propto \pi(\theta, \varphi, w, x) \\ &= \pi(\theta) q(\varphi|\theta) \pi(x|\theta) \pi(w|\varphi) \\ &= \frac{1}{C(\theta)C(\varphi)} \pi(\theta) q(\varphi|\theta) \nu(x|\theta) \nu(w|\varphi). \end{aligned}$$

We simulate from the distribution $\pi(\theta, \varphi, w|x)$ by an MH scheme which consists of two steps. The first is to update (φ, w) by the block Gibbs step

$$\begin{aligned} (\varphi, w) &\sim \pi(\varphi, w|\theta, x) \\ &= q(\varphi|\theta) \pi(w|\varphi). \end{aligned}$$

The second is to update (θ, φ) by a block MH update. The potential new state is defined by interchanging θ and φ, i.e. $(\theta', \varphi') = (\varphi, \theta)$. The acceptance probability is

$$\begin{aligned} \alpha(\theta', \varphi'|\theta, \varphi) &= \alpha(\varphi, \theta|\theta, \varphi) \\ &= \frac{\pi(\varphi, \theta, w|x)}{\pi(\theta, \varphi, w|x)} \\ &= \frac{\pi(\varphi) q(\theta|\varphi) \nu(x|\varphi) \nu(w|\theta)}{\pi(\theta) q(\varphi|\theta) \nu(x|\theta) \nu(w|\varphi)}, \end{aligned} \tag{9}$$

which can be efficiently computed since all terms are explicitly expressed. The pseudocode concerning how to sample from $\pi(\theta|x)$ is illustrated in Figure 7.

2.5 Partially ordered Markov models

Partially ordered Markov models (POMMs) are a class of spatial models whose members have probability distributions that can be written in closed form. For deeper and broader discussions, read Cressie and Davidson (1998), Tjelmeland and Austad (2012).

```
Input: Initial setting $\theta$, number of iterations $T$

• for $t = 1, 2, \ldots, T$

    – propose $\varphi \sim q(\cdot|\theta)$
    – use CFTP to generate $w$, i.e. $w \sim \pi(\cdot|\varphi)$
    – compute $\alpha(\varphi, \theta|\theta, \varphi) = \frac{\pi(\varphi)q(\theta|\varphi)\nu(x|\varphi)\nu(w|\theta)}{\pi(\theta)q(\varphi|\theta)\nu(x|\theta)\nu(w|\varphi)}$
    – draw $u \sim$Unif$[0, 1]$
    – set $\theta = \varphi$ if $u < \alpha(\varphi, \theta|\theta, \varphi)$

• end for
```

Figure 7: The exchange algorithm for generating the vector of parameters of interest θ from the posterior distribution $\pi(\theta|x)$.

Here we only describe the computational algorithm of POMMs without the rigorous proof as it is the programming techniques regarding POMMs that are used in the later sections. Let $x = (x_1, x_2, \ldots, x_n) \in \{0, 1\}^n$ be a binary MRF with respect to a graph, and assume that $\pi(x) = \frac{1}{C}\exp\{-U(x)\}$ is the corresponding joint distribution. Using the multiplication rule of probability successively, we can in principle obtain

$$\pi(x) = \pi(x_1|x_{-1})\pi(x_{-1}),$$

$$\pi(x_{-1}) = \pi\left(x_2|x_{-\{1,2\}}\right)\pi\left(x_{-\{1,2\}}\right)$$

and so on. Eventually, this recursion gives

$$\pi(x) = \pi(x_n)\prod_{i=1}^{n-1}\pi(x_i|x_{i+1}, \ldots, x_{n-1}). \tag{10}$$

Equation (10) is one possible form that is convenient to apply, but not the general definition form. The normalizing constant C of $\pi(x)$ is actually included in $\pi(x_n)$, so C can be evaluated from the fact $\pi(x_n = 1) + \pi(x_n = 0) = 1$. Meanwhile, simulating x from $\pi(x)$ is tantamount to firstly simulating x_n from $\pi(x_n)$, secondly simulating x_{n-1} from $\pi(x_{n-1}|x_n)$ and so on.

From Equation (10) we essentially know the rules concerning how to generate x from $\pi(x)$, but such an analytical exact algorithm based on these rules is not computationally feasible when the number of neighbors becomes large as it needs a lot of time consumptions to compute and potentially requires much memory storage. Thereby, Tjelmeland and Austad (2012) proposed an approximation method to overcome these two difficulties. The core idea is to substitute the conditional distributions with approximations by ignoring higher-order interaction effects if some threshold conditions hold. More specifically, the joint distribution $\pi(x)$ is decomposed into the exact full conditional $\pi(x_1|x_{-1})$ and an approximation $\widetilde{\pi}(x_{-1})$ of $\pi(x_{-1})$. Then decompose the approximation $\widetilde{\pi}(x_{-1})$ into an approximation $\widetilde{\pi}\left(x_2|x_{-\{1,2\}}\right)$ of $\pi\left(x_2|x_{-\{1,2\}}\right)$ and an approximation $\widetilde{\pi}\left(x_{-\{1,2\}}\right)$ of $\pi\left(x_{-\{1,2\}}\right)$ and so on. Recursively, the form of (10) turns out to be

$$\pi(x) \approx \widetilde{\pi}(x) = \pi(x_1|x_{-1})\widetilde{\pi}(x_n)\prod_{i=2}^{n-1}\widetilde{\pi}(x_i|x_{i+1},\dots,x_{n-1}).$$

One should also note that the normalizing constant of $\pi(x)$ can be approximated by that of $\widetilde{\pi}(x)$, and approximate samples from $\pi(x)$ can be generated by sampling from $\widetilde{\pi}(x)$.

3 Methodology and algorithm

As aforementioned, our target is to simulate from the joint distribution $\pi(\theta, x|y)$ provided noisy observations $y = (y_1, y_2, \dots, y_n)$. The model components include a prior $\pi(\theta)$ for $\theta \in \Theta$, the distribution of a discrete MRF $\pi(x|\theta)$ for an unobserved field $x = (x_1, x_2, \dots, x_n) \in \mathcal{X}$ and a likelihood function $\pi(y|x)$ for the observed data y. Assuming y to be conditionally independent of θ given x, we can visualize the model as done in Figure 8. Specifically, given a vector of parameters $\theta = (\theta_0, \theta_1)$, the distribution $\pi(x|\theta)$ is a simplified version of the autologistic model (5) by setting $\alpha_i = \theta_0,\ i = 1, 2, \dots, n$ and $\beta = \theta_1$ and thereby $\pi(x|\theta)$ is expressed as

$$\pi(x|\theta) = \frac{1}{C(\theta)}\exp\left\{\theta_0\sum_{i=1}^{n}x_i + \theta_1\sum_{i\sim j}I(x_i = x_j)\right\}. \tag{11}$$

The likelihood for the observed data y is assumed to be

$$\pi(y|x,\theta) = \pi(y|x) = \prod_{i=1}^{n}\pi(y_i|x_i) = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left\{-\frac{\sum_{i=1}^{n}(y_i - x_i)^2}{2\sigma^2}\right\},$$

where σ^2 is a common known variance for each Gaussian distribution $\pi(y_i|x_i)$.

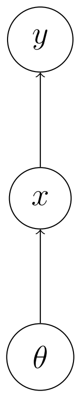


Figure 8: The graphical model that indicates relationships among hidden variables x, observations y and parameters θ.

Thus, according to the graphical model depicted in Figure 8, the posterior joint distribution $\pi(\theta, x|y)$ of interest can be derived by Bayes' theorem and conditional independence as

$$\pi(\theta, x|y) \propto \pi(x, y, \theta) = \pi(\theta)\pi(x|\theta)\pi(y|x, \theta) = \pi(\theta)\pi(x|\theta)\pi(y|x).$$

However, the existence of $C(\theta)$ in $\pi(x|\theta)$ leads to unsolvable summations when computing the acceptance ratio using naïve MCMC schemes. As a result, we should expand the statistical relationship amongst θ, x and y so that the normalizing constants can disappear under the conditions of the MH technique. The exchange algorithm invented by Murray et al. (2006) is a useful key to such problems, but it is not valid for latent variables, so we put forward two different strategies that can be regarded as extended versions of the exchange algorithm.

3.1 Simulation strategy I

In this strategy, we need auxiliary variables $\varphi \in \Theta$ and $w \in \mathcal{X}$, and the corresponding graphical model is illustrated in Figure 9. Besides the assumptions and models in the last section, we additionally let $q(\varphi|\theta)$ denote the conditional distribution for φ given θ, and the distribution function for w given φ is in the same form as the distribution for x given θ, i.e.

$$\pi(w|\varphi) = \frac{1}{C(\varphi)} \exp\left\{\varphi_0 \sum_{i=1}^{n} w_i + \varphi_1 \sum_{i\sim j} I(w_i = w_j)\right\},$$

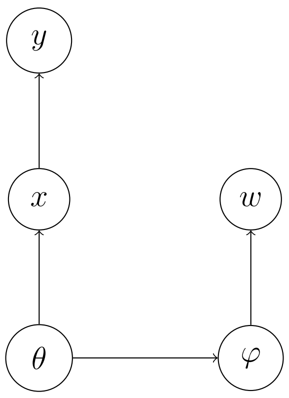


Figure 9: The graphical model that uses an extended version of the exchange algorithm with an auxiliary variable w to simulate θ and x from $\pi(\theta, x|y)$.

where $\varphi = (\varphi_0, \varphi_1)$. Similarly with Section 2.4, the distribution $\pi(\theta, x|y)$ can be derived from $\pi(\theta, \varphi, w, x|y)$ by marginalization, i.e.

$$\pi(\theta, x|y) = \int_{\varphi\in\Theta} \left[\sum_{w\in\mathcal{X}} \pi(\theta, \varphi, w, x|y) \right] d\varphi.$$

Thereby, the joint distribution $\pi(\theta, \varphi, w, x|y)$ becomes what we are interested in, and it can be written as

$$\begin{aligned}\pi(\theta, \varphi, x, w|y) &\propto \pi(\theta, \varphi, x, w, y) \\ &= \pi(\theta)q(\varphi|\theta)\pi(x|\theta)\pi(w|\varphi)\pi(y|x).\end{aligned}$$

For this simulation strategy, we at first use the block Gibbs update for (φ, w), that is,

$$(\varphi, w) \sim \pi(\varphi, w|\theta, x, y) = q(\varphi|\theta)\pi(w|\varphi),$$

meaning to propose φ from $q(\varphi|\theta)$ and thereafter to generate w from $\pi(w|\varphi)$ by perfect sampling. Afterwards, we use the exchange technique for (θ, φ) by setting $(\theta', \varphi') = (\varphi, \theta)$ with acceptance probability

$$\begin{aligned}\alpha(\theta', \varphi'|\theta, \varphi) &= \alpha(\varphi, \theta|\theta, \varphi) \\ &= \frac{\pi(\varphi)q(\theta|\varphi)\nu(x|\varphi)\nu(w|\theta)\pi(y|x)}{\pi(\theta)q(\varphi|\theta)\nu(x|\theta)\nu(w|\varphi)\pi(y|x)} \\ &= \frac{\pi(\varphi)q(\theta|\varphi)\nu(x|\varphi)\nu(w|\theta)}{\pi(\theta)q(\varphi|\theta)\nu(x|\theta)\nu(w|\varphi)}.\end{aligned} \tag{12}$$

We have already analyzed how to update θ, φ and w, so the remainder is to figure out an update for x. By observing Figure 9, we realize that x is not only related to θ, but depends on y as well, so the distribution of x conditioned on θ and y is

$$\pi(x|\theta, y) \propto \pi(x, \theta, y) \propto \pi(x|\theta)\pi(y|x).$$

The normalizing constant of $\pi(y|x)$ is assumed to be independent of θ, so the unnormalized function of $\pi(x|\theta, y)$ is the product of the unnormalized function of $\pi(x|\theta)$ and the unnormalized function of $\pi(y|x)$, i.e. the joint probability function of $\pi(x|\theta, y)$ is

$$\begin{aligned} \pi(x|\theta, y) =& \frac{1}{C(\theta, y)} \exp\left\{ \theta_0 \sum_{i=1}^{n} x_i + \theta_1 \sum_{i\sim j} I(x_i = x_j) - \frac{\sum_{i=1}^{n}(y_i - x_i)^2}{2\sigma^2} \right\} \\ \propto& \exp\left\{ \sum_{i=1}^{n} \left(\theta_0 + \frac{2y_i - 1}{2\sigma^2} \right) x_i + \theta_1 \sum_{i\sim j} I(x_i = x_j) \right\}. \end{aligned} \tag{13}$$

Furthermore, we notice that (13) is the distribution function of an autologistic model with $\alpha_i = \theta_0 + \frac{2y_i - 1}{2\sigma^2}$ and $\beta = \theta_1$. Thus, it is available to use CFTP to sample from $\pi(x|\theta, y)$. Hence, we can use the Gibbs update for x, i.e. $x \sim \pi(x|\theta, \varphi, w, y) = \pi(x|\theta, y)$. In order to make this simulation strategy easier to understand, one can see the pseudocode shown in Figure 10.

An alternative way to formulate this algorithm is as follows. We discard φ and w in Figure 9, and instead consider φ as a proposal of θ and w as a proposal of x. Then the joint distribution $\pi(\theta, x|y)$ is of interest, and a proposal distribution is

$$q(\varphi, w|\theta, x, y) = q(\varphi|\theta)\pi(w|\varphi)$$

is used. The corresponding acceptance probability becomes

$$\begin{aligned} \alpha(\varphi, w|\theta, x) &= \frac{\pi(\varphi, w|y)q(\theta, x|\varphi, w, y)}{\pi(\theta, x|y)q(\varphi, w|\theta, x, y)} \\ &= \frac{\pi(\varphi)q(\theta|\varphi)\nu(x|\varphi)\nu(w|\theta)}{\pi(\theta)q(\varphi|\theta)\nu(x|\theta)\nu(w|\varphi)}, \end{aligned}$$

which is precisely equal to (12). This offers an alternative to think about the problem.

Studying the realizations in Figure 9, we find out that x is directly generated from $\pi(x|\theta)$ if θ is given, so the dependency between x and θ cannot be ignored. Thus, using the block update that binds x with θ may be a good option, and this ideal is introduced in the next section.

```
Input: Initial setting $\theta$, $x$ and number of iterations $T$

• for $t = 1, 2, \ldots, T$

    – propose $\varphi \sim q(\cdot|\theta)$
    – use CFTP to generate $w$, i.e. $w \sim \pi(\cdot|\varphi)$
    – compute $\alpha(\theta', \varphi'|\theta, \varphi) = \frac{\pi(\varphi)q(\theta|\varphi)\nu(x|\varphi)\nu(w|\theta)}{\pi(\theta)q(\varphi|\theta)\nu(x|\theta)\nu(w|\varphi)}$
    – draw $u \sim$Unif$[0, 1]$
    – set $\theta = \varphi$ if $u < \alpha(\theta', \varphi'|\theta, \varphi)$
    – simulate $x$ according to $x \sim \pi(\cdot|\theta, y)$

• end for
```

Figure 10: The extended exchange algorithm for generating the parameters of interest θ and the hidden variables x from the joint posterior distribution $\pi(\theta, x|y)$.

3.2 Simulation strategy II

In this section, the simulation strategy with the blocking technique requires four auxiliary variables $\varphi \in \Theta$, $w \in \mathcal{X}$, $x' \in \mathcal{X}$ and $w' \in \mathcal{X}$. The graphical model for this strategy is depicted in Figure 11. Apart from $\pi(w|\varphi)$ and $q(\varphi|\theta)$ chosen as in Section 3.1, assume that the distribution for w' given θ is the POMM approximation $\widetilde{\pi}(w'|\theta)$ of $\pi(w'|\theta)$, and the distribution for x' given (φ, y) is the POMM approximation $\widetilde{\pi}(x'|\varphi, y)$ of $\pi(x'|\varphi, y)$, where

$$\pi(w'|\theta) = \frac{1}{C(\theta)} \exp\left\{\theta_0 \sum_{i=1}^{n} w'_i + \theta_1 \sum_{i\sim j} I(w'_i = w'_j)\right\},$$

and $\pi(x'|\varphi, y)$ is of the form (13), i.e.

$$\pi(x'|\varphi, y) \propto \exp\left\{\sum_{i=1}^{n}\left(\varphi_0 + \frac{2y_i - 1}{2\sigma^2}\right) x'_i + \varphi_1 \sum_{i\sim j} I(x'_i = x'_j)\right\}.$$

Analogously as before, we can obtain $\pi(\theta, x|y)$ by marginalizing $\pi(x, \theta, w, \varphi, x', w'|y)$, that is,

$$\pi(\theta, x|y) = \int_{\varphi\in\Theta} \left[\sum_{w'\in\mathcal{X}} \sum_{x'\in\mathcal{X}} \sum_{w\in\mathcal{X}} \pi(x, \theta, w, \varphi, x', w'|y)\right] d\varphi.$$

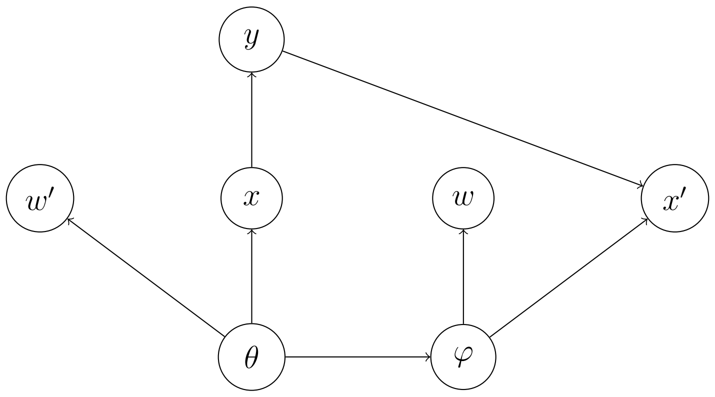


Figure 11: The graphical model that uses an extended version of the exchange algorithm with auxiliary variables φ, w, x' and w' based on blocking to simulate θ and x from $\pi(\theta, x|y)$.

Thus, the joint distribution of interest $\pi(x, \theta, w, \varphi, x', w'|y)$ is

$$\begin{aligned}\pi(x, \theta, w, \varphi, x', w'|y) &\propto \pi(x, \theta, w, \varphi, x', w', y)\\ &= \pi(\theta)\pi(x|\theta)q(\varphi|\theta)\pi(w|\varphi)\pi(y|x)\widetilde{\pi}(w'|\theta)\widetilde{\pi}(x'|\varphi, y)\\ &= \frac{\pi(\theta)\nu(x|\theta)q(\varphi|\theta)\nu(w|\varphi)\nu(y|x)\widetilde{\pi}(w'|\theta)\widetilde{\pi}(x'|\varphi, y)}{C(\theta)C(\varphi)}.\end{aligned} \tag{14}$$

With respect to block updates, initially we simulate (w, φ, x', w') by the block Gibbs update

$$\pi(w, \varphi, x', w'|\theta, x, y) = q(\varphi|\theta)\widetilde{\pi}(w'|\theta)\pi(w|\varphi)\widetilde{\pi}(x'|\varphi, y),$$

meaning first to propose φ from $q(\varphi|\theta)$, then to generate w' from $\widetilde{\pi}(w'|\theta)$ by the POMM approximation and draw w from $\pi(w|\varphi)$ through perfect sampling, and finally to sample x' from $\widetilde{\pi}(x'|\varphi, y)$ using the POMM approximation. Afterwards, suppose the potential new update written as $S' = (x', \varphi, w', \theta, x, w)$ is proposed from the current state $S = (x, \theta, w, \varphi, x', w')$. Specifically, interchange x and x', swap θ and φ, exchange w and w'. Thus, this forms the state shown in Figure 12. The acceptance probability is

$$\begin{aligned}\alpha(S'|S) &= \frac{\pi(x', \varphi, w', \theta, x, w|y)}{\pi(x, \theta, w, \varphi, x', w'|y)}\\ &= \frac{\pi(\varphi)\nu(x'|\varphi)q(\theta|\varphi)\nu(w'|\theta)\nu(y|x')\widetilde{\pi}(w|\varphi)\widetilde{\pi}(x|\theta, y)}{\pi(\theta)\nu(x|\theta)q(\varphi|\theta)\nu(w|\varphi)\nu(y|x)\widetilde{\pi}(w'|\theta)\widetilde{\pi}(x'|\varphi, y)}\\ &= \frac{\pi(\varphi)\nu(x'|\varphi)q(\theta|\varphi)\nu(w'|\theta)\nu(y|x')\widetilde{\pi}(w|\varphi)\widetilde{\pi}(x|\theta, y)}{\pi(\theta)\nu(x|\theta)q(\varphi|\theta)\nu(w|\varphi)\nu(y|x)\widetilde{\pi}(w'|\theta)\widetilde{\pi}(x'|\varphi, y)}.\end{aligned} \tag{15}$$

By the same arguments as in Section 3.1, we also sample x from $\pi(x|\theta, y)$ whose distribution function is of the form (13). Figure 13 shows how to simulate θ and x from $\pi(\theta, x|y)$ through

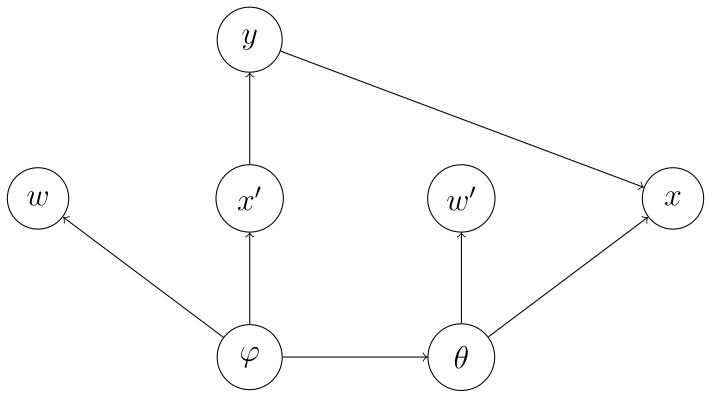


Figure 12: The state after using the exchange proposal with respect to the graphical model in Figure 11.

```
Input: Initial setting $\theta$, $x$ and number of iterations $T$

  • for $t = 1, 2, \ldots, T$

      – propose $\varphi \sim q(\cdot|\theta)$
      – use CFTP to generate $w$, i.e. $w \sim \pi(\cdot|\varphi)$
      – generate $x'$ and $w'$ from the approximate distributions $\widetilde{\pi}(x'|\varphi, y)$
        and $\widetilde{\pi}(w'|\theta)$, respectively
      – compute $\alpha(S'|S)$ of the form (15)
      – draw $u \sim$Unif$[0, 1]$
      – set $\theta = \varphi$, $x = x'$ and $w = w'$ if $u < \alpha(S'|S)$

  • end for
```

Figure 13: The extended exchange algorithm with blocking for generating the parameters of interest θ and the hidden variables x from the joint posterior distribution $\pi(\theta, x|y)$.

the block update.

Now we go back to the issue about the unsolvable fraction of normalizing constants, and discuss a bit more. Obviously, if we had used $\pi(w|\varphi)$ and $\pi(x|\theta, y)$ instead of their corresponding POMM approximations in the model in Figure 11, the formula of acceptance probability (15) would have become

$$\alpha(S'|S) = \frac{C(\theta)C(\varphi, y)}{C(\varphi)C(\theta, y)} \cdot \frac{\pi(\varphi)\nu(x'|\varphi)q(\theta|\varphi)\nu(y|x')\nu(x|\theta, y)}{\pi(\theta)\nu(x|\theta)q(\varphi|\theta)\nu(y|x)\nu(x'|\varphi, y)},$$

where the fraction $\frac{C(\theta)C(\varphi,y)}{C(\varphi)C(\theta,y)}$ is not computationally tractable in practice, and thereby this is why we adopt the POMM approximations rather than the corresponding analytical distributions.

Corresponding to what we have explained for simulation strategy I in Section 3.1, it is again possible to consider the simulation algorithm as simulating from a model with less auxiliary variables. If we discard x' and w' in Figure 11, and instead regard x' as a proposal of x and w' as a proposal of w, then the proposal distribution is

$$q(x', w'|x, w, \theta, \varphi, y) = \widetilde{\pi}(w'|\theta)\widetilde{\pi}(x'|\varphi, y),$$

so the corresponding acceptance probability becomes

$$\begin{aligned}\alpha(x', \varphi, w', \theta|x, \theta, w, \varphi) &= \frac{\pi(x', \varphi, w', \theta|y)q(x, w|x', w', \varphi, \theta, y)}{\pi(x, \theta, w, \varphi|y)q(x', w'|x, w, \theta, \varphi, y)} \\ &= \frac{\pi(\varphi)\nu(x'|\varphi)q(\theta|\varphi)\nu(w'|\theta)\nu(y|x')\widetilde{\pi}(w|\varphi)\widetilde{\pi}(x|\theta, y)}{\pi(\theta)\nu(x|\theta)q(\varphi|\theta)\nu(w|\varphi)\nu(y|x)\widetilde{\pi}(w'|\theta)\widetilde{\pi}(x'|\varphi, y)},\end{aligned}$$

i.e. identical to (15). Therefore, such a distinct perspective is feasible for these two strategies.

4 Simulation examples

In this section, we apply different parameter settings for $\theta = (\theta_0, \theta_1)$ and σ to simulate hidden variables x and noisy observations y from $\pi(x, y|\theta) = \pi(x|\theta)\pi(y|x)$ under the model conditions given in Section 3. On the basis of these realizations, we use the simulated noisy data y to estimate parameters θ and latent values x from the posterior distribution $\pi(\theta, x|y)$, and compare these simulations with the true values of x and θ that was used to generate y. In the backward process that generates θ and x from $\pi(\theta, x|y)$, we try two different methods

– simulation strategy I and simulation strategy II, and the main goal of our study is to compare the efficiencies for these two simulation strategies rather than how much useful the strategies are. Specifically, for both strategies, we adopt the improper prior $\pi(\theta_0, \theta_1) \propto 1$, and let the proposal $q(\varphi|\theta)$ be a bivariate Gaussian distribution in which $\varphi_0|\theta \sim N(\theta_0, \sigma_*^2)$ and $\varphi_1|\theta \sim N(\theta_1, \sigma_*^2)$ independently, where $\varphi = (\varphi_0, \varphi_1)$ and σ_* is a tuning parameter. For each of the parameter settings, we choose a suitable σ_* such that the overall acceptance probability in every simulation is approximately equal to 0.25. The CPU time (minutes) spent on 500 iterations is recorded, and we use estimated autocorrelation functions (ACFs) and the ratio of the Effective Sample Size (ESS) (Kass et al., 1998) to the simulation sample size per minute for mixing diagnostics. Here we just briefly introduce the basic concepts and definitions of ACFs and ESS. Define a collection of the samples of a Markov chain $\{x^b\}_{b=1}^{B}$, then the sample autocovariance function of lag h is defined by

$$\hat{\gamma}(h) = \frac{1}{B} \sum_{b=1}^{B-h} \left(x^b - \bar{x}\right)\left(x^{b+h} - \bar{x}\right), \quad 0 \leq h < B,$$

where $\bar{x} = \frac{1}{B}\sum_{b=1}^{B} x^b$. The sample autocorrelation of lag h is thus defined in terms of the sample autocovariance function

$$\hat{\rho}(h) = \frac{\hat{\gamma}(h)}{\hat{\gamma}(0)}, \quad |h| < B.$$

Thereby, we can define ESS which is a closely related measure of mixing as follows,

$$\text{ESS} = \frac{B}{\tau} = \frac{B}{1 + 2\sum_{h=1}^{\infty} \rho(h)},$$

where $\rho(h)$ is the autocorrelation of lag h, and τ is referred to as the autocorrelation time. To estimate τ, the Bayesian procedures first find a cutoff point k after which the autocorrelations are very close to zero, and then sum all the $\rho(h)$ up to that point. The cutoff point k is typically such that $\rho(k) < 0.01$ or $\rho(k) < 2s_k$, where s_k is the estimated standard deviation given by

$$s_k = 2\sqrt{\frac{1}{B}\left(1 + 2\sum_{h=1}^{k-1} (\rho(h))^2\right)}.$$

Small discrepancy between ESS and the simulation sample size indicates good mixing. When it comes to evaluating convergence, we just choose samples in equilibrium by observing trace plots to approximately skip burn-in periods. Moreover, all simulations are run in the same computer, and the time consumed in each iteration is assumed to be identical. In the following we first consider the parameter settings with $\theta_0 = 0$ and then with $\theta_0 = 0.5$. For both cases, the number of iterations after the burn-in period for each simulation of θ is chosen to be 500 counted from the 500th iteration to the 999th iteration. Then due to the number of these samples constantly equal to 500, we just need ESS per minute to judge if the mixing is good in lieu of the fraction of ESS to the simulation sample size per minute.

4.1 Simulations in the parameter settings with $\theta_0 = 0$

In addition to setting $\theta_0 = 0$, we consider four combinations of (θ_1, σ) used to simulate four realizations of x and four realizations of y by perfect sampling. These four combinations are set by $(\theta_1, \sigma) = (0.4, 0.3)$, $(0.7, 0.3)$, $(0.4, 0.6)$ and $(0.7, 0.6)$, respectively. In Figure 14, observing each of the four rows, the left scene is a simulation x and the right is the corresponding noisy observation y generated from the autologistic model of the form (11) with fixed $\theta_0 = 0$ and a certain value of θ_1 on a 100×100 lattice. The four scenes on the left from top to bottom are generated with $\theta_1 = 0.4, 0.7, 0.4$ and 0.7, respectively; The four scenes from top to bottom on the right are based on $\sigma = 0.3, 0.3, 0.6$ and 0.6, respectively. The data shown in the four scenes on the right-hand panel in Figure 14 is used for the parameter estimation, and the values of σ (0.3 and 0.6) are assumed to be known in all computer simulations.

Figure 15 presents the trace plots of θ generated from $\pi(\theta, x|y)$ for each of the noisy observations under the parameter setting $\theta_0 = 0$. Red lines stand for the trace plots of θ_0, and blue lines represent that of θ_1. The four scenes on the left in this figure are simulated by simulation strategy I, and the four scenes on the right by simulation strategy II. From top to bottom in this figure, two trace plots of θ in any row are generated according to the observation y that locates in the same row on the right in Figure 14. Comparing two scenes in each row in Figure 15, we see that it takes larger number of iterations to pass the burn-in period for simulation strategy I than that for simulation strategy II. In addition, the trace plots in all cases show that any simulated chain of θ_0 or θ_1 can reach its stationary states after the 500th iteration, so it is sufficient to use 500 samples from the 500th iteration to the

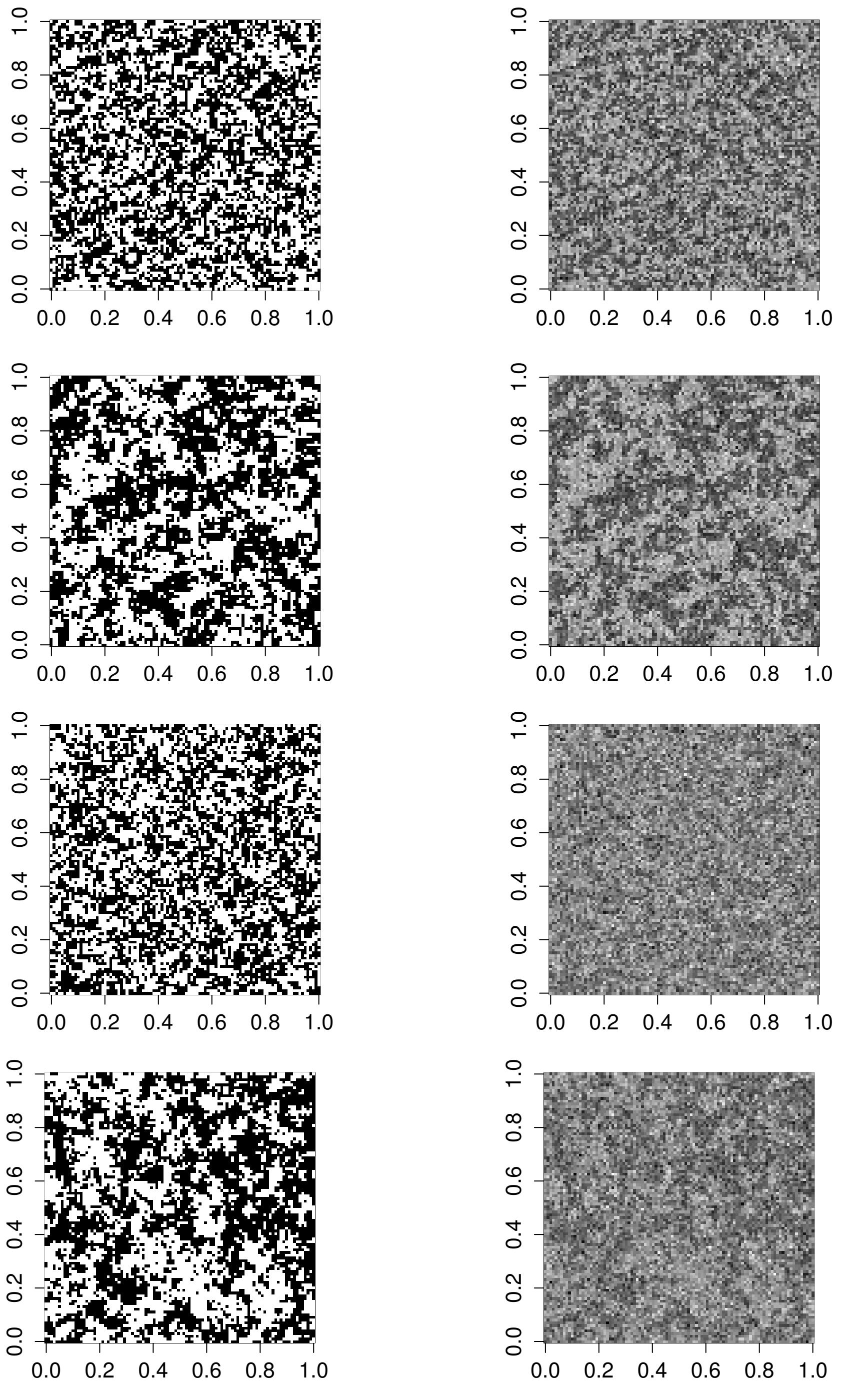


Figure 14: Simulations x and their noisy observations y of the autologistic model of the form (11) with fixed $\theta_0 = 0$ and different θ_1 on a 100×100 lattice. In each of four rows in this figure, the left scene is a simulation x and the right is the corresponding noisy observation y. The four scenes on the left from top to bottom are generated with $\theta = 0.4, 0.7, 0.4$ and 0.7, respectively; The four scenes from top to bottom on the right are based on $\sigma = 0.3, 0.3, 0.6$ and 0.6, respectively.

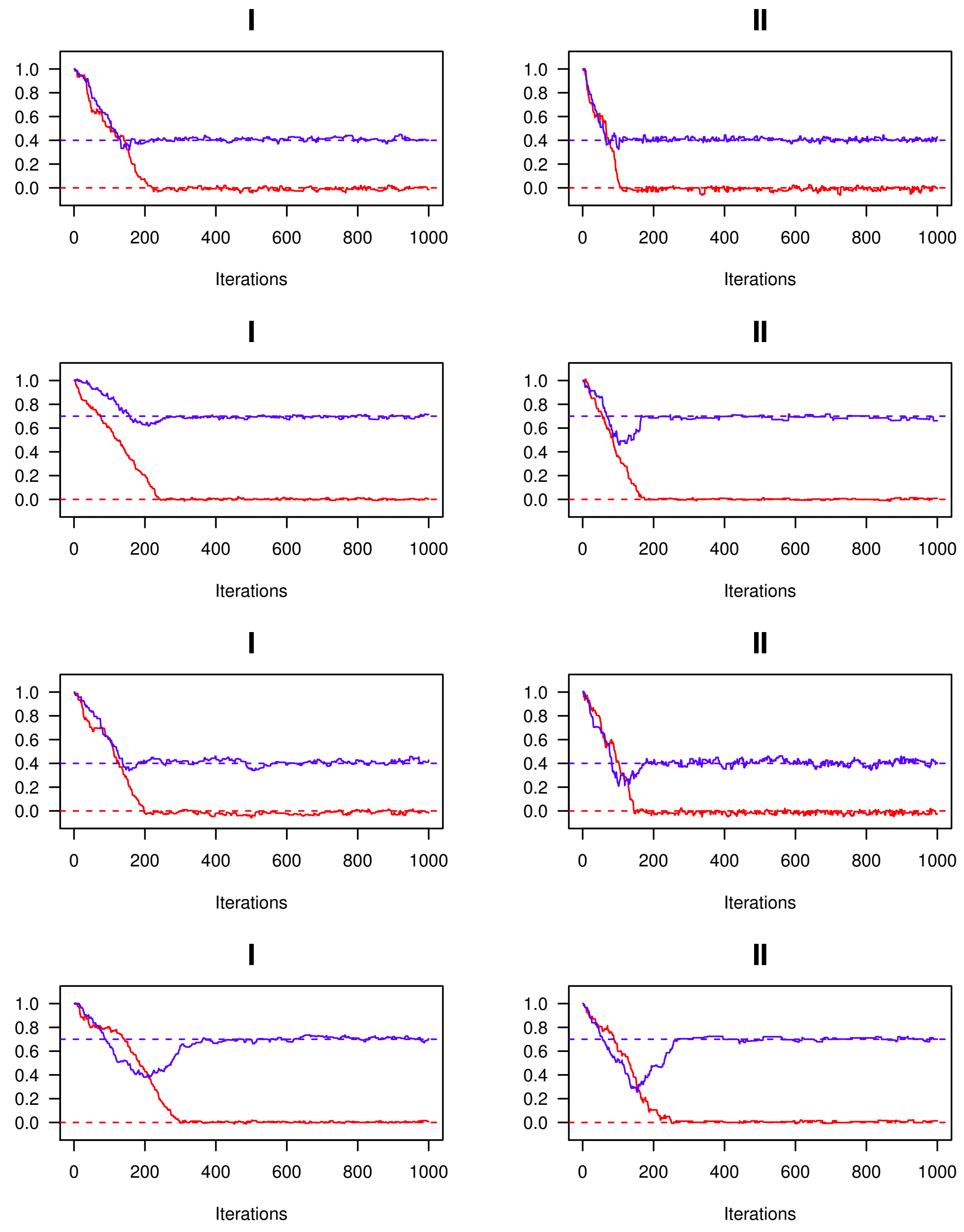


Figure 15: Trace plots of θ generated from $\pi(\theta, x|y)$ for each of the noisy observations under the parameter setting $\theta_0 = 0$. Red and blue solid lines represent the trace plots of θ_0 and θ_1, respectively. Red and blue dashed lines represent the true values of θ_0 and θ_1, respectively. The four scenes on the left in this figure are simulated by simulation strategy I, and the four scenes on the right by simulation strategy II. From top to bottom in this figure, two trace plots of θ in any row are generated according to the observation y in the same row on the right in Figure 14.

Table 1: The choices of tuning parameter σ_* in the four cases under the parameter setting $\theta_0 = 0$ using simulation strategy I and in the four cases using simulation strategy II. Roman numbers I and II are the abbreviations of simulation strategy I and simulation strategy II, respectively.

		$\theta_1 = 0.4$	$\theta_1 = 0.7$
I	$\sigma = 0.3$	0.018	0.011
	$\sigma = 0.6$	0.017	0.011
II	$\sigma = 0.3$	0.028	0.021
	$\sigma = 0.6$	0.024	0.018

999th iteration as we mention before.

The values of tuning parameter σ_* for different combinations of parameters (θ_1, σ) are shown in Table 1. To study the stationary distribution of θ_0 is not our main purpose in this thesis, but it is still of interest, so we make histograms in Figure 16 for samples of θ_0 based on the four noisy observations. From top to bottom in this figure, the scenes in the four rows correspond to the noisy observations generated by setting (θ_1, σ) to $(0.4, 0.3)$, $(0.7, 0.3)$, $(0.4, 0.6)$ and $(0.7, 0.6)$, respectively. The four scenes on the left in this figure are from the samples of θ_0 generated by simulation strategy I, and the four scenes on the right by simulation strategy II. For every row, the variance caused by simulation strategy II is slightly greater than that by simulation strategy I due to Monte Carlo simulation errors. Observing the first row and the third row, or the second and the fourth in Figure 16, it is simple to realize that the variance of θ_0 generated with $\sigma = 0.6$ (larger value) is greater than that with $\sigma = 0.3$ (small value). Figure 17 shows the histograms for samples of θ_1. By analogy, we can draw the same conclusions for θ_1 as in Figure 16.

The estimated ACFs for θ_0 are shown in Figure 18. From top to bottom, the scenes in the four rows correspond to the noisy observations generated by setting (θ_1, σ) to $(0.4, 0.3)$, $(0.7, 0.3)$, $(0.4, 0.6)$ and $(0.7, 0.6)$, respectively. The four scenes on the left in this figure are estimated from θ_0 generated by simulation strategy I, and the four scenes on the right by simulation strategy II. The results from simulation strategy II slip down to zero faster than that from simulation strategy I in terms of all the four parameter settings. Thus, the mixing of the Markov chains using simulation strategy II is better than that using simulation

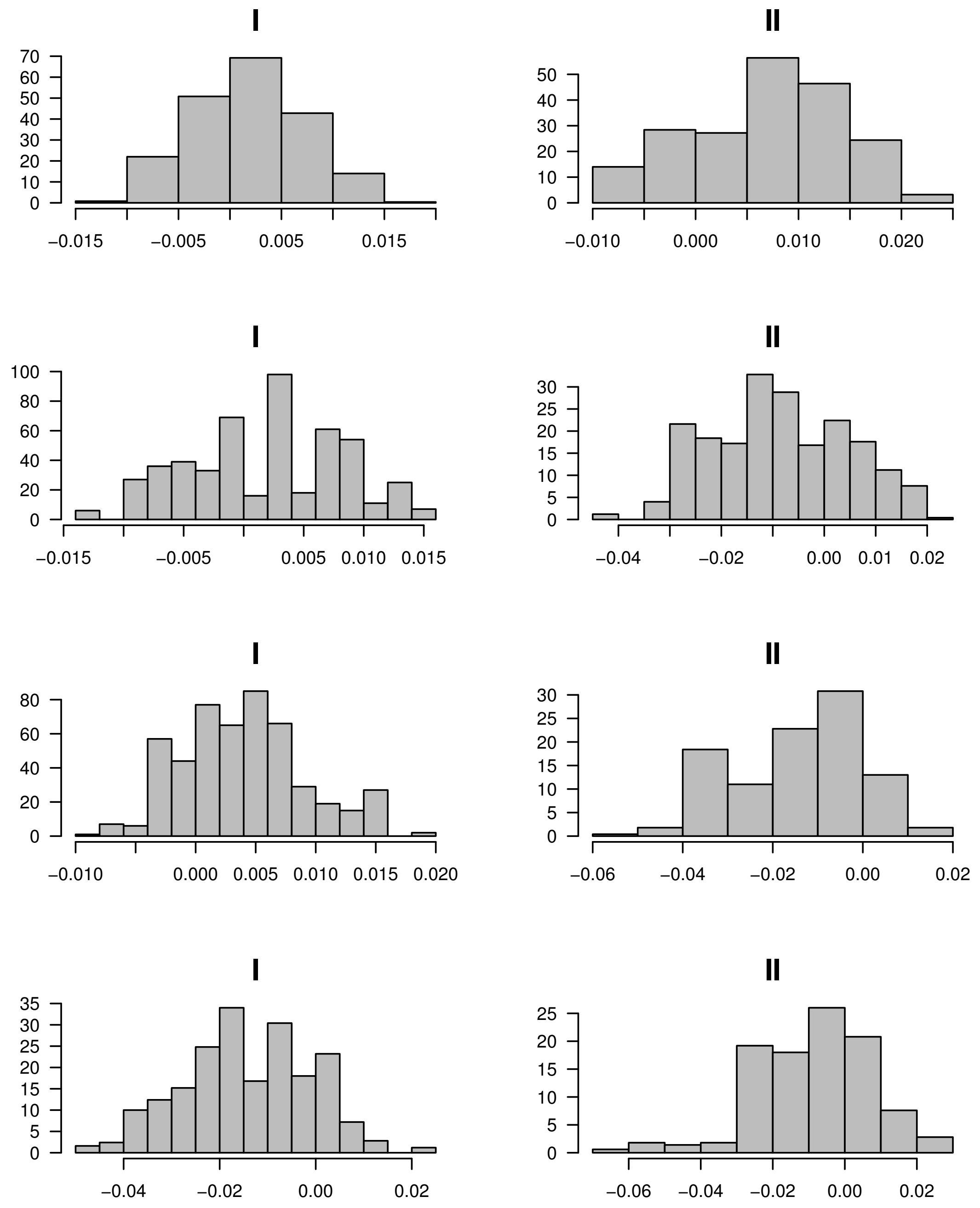


Figure 16: Histograms of samples of θ_0 under the parameter setting $\theta_0 = 0$. From top to bottom, the scenes in the four rows correspond to the noisy observations generated by setting (θ_1, σ) to $(0.4, 0.3)$, $(0.7, 0.3)$, $(0.4, 0.6)$ and $(0.7, 0.6)$, respectively. The four scenes on the left in this figure are from the samples of θ_0 generated by simulation strategy I, and the four scenes on the right by simulation strategy II.

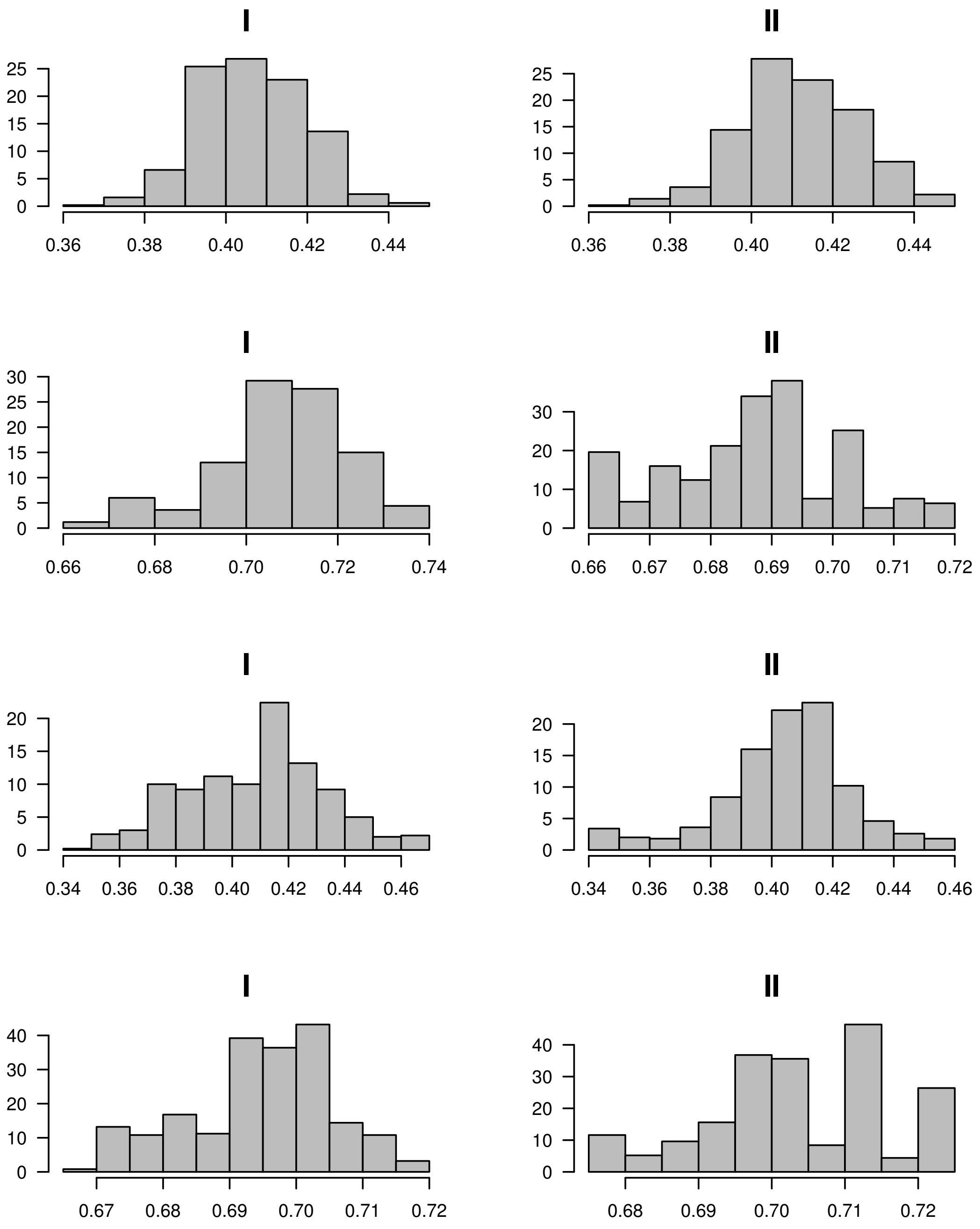


Figure 17: Histograms of samples of θ_1 under the parameter setting $\theta_0 = 0$. From top to bottom, the scenes in the four rows correspond to the noisy observations generated by setting (θ_1, σ) to $(0.4, 0.3)$, $(0.7, 0.3)$, $(0.4, 0.6)$ and $(0.7, 0.6)$, respectively. The four scenes on the left in this figure are from the samples of θ_1 generated by simulation strategy I, and the four scenes on the right by simulation strategy II.

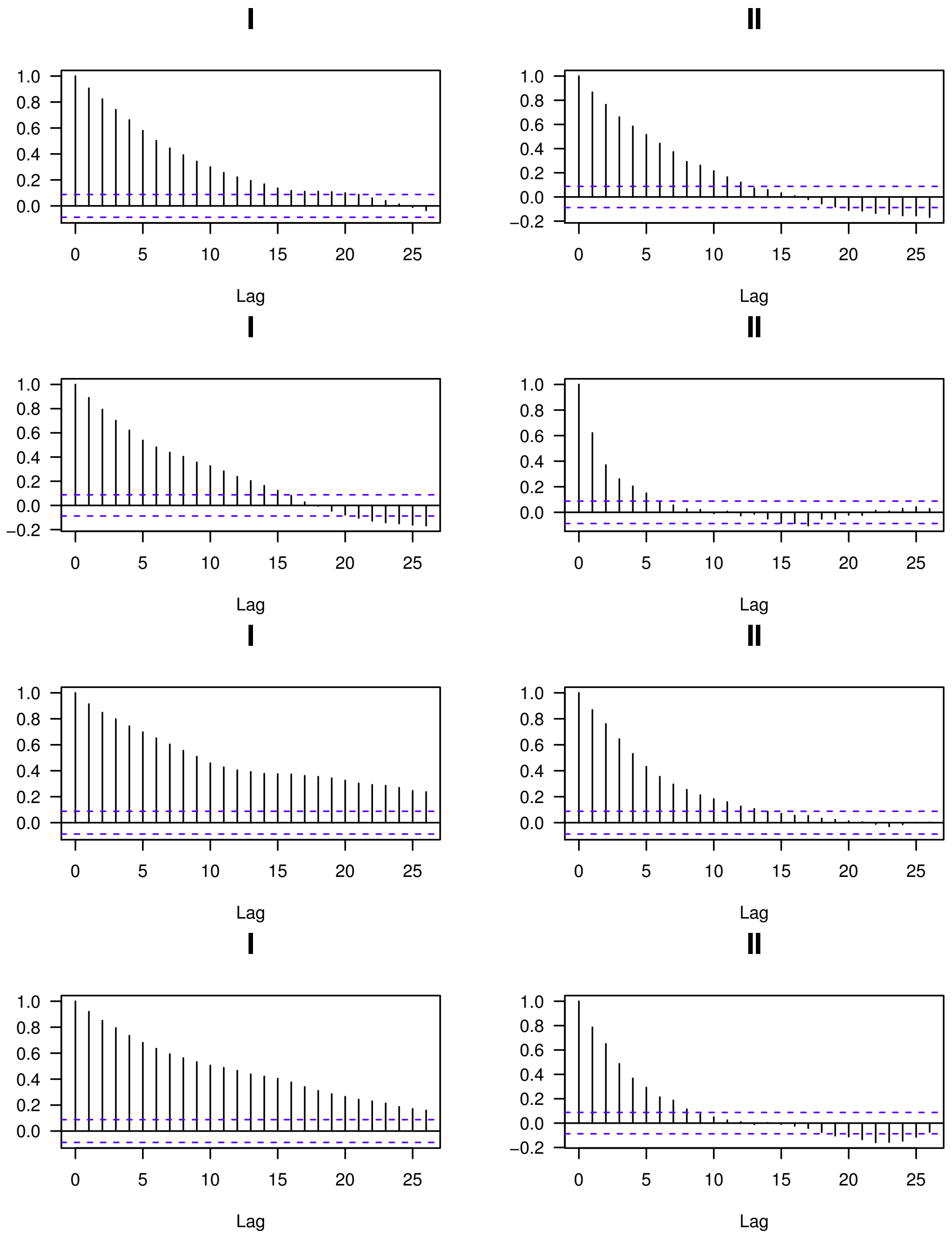


Figure 18: The estimated ACFs for θ_0 under the parameter setting $\theta_0 = 0$. From top to bottom, the scenes in the four rows correspond to the noisy observations generated by setting (θ_1, σ) to $(0.4, 0.3)$, $(0.7, 0.3)$, $(0.4, 0.6)$ and $(0.7, 0.6)$, respectively. The four scenes on the left in this figure are estimated from θ_0 generated by simulation strategy I, and the four scenes on the right by simulation strategy II.

Table 2: ESS, the CPU times, ESS per minute and their ratios under the parameter setting $\theta_0 = 0$. The CPU times are recorded for simulating 500 stationary samples. The calculation results in the second, third, fourth and fifth columns in this table are based on the four noisy observations on the right panel from top to bottom in Figure 14, respectively. CPU is the abbreviation of the CPU time. Roman numbers I and II represent simulation strategy I and simulation strategy II, respectively.

		(θ_0, θ_1)	(θ_0, θ_1)	(θ_0, θ_1)	(θ_0, θ_1)
I	ESS	$(35, 26)$	$(42, 31)$	$(20, 17)$	$(34, 28)$
	CPU	5.28	19.27	6.65	21.12
	ESS/CPU	$(17.50, 13.00)$	$(2.18, 1.61)$	$(3.01, 2.56)$	$(1.61, 1.33)$
II	ESS	$(67, 70)$	$(128, 130)$	$(34, 43)$	$(99, 106)$
	CPU	202.01	215.72	202.21	219.3
	ESS/CPU	$(0.33, 0.35)$	$(0.59, 0.60)$	$(0.17, 0.21)$	$(0.45, 0.49)$
ESS(II)/ESS(I)		$(1.91, 2.69)$	$(3.05, 4.19)$	$(1.70, 2.53)$	$(2.91, 3.79)$
CPU(II)/CPU(I)		38.26	11.19	30.41	10.38
(ESS/CPU)(I)/(ESS/CPU)(II)		$(53.03, 37.14)$	$(3.69, 2.68)$	$(17.71, 12.19)$	$(3.58, 2.71)$

strategy I. On the right-hand panel, comparing the first row with the second, or the third with the fourth, i.e. keep σ fixed, we see that the estimated ACFs with $\theta_1 = 0.7$ decrease more dramatically than that with $\theta_1 = 0.4$. Hence, when the value of θ_1 is larger, Markov chains simulated by simulation strategy II give more satisfactory mixing since the larger value of θ_1 implies the stronger dependency between θ and x, and thus it is more efficient to use simulation strategy II. The ACFs for θ_1 are depicted in Figure 19, we use the same organization, and can obtain the same conclusions as before.

Table 2 summarizes the information about the mixing of the Markov chains and relevant CPU times derived from the simulations. The data shown in the second, third, fourth and fifth columns in this table are based on the four noisy observations on the right panel from top to bottom in Figure 14, respectively, i.e. $(\theta_1, \sigma) = (0.4, 0.3), (0.7, 0.3), (0.4, 0.6)$ and $(0.7, 0.6)$, respectively. For convenience, we use CPU and Roman numbers I and II to represent the total CPU time for simulating 500 stationary samples, simulation strategy I and simulation strategy II, respectively. Given a fixed column, comparing rows vertically, the ratios of

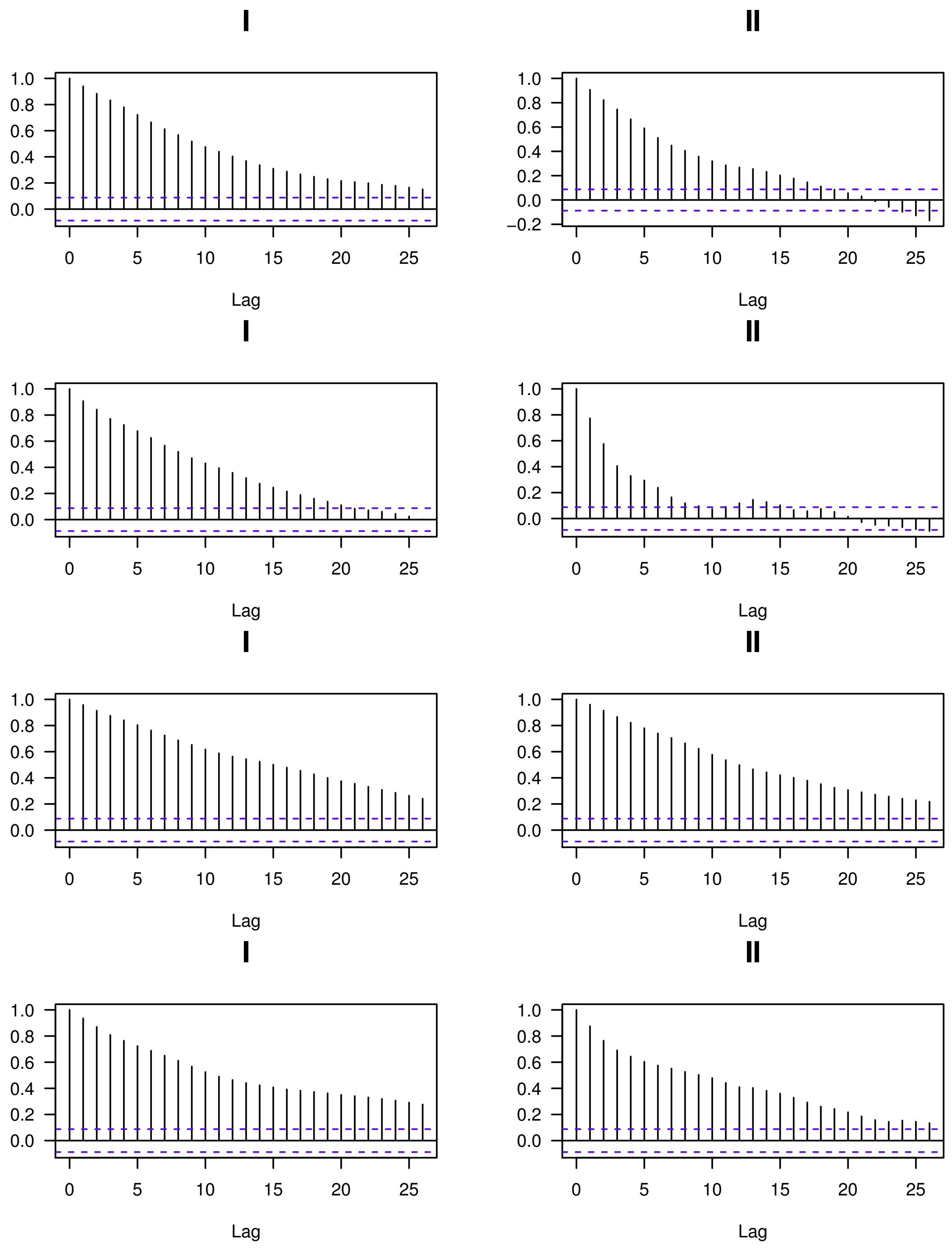


Figure 19: The estimated ACFs for θ_1 under the parameter setting $\theta_0 = 0$. From top to bottom, the scenes in the four rows correspond to the noisy observations generated by setting (θ_1, σ) to $(0.4, 0.3)$, $(0.7, 0.3)$, $(0.4, 0.6)$ and $(0.7, 0.6)$, respectively. The four scenes on the left in this figure are estimated from θ_1 generated by simulation strategy I, and the four scenes on the right by simulation strategy II.

ESS for simulation strategy II to that for simulation strategy I are larger than 1 with the maximum value equal to 4.19. This indicates the mixing resulted from simulation strategy II is better that from simulation strategy I. However, the CPU time and the values of ESS per minute for simulation strategy I are greater than that for simulation strategy II, so one should select simulation strategy I if taking into account time consumptions and the efficiency of simulations. Studying the third and sixth rows – the CPU times for two strategies, all of the CPU times spent on simulation strategy II are beyond 200 minutes, and their relative differences are not very big. For instance, the relative difference of CPU times between 202.01 and 219.3 is the greatest for simulation strategy II, i.e. $\frac{219.3-202.01}{202.01} \approx 0.086$. However, such differences for simulation strategy I cannot be ignored since the minimum difference value is based on 19.27 and 21.12, i.e. $\frac{21.12-19.27}{19.27} \approx 0.096$ which is larger than the maximum value 0.086 for simulation strategy II. Thus, the computation time used to calculate POMM approximations plays a dominant role in the total CPU time. Comparing the second column ($\theta_1 = 0.4$) with the third column ($\theta_1 = 0.7$), or the fourth column ($\theta_1 = 0.4$) with the fifth column ($\theta_1 = 0.7$) in terms of ESS(II)/ESS(I), the discrepancy of mixing between simulation strategy I and simulation strategy II is larger and larger as the value of θ_1 increases since θ_1 affects the dependency between θ and x. In other words, the greater value of θ implies the stronger correlation between θ and x, so using simulation strategy II with block updates is more efficient verifying the assumption in the end of Section 3.1. With respect to (ESS/CPU)(I)/(ESS/CPU)(II), $53.03 > 3.69, 37.14 > 2.68$ and $17.71 > 3.58, 12.19 > 2.71$, so the superiority for simulation strategy I in ESS per minute is increasingly weak with the larger value of θ_1.

The visual differences between the true x shown on the left-hand panel in Figure 14 and simulated x illustrated as gray-scale images are insignificant whichever strategy is used. The simulated x is picked from the final sample of x from $\pi(\theta, x|y)$, i.e. the sample is the 999th iteration. In Table 3 we present the number of vertices in the 100×100 lattice that are not identical in the plot of the true x and its counterpart. The proportion of different vertices is small since $\frac{30}{10^4} = 0.3\% \ll 1$. A larger value of ESS indicates better mixing, so obviously, simulation strategy II generates stationary chains with more satisfactory mixing in all our four cases than simulation strategy I. The strategy with a smaller value of ESS per minute

Table 3: The number of different vertices in the 100×100 lattice between the true x and simulated x for simulation strategy I and simulation strategy II under the parameter setting $\theta_0 = 0$.

	$(\theta_1, \sigma) = (0.4, 0.3)$	$(\theta_1, \sigma) = (0.7, 0.3)$	$(\theta_1, \sigma) = (0.4, 0.6)$	$(\theta_1, \sigma) = (0.7, 0.6)$
I	0	0	28	30
II	0	0	28	30

owns more useful availabilities, so one should choose simulation strategy I in practice.

4.2 Simulations in the parameter settings with $\theta_0 = 0.5$

In this section, for a fair comparison, we do simulations with the same configurations as in the last section except the setting $\theta_0 = 0.5$. The alignment structures and symbols in all figures in this section are the same as their corresponding figures in Section 4.1. For instance, the group of trace plots in this section has the same arrangement for its subfigures as that in Figure 15 in the last section. If there exist some differences, we will point them out. Moreover, the conclusions and results in this section which are the same as that in Section 4.1 are not fully stated or they are presented by a few words.

We set $\theta_0 = 0.5$ so as to make the proportion of 1's (black pixels) equal to 70% to 80% in a 100×100 lattice. The parameter settings of (θ_1, σ) are the same as that in Section 4.1. Figure 20 shows four simulations of x and their corresponding noisy observations y of the autologistic model of the form (11) with fixed $\theta_0 = 0.5$ and different values of θ_1 on a 100×100 lattice. Compared with Figure 14, there are more black pixels in each scene on the left in Figure 20 since $\theta_1 = 0.5 > 0$ determines the proportion of 1's as we discuss in the example in Section 2.3.

The trace plots of θ_0 and θ_1 under the conditions of four parameter settings are shown in Figure 21. In addition to the same conclusions as for Figure 15, we find out that Markov chains simulated by simulation strategy II on the right in Figure 21 converge to their corresponding stationary distributions faster than that on the right in Figure 15 as $\theta_0 \neq 0$ enhances the dependency between θ and x, and thereby simulation strategy II with the blocking scheme is more efficient. However, in this figure the chains simulated by simulation

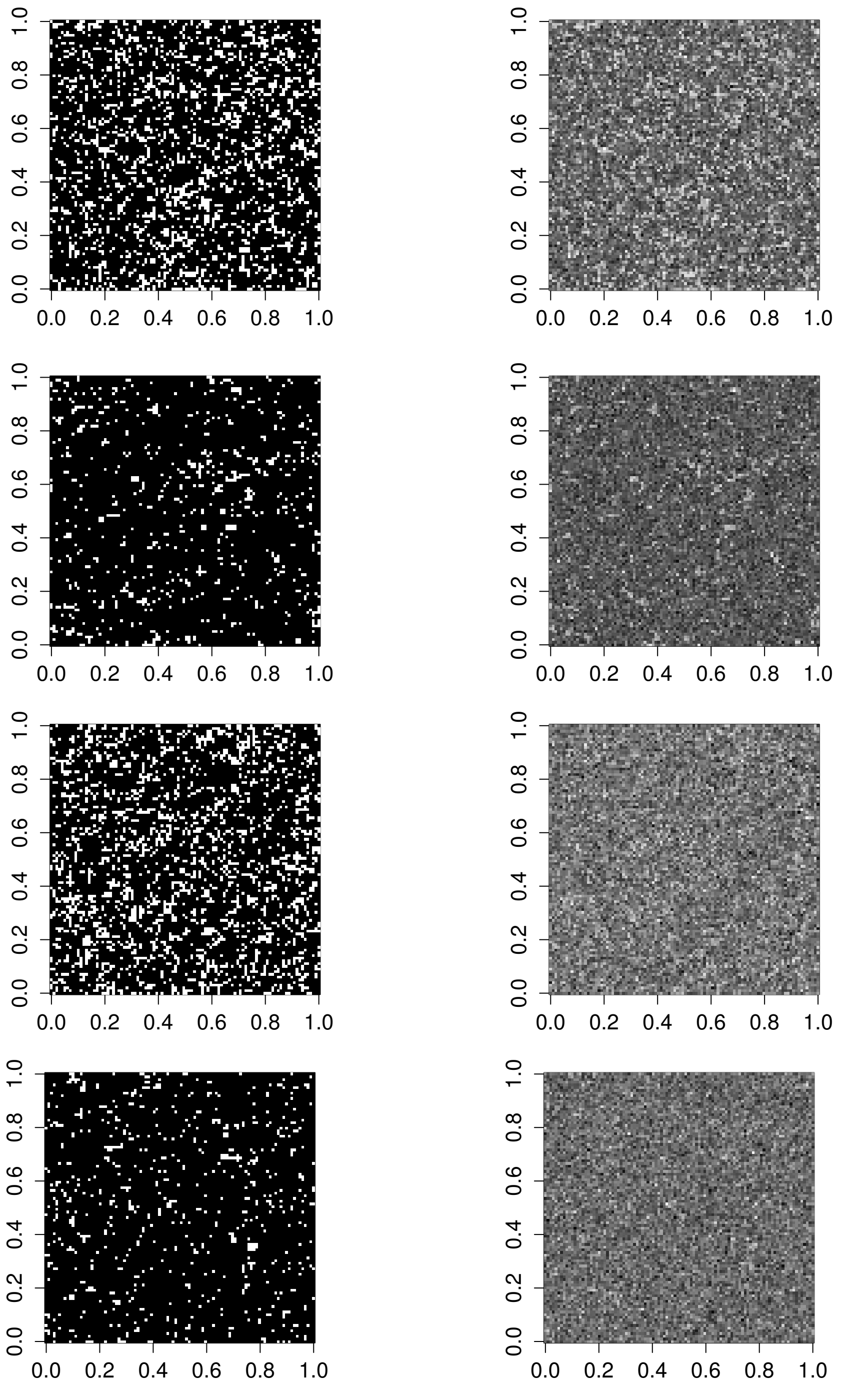


Figure 20: Simulations x and their noisy observations y of the autologistic model of the form (11) with fixed $\theta_0 = 0.5$ and different θ_1 on a 100×100 lattice. In each of four rows in this figure, the left scene is a simulation x and the right is the corresponding noisy observation y. The four scenes on the left from top to bottom are generated with $\theta = 0.4, 0.7, 0.4$ and 0.7, respectively; The four scenes from top to bottom on the right are based on $\sigma = 0.3, 0.3, 0.6$ and 0.6, respectively.

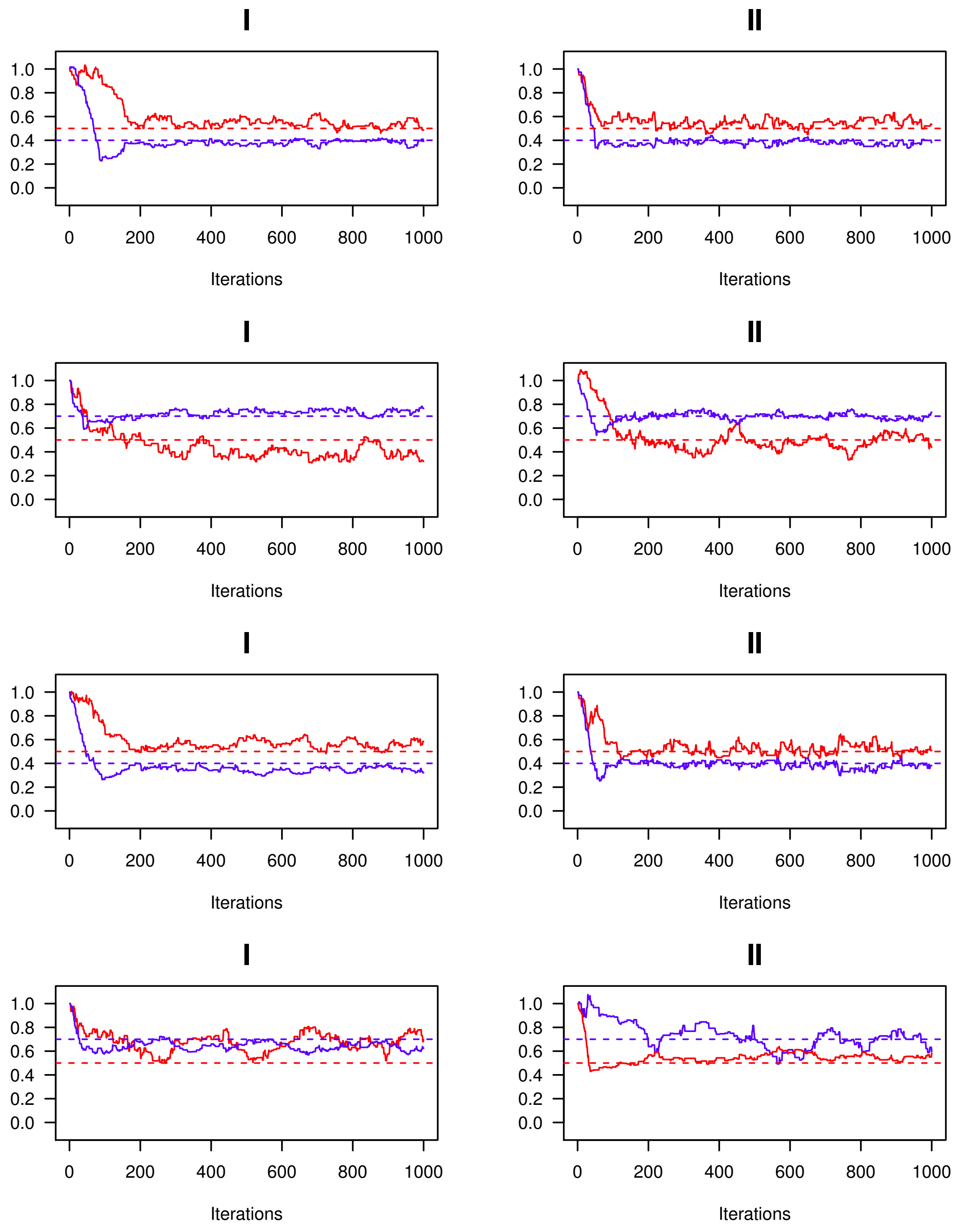


Figure 21: Trace plots of θ generated from $\pi(\theta, x|y)$ for each of the noisy observations under the parameter setting $\theta_0 = 0.5$. Red and blue solid lines represent the trace plots of θ_0 and θ_1, respectively. Red and blue dashed lines represent the true values of θ_0 and θ_1, respectively. The four scenes on the left in this figure are simulated by simulation strategy I, and the four scenes on the right by simulation strategy II. From top to bottom in this figure, two trace plots of θ in any row are generated according to the observation y in the same row on the right in Figure 20.

Table 4: The choices of tuning parameter σ_* in the four cases under the parameter setting $\theta_0 = 0.5$ using simulation strategy I and in the four cases using simulation strategy II. Roman numbers I and II are the abbreviations of simulation strategy I and simulation strategy II, respectively.

		$\theta_1 = 0.4$	$\theta_1 = 0.7$
I	$\sigma = 0.3$	0.023	0.023
	$\sigma = 0.6$	0.023	0.025
II	$\sigma = 0.3$	0.031	0.032
	$\sigma = 0.6$	0.030	0.035

strategy I without blocking converges in the almost same speed as or even more slowly than that on the left in Figure 15.

In the rest of this section, the histograms of samples for θ_0 are shown in Figure 22. The variance of θ_0 generated by some strategy with some parameter setting in Figure 22 is larger than the variance of the corresponding one in Figure 16. The reasons can be explained as follows. Now that $\theta_0 = 0.5 \neq 0$, then both θ_0 and θ_1 affect the pattern of x. Reversely, it is hard to distinguish the influence coming from θ_0 and θ_1, and thereby the errors and the uncertainties should be greater than that in simulations with $\theta_0 = 0$. Moreover, large values of $|\theta_0|$ and θ_1 lead to large variances of samples for θ_0 and θ_1. For example, given a part of the lattice in which the values of all vertices are 1, then it is difficult to diagnose whether this pattern is mainly because of either a large value of $\theta_0 > 0$ or a large interaction effect caused by large θ_1. The situations in Figure 23 are the same as that in Figure 22, and can be interpreted by the same arguments. The values of tuning parameter σ_* for different combinations of parameters (θ_1, σ) when $\theta_0 = 0.5$ are shown in Table 4. In this table, the values of σ_* used for simulation strategy II are still larger than that for simulation strategy I. The difference between σ_* for strategy I and σ_* for strategy II in terms of a certain parameter setting in Table 4 is approximately equal to such a difference in Table 1. Therefore, the consistency for the choice of tuning parameter σ_* in all the cases for two different θ_0 holds.

The estimated ACFs for θ_0 are shown in Figure 24 and that for θ_1 are shown in Figure 25.

The estimated ACFs in Figure 24 and in Figure 25 reflect the mixing of the stationary

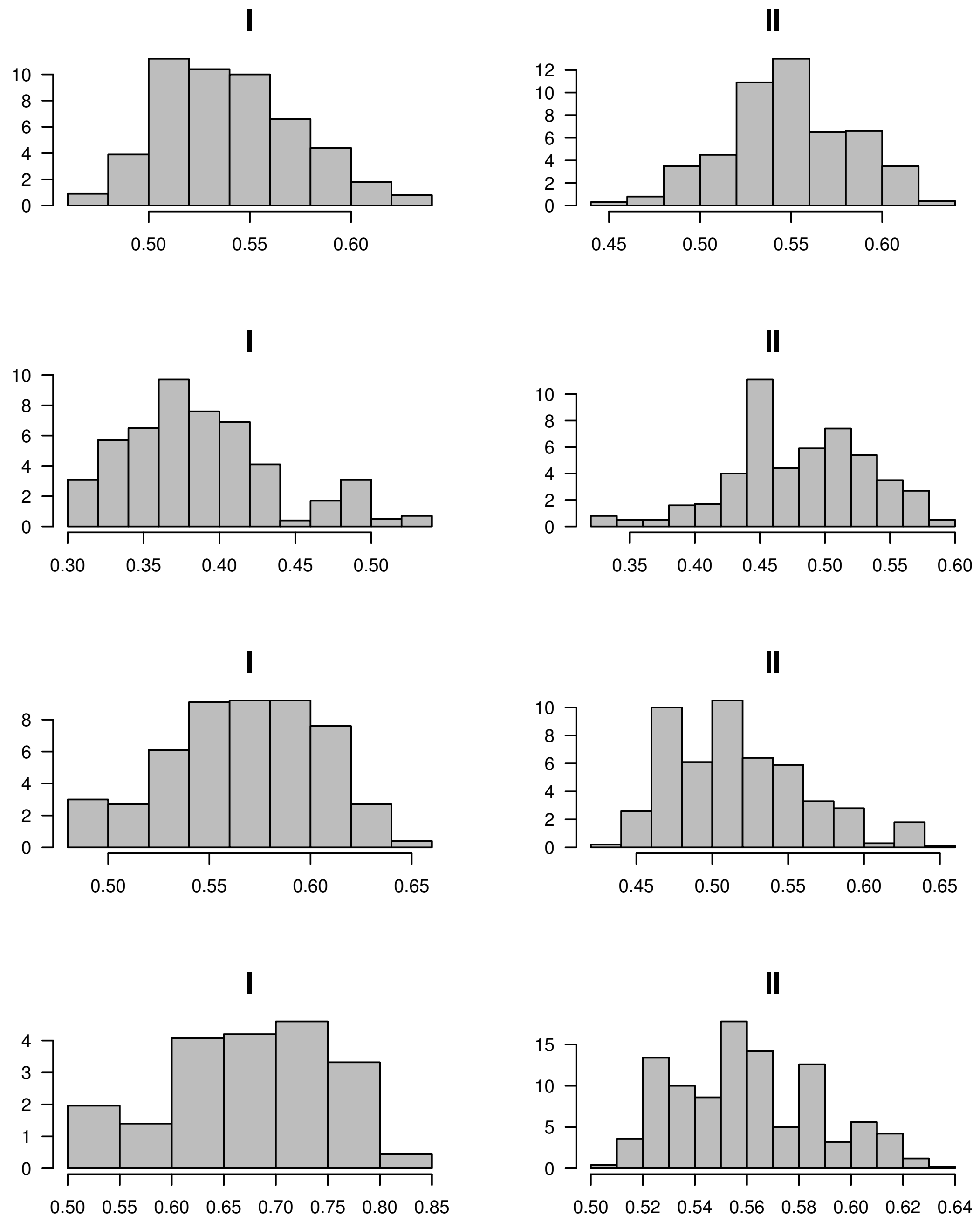


Figure 22: Histograms of samples of θ_0 under the parameter setting $\theta_0 = 0.5$. From top to bottom, the scenes in the four rows correspond to the noisy observations generated by setting (θ_1, σ) to $(0.4, 0.3)$, $(0.7, 0.3)$, $(0.4, 0.6)$ and $(0.7, 0.6)$, respectively. The four scenes on the left in this figure are from the samples of θ_0 generated by simulation strategy I, and the four scenes on the right by simulation strategy II.

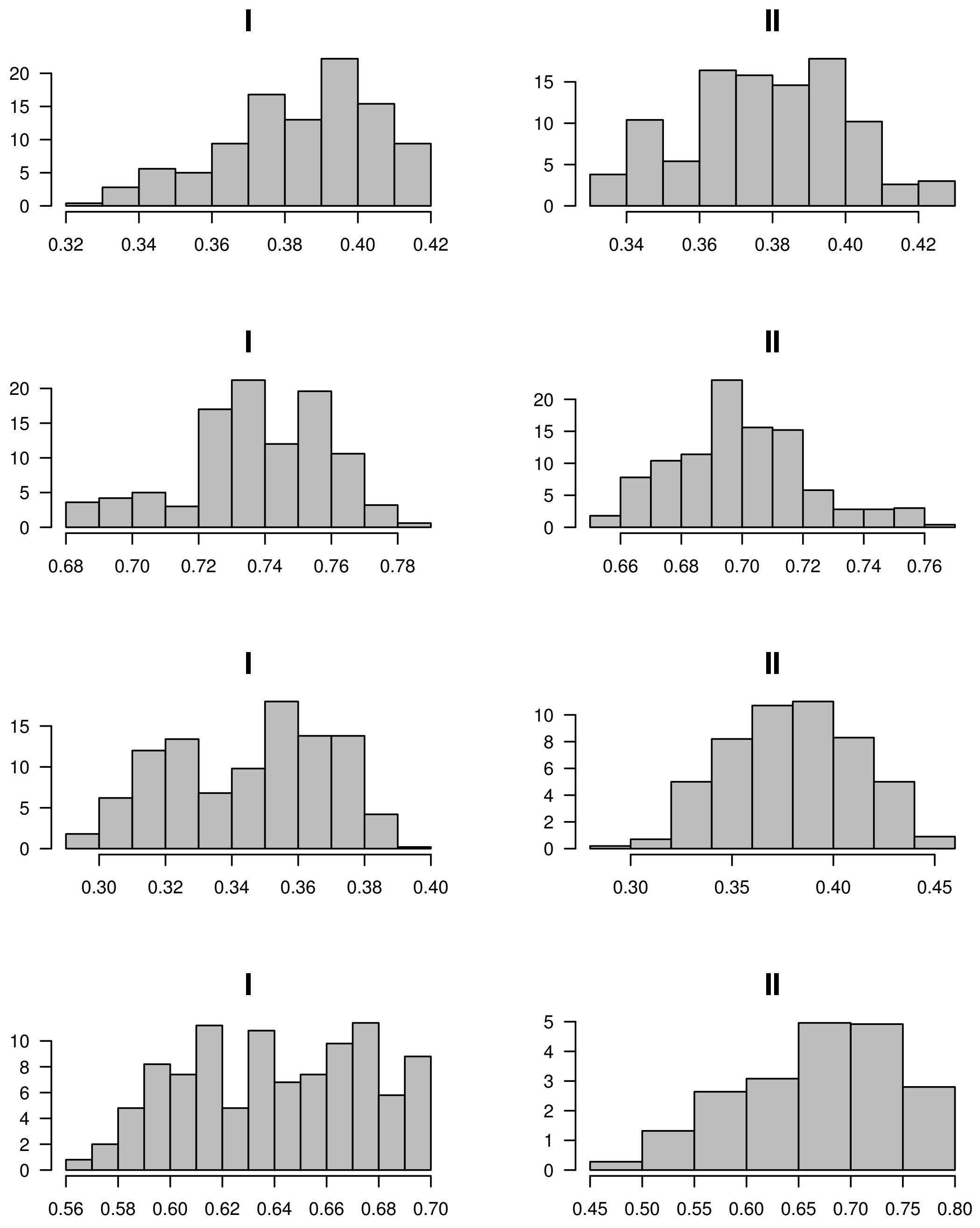


Figure 23: Histograms of samples of θ_1 under the parameter setting $\theta_0 = 0.5$. From top to bottom, the scenes in the four rows correspond to the noisy observations generated by setting (θ_1, σ) to $(0.4, 0.3)$, $(0.7, 0.3)$, $(0.4, 0.6)$ and $(0.7, 0.6)$, respectively. The four scenes on the left in this figure are from the samples of θ_1 generated by simulation strategy I, and the four scenes on the right by simulation strategy II.

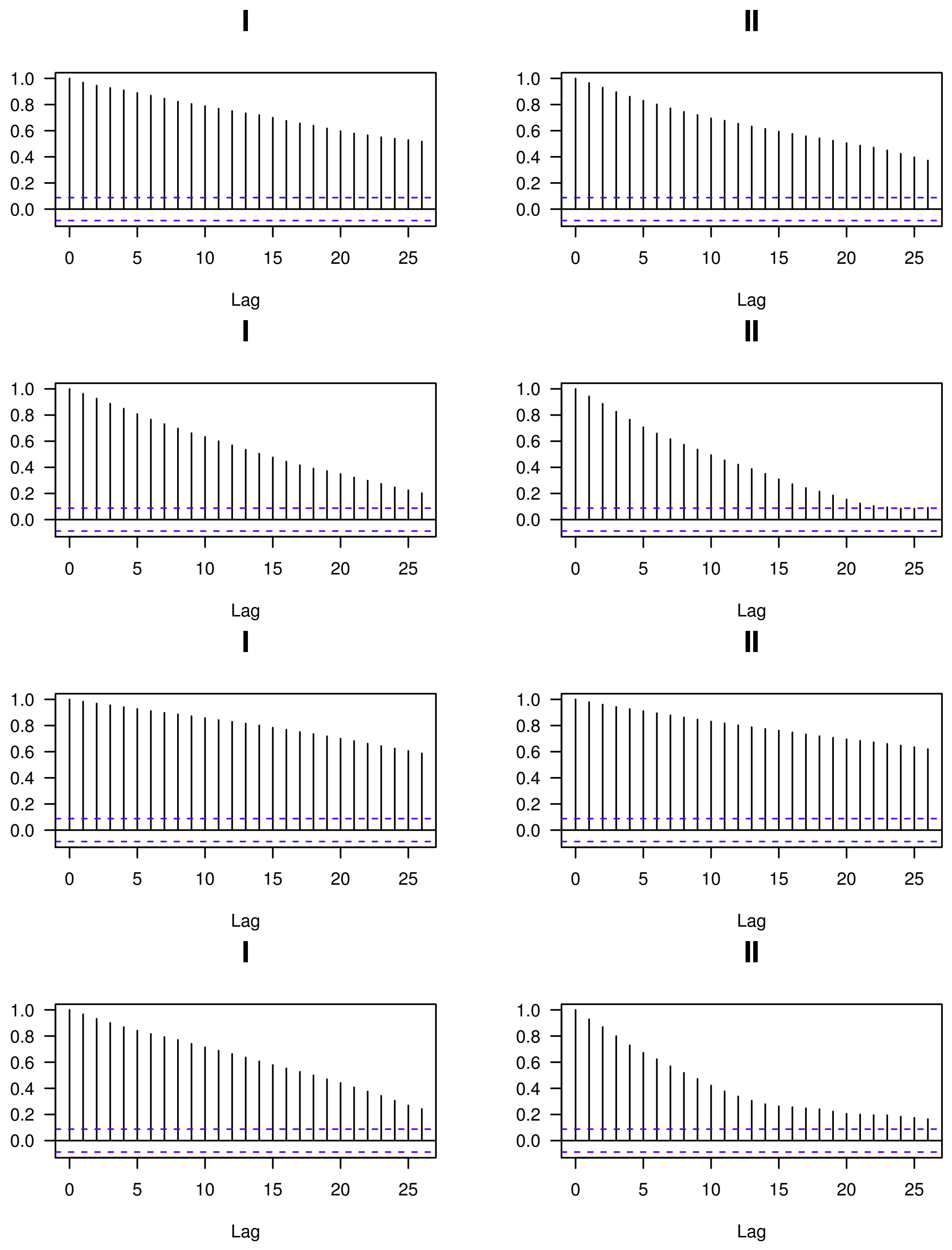


Figure 24: The estimated ACFs for θ_0 under the parameter setting $\theta_0 = 0.5$. From top to bottom, the scenes in the four rows correspond to the noisy observations generated by setting (θ_1, σ) to $(0.4, 0.3)$, $(0.7, 0.3)$, $(0.4, 0.6)$ and $(0.7, 0.6)$, respectively. The four scenes on the left in this figure are estimated from θ_0 generated by simulation strategy I, and the four scenes on the right by simulation strategy II.

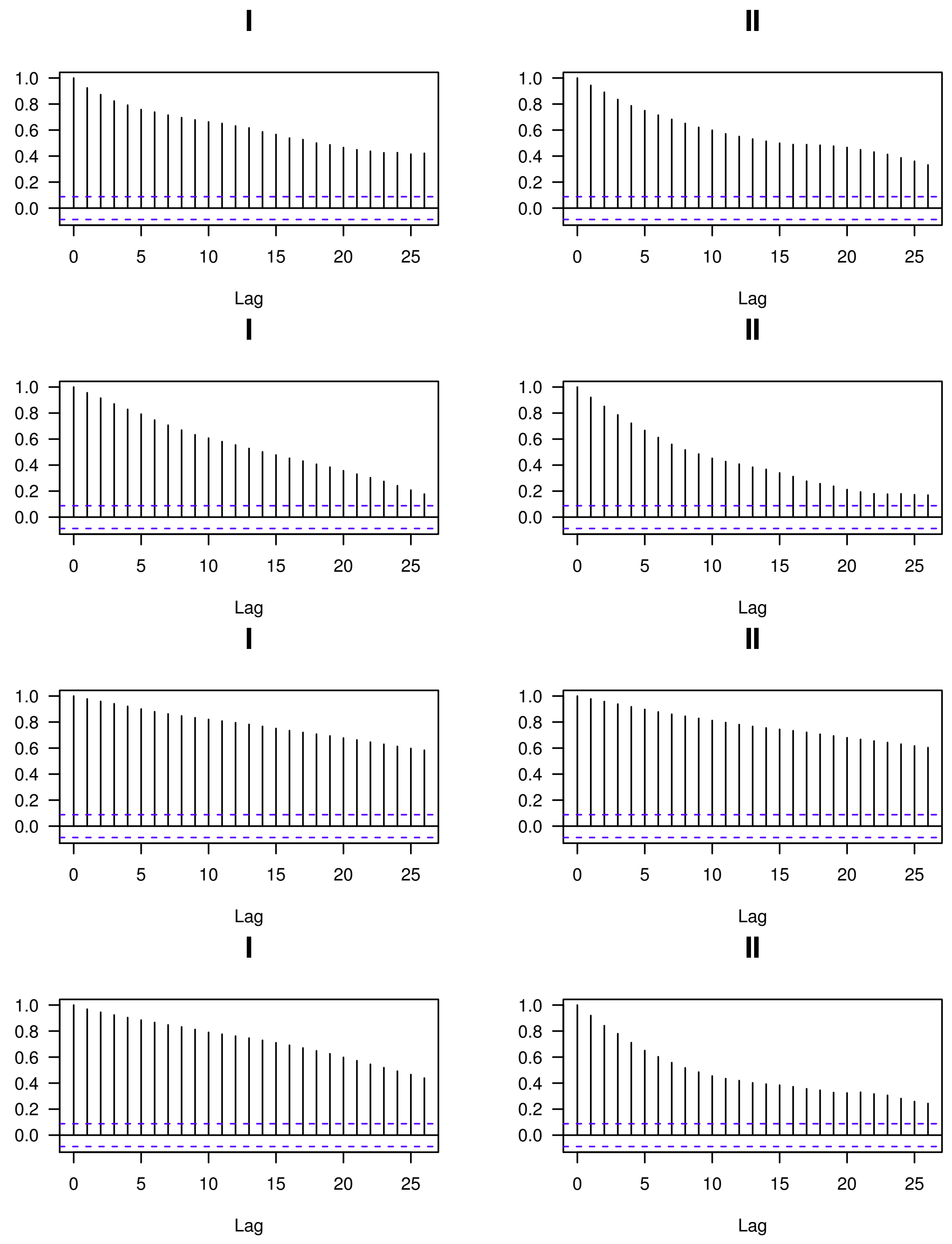


Figure 25: The estimated ACFs for θ_1 under the parameter setting $\theta_0 = 0.5$. From top to bottom, the scenes in the four rows correspond to the noisy observations generated by setting (θ_1, σ) to $(0.4, 0.3)$, $(0.7, 0.3)$, $(0.4, 0.6)$ and $(0.7, 0.6)$, respectively. The four scenes on the left in this figure are estimated from θ_1 generated by simulation strategy I, and the four scenes on the right by simulation strategy II.

Table 5: ESS, the CPU times, ESS per minute and their ratios under the parameter setting $\theta_0 = 0.5$. The CPU times are recorded for simulating 500 stationary samples. The calculation results in the second, third, fourth and fifth columns in this table are based on the four noisy observations on the right panel from top to bottom in Figure 20, respectively. CPU is the abbreviation of the CPU time. Roman numbers I and II represent simulation strategy I and simulation strategy II, respectively.

		(θ_0, θ_1)	(θ_0, θ_1)	(θ_0, θ_1)	(θ_0, θ_1)
I	ESS	$(20, 16)$	$(31, 22)$	$(16, 10)$	$(23, 20)$
	CPU	3.19	10.22	5.98	14.27
	ESS/CPU	$(6.27, 5.02)$	$(3.03, 2.15)$	$(2.68, 1.67)$	$(1.61, 1.40)$
II	ESS	$(43, 49)$	$(125, 138)$	$(27, 30)$	$(98, 104)$
	CPU	198.22	204.93	200.01	208.46
	ESS/CPU	$(0.22, 0.25)$	$(0.61, 0.67)$	$(0.13, 0.15)$	$(0.47, 0.50)$
ESS(II)/ESS(I)		$(2.15, 3.06)$	$(4.03, 6.27)$	$(1.67, 3.00)$	$(4.26, 5.20)$
CPU(II)/CPU(I)		62.14	20.05	33.44	14.61
(ESS/CPU)(I)/(ESS/CPU)(II)		$(28.50, 20.08)$	$(4.97, 3.21)$	$(20.62, 11.13)$	$(3.43, 2.80)$

chains of θ_0 and θ_1, respectively. The mixing in all cases with $\theta_0 = 0.5$ is not better than that with $\theta_0 = 0$, but according to the eighth rows (ratio of ESS) in both Table 2 and Table 5, $4.03 > 3.05, 6.27 > 4.19$ and $4.26 > 2.91, 5.20 > 3.79$ etc., the goodness of using simulation strategy II when $\theta_0 = 0.5$ is more significant than that when $\theta_0 = 0$. This implies and verifies that large $|\theta_0| \neq 0$ increases the dependency between θ and x. Comparing the third row and the sixth row in Table 5 with the corresponding rows in Table 2, respectively, we can observe that the relative reduction of CPU time using simulation strategy I is more drastic than that using simulation strategy II since it is easier to make a perfect sample when $\theta_0 \neq 0$. For example, an image produced from x with some pattern and its inverse image can be generated with the same probability if $\theta_0 = 0$, but it is more likely to generate the image with more black pixels if $\theta_0 > 0$, so this implies the relative reduction of CPU time $\frac{14.27-10.22}{14.27} \approx 0.28$(min for I) $> \frac{208.46-198.22}{208.46} \approx 0.049$(max for II) for simulation strategies I and II. The simulations in equilibrium when $\theta_0 = 0$ have less variance and better mixing than that when $\theta_0 = 0.5$.

Table 6: The number of different vertices in the 100×100 lattice between the true x and simulated x for simulation strategy I and simulation strategy II under the parameter setting $\theta_0 = 0.5$.

	$(\theta_1, \sigma) = (0.4, 0.3)$	$(\theta_1, \sigma) = (0.7, 0.3)$	$(\theta_1, \sigma) = (0.4, 0.6)$	$(\theta_1, \sigma) = (0.7, 0.6)$
I	0	0	39	54
II	0	0	39	54

In this case with $\theta_0 = 0.5$, in order to inspect different vertices between the true x on the left-hand panel in Figure 20 and simulated x from the last simulation iteration, we present the number of different vertices in the 100×100 lattice in Table 6. As we discuss before, when $\theta_0 = 0.5$, the variance of the simulations is greater than that in Section 4.1, so this is why $39 > 28$ and $53 > 30$.

5 Closing remarks

The Bayesian approach results in computational inefficiencies for hidden MRFs using conventional MCMC as normalization constants do not vanish when calculating the acceptance probabilities. The exchange algorithm is a useful excalibur when confronting normalizing constants, but the standard version of this algorithm is only applicable to the case without latent variables. Thus, we incorporate the exchange algorithm with MCMC schemes into our two strategies. Both of them are based on alternate usage of the Gibbs update and the MH update. The major difference between these two strategies is that simulation strategy I swaps two variables in one MH update while simulation strategy II interchanges all variables under some rule in one MH update, i.e. the latter uses the block MH update. For a fair comparison of our two strategies, it is important to take the CPU time used into consideration. Note that in strategy II with blocking we replace the exact distributions with the POMM approximations, so this means that if the approximations are close enough to the analytical results the distributions of interest will be sufficiently precise. Moreover, the complexity of the structures of POMM approximations we adopt determines the computation time of the simulations. In other words, if we select so sophisticated approximations that they are close to the true values, the simulation results should be "perfect", but it requires incredibly long computation time, and this is not what we need in practice. Therefore, we have to seek a trade-off between the precision and the temporal availability.

In our simulation examples, we have demonstrated that the strategy with blocking costs more CPU time than the other one under all parameter settings However, it is not rigorous to conclude that the strategy without blocking is more efficient than that with blocking only according to the CPU time. This is because mixing is also a crucial factor contributing to judging whether a stationary chain is good or not, i.e. samples are generated independently from our target distribution if a Markov chain has mixing of high quality. Therefore, we apply ESS per time unit given a fixed value of simulated sample size rather than the total CPU time after burn-in period. Note that for convergence diagnostics we only observe their trace plots, and then estimate approximate lengths of burn-in periods, but the mixing diagnostics are based on ESS. Whichever strategy, simulation strategy I or II, is used, simulation strategy I spends less CPU time and has greater values of ESS per minute. However, simulation strategy II can generate samples with better mixing. Furthermore, with respect to mixing, as the values of $|\theta_0|$ and θ_1 increase, the superiority that simulation strategy II is better than simulation strategy I becomes more and more significant. Based on the results and the conclusions in the previous sections, we speculate that the efficiency of using simulation strategy II will rise as the value of θ_1 increase, and will be identical to the efficiency of using simulation strategy I until θ_1 reach some threshold value, and after this value the efficiency of using simulation strategy II will become higher than that using simulation strategy I.

There are several approximations used for constructing the algorithmic framework of simulation strategy II, and the computation time is not acceptable compared with that in simulation strategy I. Hence, alternatively we can use other sorts of approximations such as particle Markov chain Monte Carlo methods (Andrieu et al., 2010) and approximate Bayesian computation (Beaumont et al., 2002), and other application is in Everitt (2012). In addition, we can also focus on reformulating the approximation of an MRF such that it is less computationally time-consuming in our further work. In the proposed models, we assume that the common variance σ^2 for Gaussian distributions from which noisy observations are generated is known. More generally, we do not need this assumption or we utilize a more complicated model to establish the statistical relationship between the hidden variables x

and the observed data y. Then take into account σ^2 as a stochastic variable, and denote $\theta = (\theta_0, \theta_1, \sigma^2)$, so that the conditional independence for y and θ does not hold, i.e. $\pi(y|x, \theta) \neq \pi(y|x)$. In the future, the work without knowing σ^2 can be done by similar procedure with what we have completed. In addition, we also have some other options for the further work, for instance, to improve the POMM approximations such that it spends less CPU time, to propose a new model or theory by which it is possible to get rid of using approximations for analytical distributions.

References

Andrieu, C., Doucet, A., and Holenstein, R. (2010), "Particle Markov chain Monte Carlo methods," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **72**, 269–342.

Ashkin, J. and Teller, E. (1943), "Statistics of two-dimensional lattices with four components," *Phys. Rev.*, **64**, 178–184.

Beaumont, M. A., Zhang, W., and Balding, D. J. (2002), "Approximate Bayesian computation in population genetics," *Genetics*, **162**, 2025–2035.

Berg, B. A. (2004), "Introduction to Markov chain Monte Carlo simulations and their statistical analysis," *eprint arXiv:cond-mat/0410490*.

Berthelsen, K. K. and Møller, J. (2003), "Likelihood and non-parametric Bayesian MCMC inference for spatial point processes based on perfect simulation and path sampling," *Scandinavian Journal of Statistics*, **30**, 549–564.

Besag, J. (1974), "Spatial interaction and the statistical analysis of lattice systems," *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, **36**, 192–236.

Carlin, B. P. and Chib, S. (1995), "Bayesian model choice via Markov chain Monte Carlo methods," *Journal of the Royal Statistical Society-Series B (Statistical Methodology)*, **57**, 473–484.

Chen, M.-H. and Shao, Q.-M. (1999), "Monte Carlo estimation of Bayesian credible and HPD intervals," *Journal of Computational and Graphical Statistics*, **8**, 69–92.

Cressie, N. and Davidson, J. L. (1998), "Image analysis with partially ordered Markov models," *Computational Statistics & Data Analysis*, **29**, 1–26.

Everitt, R. G. (2012), "Bayesian parameter estimation for latent Markov random fields and social networks," *Journal of Computational and Graphical Statistics*, **21**, 940–960.

Friel, N. and Rue, H. (2007), "Recursive computing and simulation-free inference for general factorizable models," *Biometrika*, **94**, 661–672.

Gelman, A. and Meng, X.-L. (1998), "Simulating normalizing constants: from importance sampling to bridge sampling to path sampling," *Statistical Science*, **13**, 163–185.

Geman, S. and Geman, D. (1984), "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **PAMI-6**, 721–741.

Green, P. J. and Richardson, S. (2001), "Modelling heterogeneity with and without the Dirichlet process," *Scandinavian Journal of Statistics*, **28**, 355–375.

Hammersley, J. M. and Clifford, P. E. (1971), "Markov random fields on finite graphs and lattices," Unpublished manuscript.

Hastings, W. K. (1970), "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, **57**, 97–109.

Heikkinen, J. and Hogmander, H. (1994), "Fully Bayesian approach to image restoration with an application in biogeography," *Applied Statistics*, **43**, 569–582.

Huang, F. and Ogata, Y. (2002), "Generalized pseudo-likelihood estimates for Markov random fields on lattice," *Annals of the Institute of Statistical Mathematics*, **54**, 1–18.

Ising, E. (1925), "Beitrag zur theorie des ferromagnetismus," *Zeitschrift für Physik*, **31**, 253–258.

Kass, R. E., Carlin, B. P., Gelman, A., and Neal, R. M. (1998), "Markov chain Monte Carlo in practice: a roundtable discussion," *The American Statistician*, **52**, 93–100.

Kindermann, R., Snell, J. L., et al. (1980), *Markov Random Fields and Their Applications*, vol. 1, American Mathematical Society Providence, RI.

Møller, J., Pettitt, A. N., Reeves, R., and Berthelsen, K. K. (2006), "An efficient Markov chain Monte Carlo method for distributions with intractable normalising constants," *Biometrika*, **93**, 451–458.

Murray, I., Ghahramani, Z., and MacKay, D. J. C. (2006), "MCMC for doubly-intractable distributions," in *Proceedings of the 22nd Annual Conference on Uncertainty in Artificial Intelligence (UAI-06)*, AUAI Press, 359–366.

Potts, R. B. (1952), "Some generalized order-disorder transformations," in *Proceedings of the Cambridge Philosophical Society*, Cambridge University Press, 106–109.

Propp, J. G. and Wilson, D. B. (1996), "Exact sampling with coupled Markov chains and applications to statistical mechanics," *Random Structures and Algorithms*, **9**, 223–252.

Reeves, R. and Pettitt, A. N. (2004), "Efficient recursions for general factorisable models," *Biometrika*, **91**, 751–757.

Tjelmeland, H. and Austad, H. M. (2012), "Exact and approximate recursive calculations for binary Markov random fields defined on graphs," *Journal of Computational and Graphical Statistics*, **21**, 758–780.