

Prior specification for binary Markov mesh models

XIN LUO and HÅKON TJELMELAND

Department of Mathematical Sciences, Norwegian University of Science and Technology

Abstract

We propose prior distributions for all parts of the specification of a Markov mesh model. In the formulation we define priors for the sequential neighborhood, for the parametric form of the conditional distributions and for the parameter values. By simulating from the resulting posterior distribution when conditioning on an observed scene, we thereby obtain an automatic model selection procedure for Markov mesh models. To sample from such a posterior distribution, we construct a reversible jump Markov chain Monte Carlo algorithm (RJMCMC). We demonstrate the usefulness of our prior formulation and the limitations of our RJMCMC algorithm in two examples.

Key words: Markov mesh model, prior construction, pseudo-Boolean functions, reversible jump MCMC, sequential neighborhood.

1 Introduction

Discrete Markov random fields (MRFs) defined on rectangular lattices are often used in spatial statistics, see for example Kindermann and Snell (1980) and Hurn et al. (2003). A discrete MRF is typically used to model available prior information about an unobserved scene x of a discrete variable. This is combined with a likelihood function describing the relation between x and some observed data y into a posterior distribution. The posterior is then the basis for making inference about x . When specifying the MRF prior, the most frequent approach is to fix the neighborhood and parametric model structures and also to specify the values of the model parameters a priori. However, several authors have also considered a more fully Bayesian approach by assigning a prior to the model specification of the MRF. The resulting posterior distribution then becomes doubly-intractable in that the normalizing constant of the MRF, which is computationally intractable, comes in as a factor in the posterior. Recently several strategies to cope with such doubly-intractable posterior distributions have been proposed, see for example Heikkinen and Högmänder (1994); Higdon et al. (1997); Friel et al. (2009); Everitt (2012); Cucula and Marin (2013); Friel (2013); Stoehr et al. (2015) and references therein. Coping with doubly-intractable posterior distributions computationally is, however, in general very computer intensive.

In this article we propose to circumvent the problem with doubly-intractable posterior distributions in the situation discussed above by replacing the MRF prior with a Markov mesh

prior. The class of Markov mesh models was introduced already in Abend et al. (1965) and was later generalized to partially ordered Markov models (POMMs) in Cressie and Davidson (1998). In contrast to the situation for MRFs, the normalizing constants of Markov mesh models and POMMs are analytically available and easy to compute. Thus, adopting a Markov mesh model as prior in a fully Bayesian setting results in a posterior that is easy to compute up to a normalizing constant. Simulation from the resulting posterior can thereby readily be done by the Metropolis–Hastings algorithm.

That the normalizing constants are easy to compute for Markov mesh models is a clear advantage over MRFs. However, Markov mesh models have also disadvantages relative to MRFs. Whereas it is easy to specify an isotropic MRF or to specify an MRF with a certain type of anisotropy, this is seemingly impossible for a Markov mesh model. When used as a prior with manually specified model structure and parameter values, an MRF is therefore clearly to be preferred over a Markov mesh model. Then the computationally intractable normalizing constant of the MRF constitutes no problem and we can specify isotropy or anisotropy as we want in the MRF. In a fully Bayesian model setup, however, the situation is reversed. Then the computationally intractable normalizing constant of the MRFs constitutes a major problem, and since we do not want to specify the model structure and parameter values of the underlying spatial field a priori it is not so important that we can not control the anisotropy in the Markov mesh model.

Our goal in the present article is to demonstrate how one can fit a Markov mesh model to an observed scene. To do this, we consider the scene as an observation and assume it to be a realization from a Markov mesh model. We put this into a Bayesian setting and define a flexible prior for both the model structure and the parameter values of the Markov mesh model, and generate realistic Markov mesh models for the observed scene by simulating from the resulting posterior distribution. Considering the fully Bayesian setup discussed above is clearly a natural next step, but beyond the scope of the present paper. In the present article we also limit the attention to binary Markov mesh models, but our approach can be generalized to a situation where each node can take more than two possible values.

The remainder of this article is organized as follows. In Section 2 we formulate a flexible class of Markov mesh models. In Section 3 we construct our prior distribution, and in Section 4 we formulate proposal distributions that we use in a reversible jump Markov chain Monte Carlo algorithm to simulate from the corresponding posterior when conditioning on an observed scene. In Section 5 we present two examples and lastly we give some closing remarks in Section 6.

2 Construction of a flexible Markov mesh model

In this section we formulate a flexible class of homogeneous binary Markov mesh models (Abend et al., 1965) on a rectangular lattice. We first introduce basic notation related to the lattice and the binary variables. Thereafter we define the Markov assumption used to define Markov mesh models and describe how we construct the sequential neighborhoods from a template sequential neighborhood. Finally, we describe how we use a template pseudo-Boolean function to define a parametric form for Markov mesh model.

2.1 Basic notation

We consider a rectangular $m \times n$ lattice and let $\chi = \{(i, j) : i = 1, \dots, m; j = 1, \dots, n\}$ denote the set of all nodes in the lattice, where i and j specify the vertical and horizontal positions of a node, respectively. We number the rows from $i = 1$ at the top of the lattice to $i = m$ at the bottom, and number the columns from $j = 1$ at the left end to $j = n$ at the right end. We also use $v = (i, j) \in \chi$ to denote an arbitrary node in the lattice. To each node $v \in \chi$ we associate a binary stochastic variable $x_v \in \{0, 1\}$. We use $x = (x_v; v \in \chi)$ to denote the collection of all these binary variables and for a set of nodes $\lambda \subseteq \chi$ we use $x_\lambda = (x_v; v \in \lambda)$ to denote the collection of all the binary variables associated to nodes in this set.

2.2 Markov assumption and template sequential neighborhood

The Markov mesh model is based on numbering the nodes in the lattice in the lexicographical order, from left-to-right and top-to-bottom, from one to mn . For a node $v \in \chi$, the predecessor set ρ_v is the set of all nodes coming before node v , i.e.

$$\rho_{(i,j)} = \{(k, l) \in \chi : nk + l < ni + j\}, \quad (1)$$

see the illustration in Figure 1(a). The Markov mesh model uses that the distribution of x can then be given as

$$f(x) = \prod_{v \in \chi} f(x_v | x_{\rho_v}). \quad (2)$$

The Markov mesh model adopts a Markov assumption in that the conditional distribution $f(x_v | x_{\rho_v})$ in fact only depends on a subset of the variables in x_{ρ_v} , see Figure 1(b) for an illustration. More precisely, for each $v \in \chi$ we assume we have a sequential neighborhood $\nu_v \subseteq \rho_v$ so that

$$f(x_v | x_{\rho_v}) = f(x_v | x_{\nu_v}). \quad (3)$$

It should be noted that the chosen numbering of the nodes and the assumed Markov property will for most scenes x not reflect the actual data generating process. The chosen numbering

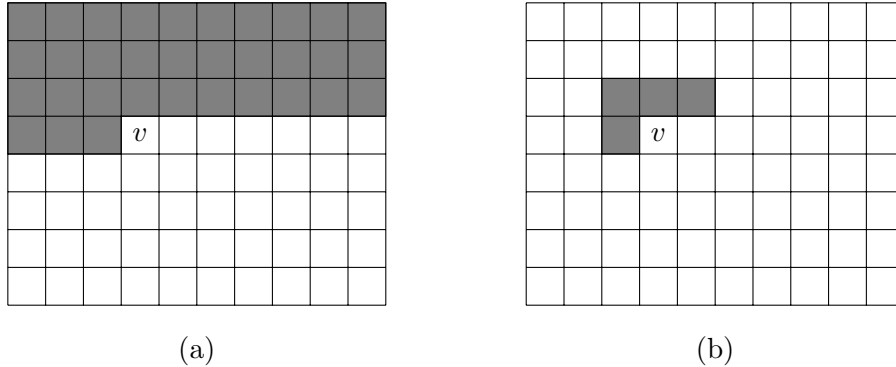


Figure 1: Illustration of the predecessor set ρ_v and a possible sequential neighborhood ν_v for node $v = (4, 4)$ in a 8×10 lattice. (a) The nodes in ρ_v are shown in gray. (b) The nodes in a possible sequential neighborhood $\nu_v = \{(4, 3), (3, 3), (3, 4), (3, 5)\}$ are shown in gray.

and the Markov property is just a pragmatic approach to obtain a joint distribution $f(x)$ for which we can easily compute the normalizing constant.

In this article we assume all the sequential neighborhoods are generated by a translation of a template sequential neighborhood τ . The τ can best be thought of as the sequential neighborhood of node $(0, 0)$ in an infinite lattice. More precisely, τ is required to contain a finite number of elements and

$$\tau \subset \psi = \{(i, j) : i \in \mathbb{Z}^-, j \in \mathbb{Z}\} \cup \{(0, j) : j \in \mathbb{Z}^-\}, \quad (4)$$

where ψ can be thought of as the predecessor set for $(0, 0)$ in an infinite lattice, and $\mathbb{Z} = \{0, \pm 1, \pm 2, \dots\}$ and $\mathbb{Z}^- = \{-1, -2, \dots\}$ are the set of all integers and the set of all negative integers, respectively. The sequential neighborhood for node $v \in \chi$ is then defined as

$$\nu_v = (\tau \oplus v) \cap \chi, \quad (5)$$

where the translation operator \oplus is defined as

$$\tau \oplus (i, j) = \{(k + i, l + j) : (k, l) \in \tau\}. \quad (6)$$

As illustrated in Figure 2, sequential neighborhoods for all nodes sufficiently far away from the lattice borders then have the same form, whereas nodes close to the borders have fewer sequential neighbors.

To model $f(x_v | x_{\nu_v})$ we then consider the logit transformation of the conditional probability for x_v being equal to one,

$$\text{logit}[f(x_v = 1 | x_{\nu_v})] = \ln \left(\frac{f(x_v = 1 | x_{\nu_v})}{1 - f(x_v = 1 | x_{\nu_v})} \right). \quad (7)$$

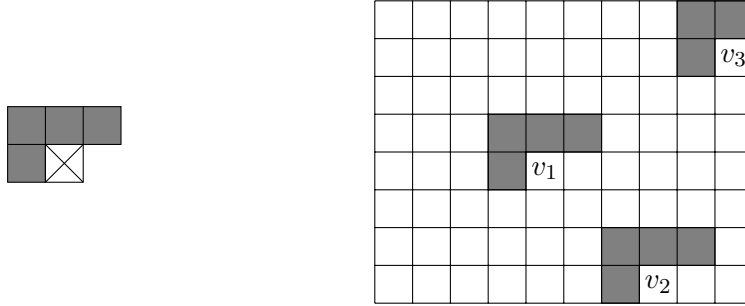


Figure 2: Illustration of the construction of sequential neighborhoods from a template τ . The left figure shows a possible template $\tau = \{(0, -1), (-1, -1), (-1, 0), (-1, 1)\}$, where the node $(0, 0)$ is represented with \boxtimes and the elements of τ are shown in gray. The right figure shows the resulting sequential neighborhoods (again in gray) for nodes $v_1 = (5, 5)$, $v_2 = (8, 8)$ and $v_3 = (2, 10)$ in a 8×10 lattice.

This is a real function of the binary variables in the vector x_{ν_v} and thereby a pseudo-Boolean function (Hammer and Holzman, 1992; Grabisch et al., 2000). To obtain simpler expressions when modeling this pseudo-Boolean function we choose to define it as a function of the set of nodes in ν_v for which the associated stochastic variable has the value one. This set of nodes we denote by

$$\xi(x, \nu_v) = \{u \in \nu_v : x_u = 1\} \quad (8)$$

and use $\theta_v(\cdot)$ to denote the pseudo-Boolean function, i.e.

$$\text{logit}[f(x_v = 1|x_{\nu_v})] = \theta_v(\xi(x, \nu_v)). \quad (9)$$

In the following we require the Markov mesh model to be homogeneous. One should note that the different pseudo-Boolean functions $\theta_v(\cdot)$, $v \in \chi$ are defined on different domains, so mathematically we can not just set them equal. To get a homogeneous model we define a template pseudo-Boolean function defined on τ ,

$$\theta : \Omega(\tau) \rightarrow \mathbb{R}, \quad (10)$$

where $\Omega(\tau)$ is the power set of τ , and relate all $\theta_v(\cdot)$, $v \in \chi$ to $\theta(\cdot)$ by the relation

$$\theta_v(\lambda) = \theta(\lambda \ominus v) \quad \text{for } \lambda \subseteq \nu_v, \quad (11)$$

where the translation operator \ominus is defined by

$$\lambda \ominus (i, j) = \{(k - i, l - j) : (k, l) \in \lambda\}. \quad (12)$$

Combining (7) with (9) and inserting the relations (5) and (11) we then get

$$f(x_v|x_{\nu_v}) = \frac{\exp\{x_v \cdot \theta(\xi(x, (\tau \oplus v) \cap \chi) \ominus v)\}}{1 + \exp\{\theta(\xi(x, (\tau \oplus v) \cap \chi) \ominus v)\}}. \quad (13)$$

Assuming, as we do, the Markov mesh model to be homogeneous is convenient in that we do not need to specify a separate pseudo-Boolean function for each node $v \in \chi$, and it is also statistically favorable as it limits the number of parameters in the model. However, one should note that this choice implies that for a node $v \in \chi$ close to the boundary of the lattice so that the set $(\tau \oplus v) \setminus \chi$ is non-empty, the conditional distribution $f(x_v | x_{\nu_v})$ is as if the nodes (for an infinite lattice) in the translation of τ that fall outside the lattice χ are all zero. Thus, even if the model is homogeneous it is not stationary, and in particular one should expect strong edge effects since we are essentially conditioning on everything outside the lattice χ to be zero. To reduce the boundary effects to a tolerable level we propose to include an unobserved boundary area around the observed scene. In the examples in Section 5 we adopt this strategy and one should note that the same strategy has previously been used to reduce the edge effects in Markov random fields.

2.3 Template pseudo-Boolean function

With the above construction the Markov mesh model is specified by the template sequential neighborhood τ and the template pseudo-Boolean function $\theta(\lambda), \lambda \subseteq \tau$. In this section we discuss the modeling of the latter.

Hammer and Rudeanu (1968) show that any pseudo-Boolean function can be uniquely represented by a collection of interaction parameters $(\beta(\lambda); \lambda \in \Omega(\tau))$ by the relation

$$\theta(\lambda) = \beta(\lambda) + \sum_{\lambda^* \subset \lambda} \beta(\lambda^*) \quad \text{for } \lambda \subseteq \tau. \quad (14)$$

The corresponding inverse relation is given by

$$\beta(\lambda) = \theta(\lambda) + \sum_{\lambda^* \subset \lambda} (-1)^{|\lambda \setminus \lambda^*|} \theta(\lambda^*) \quad \text{for } \lambda \subseteq \tau. \quad (15)$$

The one-to-one relation in (14) and (15) is known as Moebius inversion, see for example Lauritzen (1996).

To limit the number of parameters in the model we allow only a subset of the interaction parameter to differ from zero. More specifically, for some $\Lambda \subseteq \Omega(\tau)$ we restrict $\beta(\lambda) = 0$ for all $\lambda \notin \Lambda$. One can then represent the pseudo-Boolean function $\theta(\cdot)$ by the interaction parameters $\{\beta(\lambda), \lambda \in \Lambda\}$, and the relation in (14) becomes

$$\theta(\lambda) = \sum_{\lambda^* \in \Lambda \cap \Omega(\lambda)} \beta(\lambda^*) \quad \text{for } \lambda \in \Omega(\tau), \quad (16)$$

where $\Omega(\lambda)$ is the power set of λ . We then say that $\theta(\cdot)$ is represented on Λ . Moreover, we term $\lambda \in \Omega(\tau)$ an interaction and we say an interaction is active if $\lambda \in \Lambda$ and otherwise we say it is inactive.

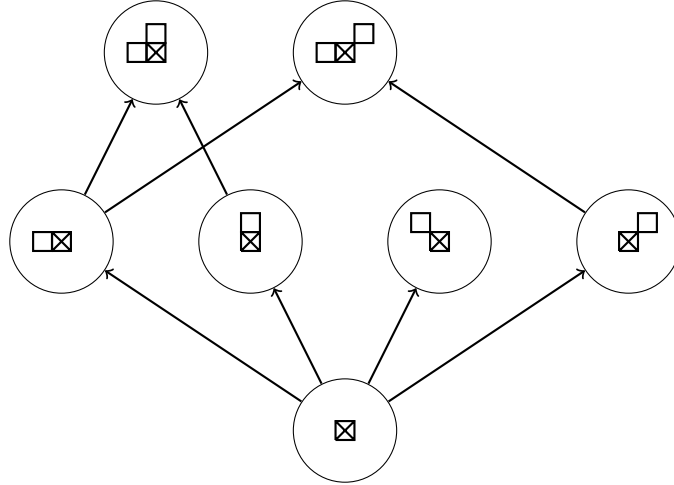


Figure 3: DAG visualization of the set $\Lambda = \{\emptyset, \{(0, -1)\}, \{(-1, 0)\}, \{(-1, -1)\}, \{(-1, 1)\}, \{(0, -1), (-1, 0)\}, \{(0, -1), (-1, 1)\}\}$ based on $\tau = \{(0, -1), (-1, -1), (-1, 0), (-1, 1)\}$. Thinking of the elements of τ as a finite set of nodes in a lattice, \boxtimes is used in the vertices of the DAG to represent the node $(0, 0)$, whereas each node $(i, j) \in \lambda$ for each $\lambda \in \Lambda$ is represented by a \square placed in position (i, j) relative to \boxtimes .

In the following we allow an interaction λ to be active only if all subsets of λ are active, i.e. we require $\lambda \in \Lambda \Rightarrow \Omega(\lambda) \subseteq \Lambda$. We then say the set Λ is dense and that we have a dense representation of $\theta(\cdot)$ on Λ . Moreover, in the following we require the template sequential neighborhood τ to be minimal for Λ in that all nodes $v \in \tau$ are included in at least one of the elements of Λ . One should note that if Λ is dense and τ is minimal for Λ then there is a one-to-one relation between the elements in τ and the sets $\lambda \in \Lambda$ which contains only one node,

$$\{\{v\} : v \in \tau\} = \{\lambda \in \Lambda : |\lambda| = 1\}. \quad (17)$$

As also discussed in Austad and Tjelmeland (2017), the set of active interactions Λ can be visualized by a directed acyclic graph (DAG), where we have one vertex for each active interaction $\lambda \in \Lambda$ and a vertex $\lambda \in \Lambda$ is a child of another vertex $\lambda^* \in \Lambda$ if and only if $\lambda = \lambda^* \cup \{v\}$ for some $v \in \tau \setminus \lambda^*$. Figure 3 shows such a DAG for $\Lambda = \{\emptyset, \{(0, -1)\}, \{(-1, 0)\}, \{(-1, -1)\}, \{(-1, 1)\}, \{(0, -1), (-1, 0)\}, \{(0, -1), (-1, 1)\}\}$, which is based on $\tau = \{(0, -1), (-1, -1), (-1, 0), (-1, 1)\}$. In the vertices of the DAG shown in the figure, node $(0, 0)$ is represented by the symbol \boxtimes , whereas each of the nodes in $\lambda \in \Lambda$ is represented by the symbol \square . Thinking of τ as a finite set of nodes in a lattice, the position of the \square representing node $(i, j) \in \lambda$ is placed at position (i, j) relative to \boxtimes .

As also discussed in Arnesen and Tjelmeland (2017), one should note that a pseudo-Boolean function $\theta(\cdot)$ that is represented on a dense set $\Lambda \subseteq \Omega(\tau)$ can be uniquely specified

by the values of $\{\theta(\lambda) : \lambda \in \Lambda\}$. The remaining values of the pseudo-Boolean function, $\theta(\lambda), \lambda \in \Omega(\tau) \setminus \Lambda$, are then given by (14) and (15) and the restriction $\beta(\lambda) = 0$ for $\lambda \notin \Lambda$. Moreover, as the relations in (14) and (15) are linear, each $\theta(\lambda), \lambda \in \Omega(\tau) \setminus \Lambda$ is a linear function of $\{\theta(\lambda) : \lambda \in \Lambda\}$.

Having defined our class of homogeneous Markov mesh models as above, a model is specified by the template sequential neighborhood τ , the set of active interactions $\Lambda \subseteq \Omega(\tau)$ on which the pseudo-Boolean function $\theta(\cdot)$ is represented, and the parameter values $\{\theta(\lambda) : \lambda \in \Lambda\}$. Thus, to adopt a fully Bayesian approach, we need to formulate prior distributions for τ , Λ and $\{\theta(\lambda) : \lambda \in \Lambda\}$, and this is the focus of the next section.

3 Prior distribution

When constructing our prior distribution for the template sequential neighborhood τ , the set of active interactions Λ and the parameter values $\{\theta(\lambda) : \lambda \in \Lambda\}$, we have two properties in mind. Firstly, the prior should be vague so that the Markov mesh model manages to adapt to a large variety of scenes. To obtain this, the number of elements in τ should be allowed to be reasonably large and higher-order interactions should be allowed in the model. Secondly, to avoid over-fitting, the prior should favor parsimonious Markov mesh models, and in particular this implies that the highest prior probabilities should be assigned to models with just a few higher-order interactions.

We define the prior as a product of three factors

$$f(\tau, \Lambda, \{\theta(\lambda) : \lambda \in \Lambda\}) = f(\tau)f(\Lambda|\tau)f(\{\theta(\lambda) : \lambda \in \Lambda\}|\tau, \Lambda), \quad (18)$$

where $f(\tau)$ is a prior for the template sequential neighborhood τ , $f(\Lambda|\tau)$ is a prior for the set of active interactions Λ when τ is given, and $f(\{\theta(\lambda) : \lambda \in \Lambda\}|\Lambda)$ is a prior for the parameter values given τ and Λ . In the following we discuss each of these factors in turn.

3.1 Prior for the template sequential neighborhood τ

We restrict the template sequential neighborhood to be a subset of a given finite set $\tau_0 \subset \psi$, where ψ is defined in (4). The τ_0 can be thought of as a set of possible sequential neighbors for node $(0, 0)$. To get a flexible prior it is important that the number of elements in τ_0 is not too small, and it is natural to let τ_0 include nodes close to $(0, 0)$. For example, one may let ψ include all nodes that are inside the circle centered at $(0, 0)$ with some specified radius r . In the examples discussed in Section 5 we use this with $r = 5$, see the illustration in Figure 4.

Given the set τ_0 we specify the prior for $\tau \subseteq \tau_0$ by first choosing a prior distribution for the number of elements in τ , and thereafter a prior for τ given the number of elements in τ .

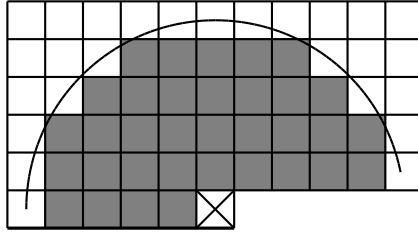


Figure 4: Illustration of the τ_0 used in the examples in Section 5. \boxtimes is node $(0, 0)$, and gray nodes are elements of τ_0 . The black curve is a part of the circle centered at $(0, 0)$ and with radius $r = 5$.

Letting $n_\tau = |\tau|$ denote the number of elements in τ we thereby have

$$f(\tau) = f(n_\tau)f(\tau|n_\tau). \tag{19}$$

For simplicity we choose both $f(n_\tau)$ and $f(\tau|n_\tau)$ to be uniform distributions. The possible values for n_τ are all integers from 0 to $|\tau_0|$, so we get

$$f(n_\tau) = \frac{1}{n_\tau + 1} \text{ for } n_\tau = 0, 1, \dots, |\tau_0|. \tag{20}$$

Moreover, having chosen τ to be uniform given $n_\tau = |\tau|$, we get

$$f(\tau|n_\tau) = \frac{1}{\binom{|\tau_0|}{n_\tau}}, \tag{21}$$

where the binomial coefficient in the denominator is the number of possible sets τ 's with n_τ elements.

One should note that our choice of the two uniforms above is very different from adopting a uniform prior for τ directly. A uniform prior on τ would have resulted in very high a priori probabilities for n_τ being close to $|\tau_0|/2$ and very small a priori probabilities for values of n_τ close to zero, which is clearly not desirable. One should also note that we do not require the sequential neighborhoods to be contiguous. In fact, in one of our two examples in Section 5 most of the a posteriori simulated models have sequential neighborhood that are not contiguous.

One can easily construct other reasonable priors for τ than the one defined above. For example, one could want to build into the prior $f(\tau|n_\tau)$ that nodes close to $(0, 0)$ are more likely to be in τ than nodes further away. Recalling that we want to simulate from a corresponding posterior distribution by a reversible jump Markov chain Monte Carlo algorithm (RJCMC) (Green, 1995), the challenge is to formulate a prior with this property so that we are able to compute the (normalized) probability $f(\tau|n_\tau)$, as this is needed to evaluate the Metropolis–Hastings acceptance probability. For the data sets discussed in Section 5,

we have also tried a prior $f(\tau|n_\tau)$ in which we split the nodes in τ_0 into two or three zones dependent on their distances from $(0, 0)$ and have a different prior probability for a node to be in τ dependent on which zone it is in. As long as the number of zones is reasonably small, it is then possible to compute the normalizing constant of $f(\tau|n_\tau)$ efficiently. However, in our examples this gave essentially the same posterior results as the very simple double uniform prior specified above.

3.2 Prior for the set of active interactions Λ

To specify a prior for the set of active interactions Λ , we first split Λ into several subsets dependent on how many nodes an element $\lambda \in \Lambda$ contains. More precisely, for $k = 0, 1, \dots, |\tau|$ we define

$$\Omega_k(\tau) = \{\lambda \in \Omega(\tau) : |\lambda| = k\} \quad \text{and} \quad \Lambda_k = \{\lambda \in \Lambda : |\lambda| = k\}. \quad (22)$$

Thus, $\Omega_k(\tau)$ contains all k 'th order interactions, and $\Lambda_k \subseteq \Omega_k(\tau)$ is the set of all k 'th order active interactions. As we have assumed τ to be minimal for Λ , τ is uniquely specifying $\Lambda_1 = \{\lambda \in \Lambda : |\lambda| = 1\}$, see the discussion in Section 2.3 and in particular (17). Moreover, we restrict \emptyset always to be active, i.e. $\emptyset \in \Lambda$ with probability one, which implies that we force the pseudo-Boolean function $\theta(\cdot)$ always to include a constant term. As we have already assumed Λ to be dense and τ to be minimal for Λ this is only an extra restriction when $\tau = \emptyset$. Thus, for given τ the sets Λ_0 and Λ_1 are known, so to formulate a prior for Λ we only need to define a prior for $\Lambda_k, k = 2, \dots, |\tau|$. We assume a Markov property for these sets in that

$$f(\Lambda|\tau) = \prod_{k=2}^{|\tau|} f(\Lambda_k|\Lambda_{k-1}). \quad (23)$$

Thus, to choose a prior $f(\Lambda|\tau)$ we only need to formulate $f(\Lambda_k|\Lambda_{k-1})$, and to do so we adopt the same strategy for all values of k . In the specification process of $f(\Lambda_k|\Lambda_{k-1})$ we should remember that we have already restricted Λ to be dense, so the chosen prior needs to be consistent with this. For a given Λ_{k-1} , an interaction $\lambda \in \Omega_k(\tau)$ can then be active only if all $k-1$ 'th order interactions $\lambda^* \in \Omega_{k-1}(\lambda)$ are active. We let Π_k denote this set of possible active k 'th order interactions, i.e. we must have

$$\Lambda_k \subseteq \Pi_k = \{\lambda \in \Omega_k(\tau) : \lambda^* \in \Lambda_{k-1} \text{ for all } \lambda^* \subset \lambda\}. \quad (24)$$

We assume each interaction $\lambda \in \Pi_k$ to be active with some probability p_k , independently of each other, and get

$$f(\Lambda_k|\Lambda_{k-1}) = p_k^{|\Lambda_k|} (1 - p_k)^{|\Pi_k| - |\Lambda_k|} \quad \text{for} \quad \Lambda_k \subseteq \Pi_k. \quad (25)$$

One should note that if $\Lambda_{k-1} = \emptyset$ one gets $\Pi_k = \emptyset$ and thereby also $f(\Lambda_k = \emptyset|\Lambda_{k-1}) = 1$.

The probabilities $p_k, k = 2, \dots, |\tau|$ should be chosen to get a reasonable number of higher-order active interactions. To obtain a parsimonious model, one need to adopt a small value for p_k if the number of elements in Π_k is large, but to favor a model to include some higher-order interactions, the value of p_k can be large when the number of elements in Π_k is small. We choose

$$p_k = \begin{cases} p^* & \text{if } |\Pi_k| \leq |\Lambda_{k-1}|, \\ p^* \cdot \frac{|\Lambda_{k-1}|}{|\Pi_k|} & \text{otherwise,} \end{cases} \quad (26)$$

where $p^* \in (0, 1)$ is a hyper-parameter to be specified. One should note that this choice in particular ensures the expected number of active k 'th order interactions to be smaller than $|\Lambda_{k-1}|$.

3.3 Prior for the parameter values $\{\theta(\lambda) : \lambda \in \Lambda\}$

Given τ and the set of active interactions Λ , the set of model parameters for which we need to formulate a prior is $\{\theta(\lambda) : \lambda \in \Lambda\}$. From (13) we have that each $\theta(\lambda), \lambda \in \Lambda$ has a one-to-one correspondence with the conditional probability

$$p(\lambda) = f(x_v = 1 | x_{\rho_v}) = \frac{\exp\{\theta(\lambda)\}}{1 + \exp\{\theta(\lambda)\}} \quad \text{for } \lambda = \xi(x, \nu_v). \quad (27)$$

Since the $\theta(\lambda)$'s define probabilities conditioning on different values for x_{ρ_v} , we find it reasonable, unless particular prior information is available and suggests otherwise, to assume the $\theta(\lambda), \lambda \in \Lambda$ to be independent. In the following we adopt this independence assumption. Moreover, as we do not have a particular class of scenes in mind but want the prior to be reasonable for a wide variety of scenes, we adopt the same prior density for all parameters $\theta(\lambda), \lambda \in \Lambda$.

To formulate a reasonable and vague prior for $\theta(\lambda)$, we use the one-to-one correspondence between $\theta(\lambda)$ and the probability $p(\lambda)$. The interpretation for $p(\lambda)$ is much simpler than that of $\theta(\lambda)$, so our strategy is first to choose a prior for $p(\lambda)$ and from this derive the corresponding prior for $\theta(\lambda)$. As we do not have a particular class of scenes in mind but want our prior to be reasonable for a wide variety of scenes, we find it most natural to adopt a uniform prior on $[0, 1]$ for $p(\lambda)$. However, as previously mentioned we want to explore a corresponding posterior distribution by running a reversible jump Metropolis–Hastings algorithm, and in particular we want to use adaptive rejection sampling (Gilks, 1992) to update $\theta(\lambda)$. For this to work, the full conditional for $\theta(\lambda)$ needs to be log-concave. Adopting the uniform on $[0, 1]$ prior for $p(\lambda)$ the resulting posterior full conditional becomes log-concave, but the second derivative of the log full conditional converges to zero when $\theta(\lambda)$ goes to plus or minus infinity. As this may generate numerical problems when running the adaptive rejection sampling algorithm, we adopt a prior for $p(\lambda)$ slightly modified relative to the uniform and obtain a posterior

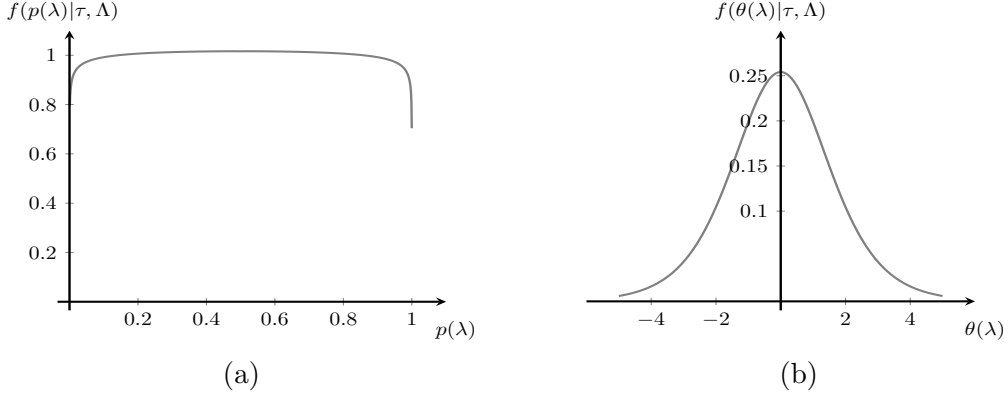


Figure 5: The prior distributions for $p(\lambda)$ and $\theta(\lambda)$. (a) The density curve of $f(p(\lambda)|\tau, \Lambda)$ when $\sigma = 10$, and (b) the corresponding density curve $f(\theta|\tau, \Lambda)$ given in (28).

distribution where the second derivative of the log full conditional for $\theta(\lambda)$ converges to a value strictly less than zero. More precisely, we adopt the following prior for $\theta(\lambda)$,

$$f(\theta(\lambda)|\tau, \Lambda) \propto \frac{e^{\theta(\lambda)}}{(1 + e^{\theta(\lambda)})^2} \cdot e^{-\frac{\theta(\lambda)^2}{2\sigma^2}}, \quad (28)$$

where the first factor is the prior resulting from assuming $p(\lambda)$ to be uniform, the second factor is the modification we adopt to avoid numerical problems when running the adaptive rejection sampling algorithm, and $\sigma > 0$ is a hyper-parameter to be specified. The resulting priors for $p(\lambda)$ and $\theta(\lambda)$ when $\sigma = 10$ are shown in Figure 5. We see that the prior for $p(\lambda)$ is close to the uniform. One can also note that $f(\theta(\lambda)|\Lambda)$ have heavier tails than a normal distribution with the same variance. One should note that the normalizing constant in (28) is required when updating Λ in a reversible jump Metropolis–Hastings algorithm targeting a corresponding posterior distribution, but since (28) is a univariate distribution this normalizing constant can easily be found by numerical integration. Letting $c(\sigma)$ denote the normalizing constant of $f(\theta(\lambda)|\tau, \Lambda)$ the complete expression for the prior for $\{\theta(\lambda) : \lambda \in \Lambda\}$ is

$$f(\{\theta(\lambda) : \lambda \in \Lambda\} | \tau, \Lambda) = \prod_{\lambda \in \Lambda} \left[c(\sigma) \cdot \frac{e^{\theta(\lambda)}}{(1 + e^{\theta(\lambda)})^2} \cdot e^{-\frac{\theta(\lambda)^2}{2\sigma^2}} \right]. \quad (29)$$

Having specified priors for τ , Λ and $\{\theta(\lambda) : \lambda \in \Lambda\}$ we formulate in the next section a reversible jump Metropolis–Hastings algorithm for simulating from the corresponding posterior when a scene x is observed.

4 Simulation algorithm

In this section we assume we have observed a complete scene $x = (x_v; v \in \chi)$ and assume this to be a realization from the Markov mesh model defined in Section 2. We adopt the prior

defined in Section 3 and want to explore the resulting posterior distribution

$$f(\tau, \Lambda, \{\theta(\lambda) : \lambda \in \Lambda\} | x) \propto f(\tau, \Lambda, \{\theta(\lambda) : \lambda \in \Lambda\}) f(x | \tau, \Lambda, \{\theta(\lambda) : \lambda \in \Lambda\}), \quad (30)$$

by a reversible jump Markov chain Monte Carlo algorithm, see Green (1995). We combine two types of updates. In the first update class, we keep τ and Λ unchanged and update the parameter vector $\{\theta(\lambda) : \lambda \in \Lambda\}$ by a Gibbs step along a direction sampled uniformly at random. In the second update class, we propose a trans-dimensional move by adding an inactive interaction to Λ or removing an active interaction from Λ , and proposing corresponding changes for the parameter vector $\{\theta(\lambda) : \lambda \in \Lambda\}$.

It is clearly of interest to consider also the resulting posterior distribution when parts of the scene x is unobserved or when x is an unobserved latent field. The former is of interest if one wants to reduce the boundary effects of the Markov mesh model by letting x include an unobserved boundary around the observed area, and the latter is a common situation in image analysis applications. However, to simplify the discussion of the simulation algorithm in this section, we assume the complete scene x to be observed. In Section 5, where we present two examples, we describe how to adapt the simulation algorithm to situation in which a part of x is unobserved.

In the following we describe each of the two update types in turn, starting with the Gibbs update for the parameter values. We only discuss the proposal distribution, as the acceptance probabilities is then given by standard formulas.

4.1 Gibbs update for the parameter values $\{\theta(\lambda) : \lambda \in \Lambda\}$

Let τ , Λ and $\{\theta(\lambda) : \lambda \in \Lambda\}$ be the current state. In this update, we keep τ and Λ unchanged and generate new parameter values $\{\theta^*(\lambda) : \lambda \in \Lambda\}$. To generate the new parameter values we first draw a random direction $\{\Delta(\lambda) : \lambda \in \Lambda\}$ by sampling $\Delta(\lambda)$ from a standard normal distribution, independently for each $\lambda \in \Lambda$. We then set

$$\theta^*(\lambda) = \theta(\lambda) + \alpha \Delta(\lambda), \quad (31)$$

where $\alpha \in \mathbb{R}$ is sampled from the full conditional

$$f(\alpha | \tau, \Lambda, \{\theta(\lambda) + \alpha \Delta(\lambda) : \lambda \in \Lambda\}, x) \propto f(\{\theta(\lambda) + \alpha \Delta(\lambda) : \lambda \in \Lambda\} | \tau, \Lambda) \cdot f(x | \tau, \Lambda, \{\theta(\lambda) + \alpha \Delta(\lambda) : \lambda \in \Lambda\}). \quad (32)$$

As α is sampled from its full conditional, this is a Gibbs update and the Metropolis–Hastings acceptance probability is one. The full conditional (32) for α is not of a standard form, but in Appendix A we show that it is log-concave, so to generate samples from it we adopt the adaptive rejection sampling algorithm of Gilks (1992).

4.2 Updating the set of active interactions

Let again τ , Λ and $\{\theta(\lambda) : \lambda \in \Lambda\}$ be the current state. In this update we modify Λ , and possibly also τ , by adding an inactive interaction to Λ or by removing an active interaction from Λ . We let τ^* and Λ^* denote the potential new values for τ and Λ , respectively. With a change in Λ , the number of parameter values $\{\theta(\lambda) : \lambda \in \Lambda\}$ is also changed, and to try to obtain a high acceptance rate, we in fact propose a change also in some of the parameter values that are in both the current and potential new states. We let $\{\theta^*(\lambda) : \lambda \in \Lambda^*\}$ denote the set of potential parameter values.

To generate τ^* , Λ^* and $\{\theta^*(\lambda) : \lambda \in \Lambda^*\}$, we first draw at random whether to add an inactive interaction to Λ or to remove an active interaction from Λ . In the following we specify in turn our procedures for proposing to remove and add an interaction.

4.2.1 Proposing to remove an active interaction from Λ

Having decided that an interaction should be removed, the next step is to decide what interaction $\lambda^* \in \Lambda$ to remove. As the potential new $\Lambda^* = \Lambda \setminus \{\lambda^*\}$ should be dense, we first find the set of active interactions λ^* that fulfills this requirement,

$$\Lambda_r = \{\lambda \in \Lambda \setminus \{\emptyset\} : \Lambda \setminus \{\lambda\} \text{ is dense}\}. \quad (33)$$

Thereafter we draw what interaction $\lambda^* \in \Lambda_r$ to be removed, with probabilities

$$q(\lambda^*) = \frac{\exp\{-\nu d(\lambda^*, \tau, \Lambda, \{\theta(\lambda) : \lambda \in \Lambda\})\}}{\sum_{\tilde{\lambda} \in \Lambda_r} \exp\{-\nu d(\tilde{\lambda}, \tau, \Lambda, \{\theta(\lambda) : \lambda \in \Lambda\})\}} \quad \text{for } \lambda^* \in \Lambda_r, \quad (34)$$

where $\nu \geq 0$ is an algorithmic tuning parameter to be specified, and $d(\lambda^*, \tau, \Lambda, \{\theta(\lambda) : \lambda \in \Lambda\})$ is a function that should measure the difference between the current pseudo-Boolean function defined by τ , Λ and $\{\theta(\lambda) : \lambda \in \Lambda\}$ and the potential new pseudo-Boolean function defined by τ^* , Λ^* and $\{\theta^*(\lambda) : \lambda \in \Lambda^*\}$. We specify below the precise formula used for $d(\lambda^*, \tau, \Lambda, \{\theta(\lambda) : \lambda \in \Lambda\})$, after having specified how to set the potential new parameter values $\{\theta^*(\lambda) : \lambda \in \Lambda^*\}$. By setting the algorithmic tuning parameter $\nu = 0$, we draw Λ^* uniformly at random from the elements in Λ_r . With a larger value for ν , we get a higher probability for proposing to remove an interaction λ^* that gives a small change in the pseudo-Boolean function. If it should happen that $\Lambda_r = \emptyset$, we simply propose an unchanged state. Assuming we have sampled a λ^* to remove, we have two possibilities. If λ^* is a higher-order interaction the sequential neighborhood is unchanged, i.e. $\tau^* = \tau$, whereas if λ^* is a first-order interaction the sequential neighborhood is reduced to $\tau^* = \tau \setminus \lambda^*$.

Having decided τ^* and Λ^* , the next step is to specify the potential new parameter values $\{\theta^*(\lambda) : \lambda \in \Lambda^*\}$. To understand our procedure for doing this, one should remember that there is a one-to-one relation between the current parameter values $\{\theta(\lambda) : \lambda \in \Lambda\}$ and a

set of current interaction parameters $\{\beta(\lambda) : \lambda \in \Lambda\}$, where the relation is given by (14) and (15). Moreover, together with the restriction $\beta(\lambda) = 0$ for $\lambda \notin \Lambda$, this defines a pseudo-Boolean function $\{\theta(\lambda) : \lambda \in \Omega(\tau_0)\}$. Correspondingly, there is a one-to-one relation between the potential new parameter values $\{\theta^*(\lambda) : \lambda \in \Lambda\}$ and a set of potential new interaction parameters $\{\beta^*(\lambda) : \lambda \in \Lambda^*\}$, and together with the restrictions $\beta^*(\lambda) = 0$ for $\lambda \notin \Lambda^*$ this defines a potential new pseudo-Boolean function $\{\theta^*(\lambda) : \lambda \in \Omega(\tau_0)\}$. To get a high acceptance probability for the proposed change, it is reasonable to choose the potential new parameter values $\{\theta^*(\lambda) : \lambda \in \Lambda^*\}$ so that the difference between the two pseudo-Boolean functions $\{\theta(\lambda) : \lambda \in \Omega(\tau_0)\}$ and $\{\theta^*(\lambda) : \lambda \in \Omega(\tau_0)\}$ is small. One may consider the potential new pseudo-Boolean function $\{\theta^*(\lambda) : \lambda \in \Omega(\tau_0)\}$ as an approximation to the current $\{\theta(\lambda) : \lambda \in \Omega(\tau_0)\}$ and, adopting a minimum sum of squares criterion, minimize

$$\text{SSE}(\{\theta^*(\lambda) : \lambda \in \Lambda^*\}) = \sum_{\lambda \in \Omega(\tau_0)} (\theta^*(\lambda) - \theta(\lambda))^2 \quad (35)$$

with respect to $\{\theta^*(\lambda) : \lambda \in \Omega(\tau_0)\}$. Grabisch et al. (2000) solved this minimization problem. Expressed in terms of the corresponding interaction parameters $\{\beta(\lambda) : \lambda \in \Lambda\}$, the optimal potential new parameter values are

$$\beta^*(\lambda) = \begin{cases} \beta(\lambda) - (-\frac{1}{2})^{|\Lambda^*| - |\lambda|} \beta(\lambda^*) & \text{if } \lambda \subset \lambda^*, \\ \beta(\lambda) & \text{otherwise,} \end{cases} \quad (36)$$

and the obtained minimum sum of squares is

$$\min \{\text{SSE}(\{\theta^*(\lambda) : \lambda \in \Lambda\})\} = \frac{\beta(\lambda^*)}{2^{|\Lambda^*|}}. \quad (37)$$

We use the latter to define the function $d(\lambda^*, \tau, \Lambda, \{\theta(\lambda) : \lambda \in \Lambda\})$, used in (34) to define the distribution for what interaction λ^* to remove. We simply set

$$d(\lambda^*, \tau, \Lambda, \{\theta(\lambda) : \lambda \in \Lambda\}) = \frac{\beta(\lambda^*)}{2^{|\Lambda^*|}}. \quad (38)$$

Combining the expression in (36) with the one-to-one relations in (14) and (15), one can find the potential new parameters $\{\theta^*(\lambda) : \lambda \in \Lambda^*\}$ in terms of the current parameters $\{\theta(\lambda) : \lambda \in \Lambda\}$. In particular, we see that this relation is linear and we have a $|\Lambda| \times |\Lambda|$ matrix A so that

$$\begin{bmatrix} \theta^* \\ \beta(\lambda^*) \end{bmatrix} = A\theta \quad \Leftrightarrow \quad \theta = A^{-1} \begin{bmatrix} \theta^* \\ \beta(\lambda^*) \end{bmatrix}, \quad (39)$$

where $\theta^* = (\theta^*(\lambda) : \lambda \in \Lambda^*)^T$ and $\theta = (\theta(\lambda) : \lambda \in \Lambda)^T$ are column vectors of the potential new and current parameter values, respectively. As the number of elements in θ^* is one less than the number of elements in θ , we use $\beta(\lambda^*)$ to obtain the one-to-one relation we need for a reversible jump proposal. The Jacobian determinant in the expression for the corresponding acceptance probability is clearly $\det(A)$, and in Appendix B we show that the absolute value of this determinant is always equal to one, i.e. $|\det(A)| = 1$.

4.2.2 Proposing to add an inactive interaction to Λ

If it is decided that an inactive interaction should be added to Λ , the next step is to decide what interaction $\lambda^* \in \Omega(\tau_0) \setminus \Lambda$ to add. We do this in two steps, first we draw at random whether a first-order or a higher-order interaction should be added to Λ . If a first-order interaction should be added, we draw uniformly at random a node v^* from $\tau_0 \setminus \tau$ and set $\lambda^* = \{v^*\}$. Then $\tau^* = \tau \cup \lambda^*$ and $\Lambda^* = \Lambda \cup \{\lambda^*\}$. If $\tau = \tau_0$, so that no such v^* exists, we simply propose an unchanged state. If a higher-order interaction should be added we need to ensure that $\Lambda \cup \{\lambda^*\}$ is dense. We therefore first find

$$\Lambda_a = \{\lambda \in \Omega(\tau_0) \setminus \Lambda : |\lambda| > 1 \text{ and } \Lambda \cup \{\lambda\} \text{ is dense}\} \quad (40)$$

and thereafter draw λ^* uniformly at random from Λ_a . Then $\tau^* = \tau$ and $\Lambda^* = \Lambda \cup \{\lambda^*\}$. If it should happen that $\Lambda_a = \emptyset$, we again simply propose an unchanged state.

Having decided τ^* and Λ^* , the next step is to generate the potential new parameter values $\{\theta^*(\lambda) : \lambda \in \Lambda^*\}$. When doing this, one should remember that this adding a potential new interaction proposal must be one-to-one with the reverse removing an interaction proposal discussed in Section 4.2.1. Therefore, the proposal distribution for the potential new parameter values $\{\theta^*(\lambda) : \lambda \in \Lambda^*\}$ must conform with (36), and thereby also with (39). A natural way to achieve this is to draw a value $\beta^*(\lambda^*)$ from some distribution and define the potential new interaction parameters by the inverse transformation of (36), i.e.

$$\beta^*(\lambda) = \begin{cases} \beta(\lambda) + \left(-\frac{1}{2}\right)^{|\lambda^*| - |\lambda|} \beta^*(\lambda^*) & \text{if } \lambda \subset \lambda^*, \\ \beta(\lambda) & \text{otherwise.} \end{cases} \quad (41)$$

It now just remains to specify from what distribution to sample $\beta^*(\lambda^*)$. The potential new parameter values $\{\theta^*(\lambda) : \lambda \in \Lambda^*\}$ are linear functions of $\beta^*(\lambda^*)$, and by setting $\beta^*(\lambda^*) = \alpha$ it can be expressed as in (31) for the Gibbs update. The difference between what we now have to do and what is done in the Gibbs update is that in the Gibbs update the values $\Delta(\lambda)$ are sampled independently from a Gaussian distribution, whereas here these are implicitly defined by (41) together with the one-to-one relations (14) and (15). It is tempting to sample $\alpha = \beta^*(\lambda^*)$ from the resulting full conditional, as this would give a high density for values of $\beta^*(\lambda^*)$ that corresponds to models with a high posterior probability. As discussed in Section 4.1 for the Gibbs update, it is computationally feasible to sample from this full conditional by adaptive rejection sampling. However, the normalizing constant of this full conditional is not computationally available, and for computing the associated acceptance probability the normalizing constant of the distribution of $\beta^*(\lambda^*)$ must be available. To construct a proposal distribution for $\beta^*(\lambda^*) = \alpha$, we therefore instead first generate r (say) independent samples $\alpha_1, \dots, \alpha_r$ from the full conditional for α , by adaptive rejection sampling, and thereafter draw $\alpha = \beta^*(\lambda^*)$ from a Gaussian distribution with mean value $\bar{\alpha} = \frac{1}{r} \sum_{i=1}^r \alpha_i$ and variance

$s_\alpha^2 = \frac{1}{r-1} \sum_{i=1}^r (\alpha_i - \bar{\alpha})^2$. Our proposal distribution for $\beta^*(\lambda^*)$ is thereby an approximation to its full conditional.

As this is a reversible jump proposal, the associated acceptance probability includes a Jacobian determinant. By construction the Jacobian determinant for this proposal is the inverse of the Jacobian determinant for the removing an interaction proposal discussed in Section 4.2.1. As we have $|\det(A)| = 1$, we also get $|\det(A^{-1})| = 1$.

5 Examples

In this section we investigate our prior and proposal distributions on two binary example scenes. As discussed also in the introduction our goal is not to analyze these two data sets, but just to demonstrate that with the Bayesian setup defined above we are able to find Markov mesh models that are reasonable models for the observed scenes.

The first scene we consider is a mortality map for liver and gallbladder cancers for white males from 1950 to 1959 in the eastern United States, compiled by Riggan et al. (1987). Using Markov random field models, this data set has previously been analyzed by Sherman et al. (2006), Liang (2010) and Austad and Tjelmeland (2017), see also Liang et al. (2011). Secondly, we consider a data set previously considered by Stien and Kolbjørnsen (2011). They also fitted a Markov mesh model to this data set, but with manually chosen neighborhood and interaction structures. In the following we first discuss some general aspects relevant for both the two examples and thereafter present details of each of the two examples in turn.

As also briefly discussed in Section 4, we reduce the boundary effects of the Markov mesh model by letting x include an unobserved boundary around the observed area. We choose the unobserved boundary large enough so that each of the observed nodes are at least 20 nodes away from the extended lattice boundary. We let χ denote the set of nodes in the extended lattice and let $x = (x_v; v \in \chi)$ be the corresponding collection of binary variables. We assume x to be distributed according to the Markov mesh model defined in Section 2, and for τ , Λ and $\{\theta(\lambda) : \lambda \in \Lambda\}$ we adopt the prior specified in Section 3. We let $\chi_o \subset \chi$ denote the set of nodes for which we have observed values. Thereby $\chi_u = \chi \setminus \chi_o$ is the set of unobserved nodes. Correspondingly, we let $x_o = (x_v, v \in \chi_o)$ be the observed values and $x_u = (x_v, v \in \chi_u)$ the unobserved values. The posterior distribution of interest is thereby $f(\tau, \Lambda, \{\theta(\lambda), \lambda \in \Lambda\} | x_o)$. To simplify the posterior simulation, we include x_u as auxiliary variables and adopt the reversible jump Metropolis–Hastings algorithm to simulate from

$$f(\tau, \Lambda, \{\theta(\lambda), \lambda \in \Lambda\}, x_u | x_o) \propto f(\tau, \Lambda, \{\theta(\lambda), \lambda \in \Lambda\}) f(x_o, x_u | \tau, \Lambda, \{\theta(\lambda), \lambda \in \Lambda\}). \quad (42)$$

To simulate from this distribution, we adopt the updates discussed in Section 4 to update τ , Λ and $\{\theta(\lambda), \lambda \in \Lambda\}$ conditioned on $x = (x_o, x_u)$, and we use single-site Gibbs updates for each unobserved node $v \in \chi_u$ given τ , Λ , $\{\theta(\lambda), \lambda \in \Lambda\}$ and $x_{\chi \setminus \{v\}}$. We define one iteration of the

algorithm to include $|\chi_u|$ single-site Gibbs updates for randomly chosen nodes in χ_u followed by either one Gibbs update of the parameter values $\{\theta(\lambda), \lambda \in \Lambda\}$ as discussed in Section 4.1 or one update of the active interactions as discussed in Section 4.2. In each iteration we independently update the parameter values or the active interactions with probabilities 0.55 and 0.45 respectively.

The prior defined in Section 3 contains three hyper-parameters, the radius r which defines the set of possible neighbors, the probability p^* in (26), and the parameter σ in (28). The radius r should be chosen large enough to allow interactions between nodes at some distance from each other. If setting the value of r too large, however, we get a more difficult posterior distribution to sample from. In our two examples we have found $r = 5$ to be a reasonable trade-off. This gives the 34 possible neighbors shown in Figure 4. To get a prior where the probability for a Markov mesh model with higher-order interactions is reasonably high, we set the value of p^* as high as 0.9, and to get an essentially uniform prior distribution for $p(\lambda)$, we set $\sigma = 100$. The proposal distribution discussed in Section 4.2 has one algorithmic tuning parameter, ν , and based on simulation results in preliminary runs we set $\nu = 0.5$.

We have implemented the RJMCMC algorithm in C++ (Online Resource 1) and in the following we present the example scene and discuss corresponding simulation results for each of our two examples. We start with the cancer mortality map compiled by Riggan et al. (1987).

5.1 Cancer mortality map

The cancer mortality map data are shown in Figure 6(a), where black ($x_v = 1$) and white ($x_v = 0$) pixels represent counties with high and low cancer mortality rates, respectively. The gray area around the observed map represents unobserved nodes which we included in the model to reduce the boundary effects of the Markov mesh model.

Adopting the Markov mesh and prior models discussed in Sections 2 and 3, respectively, with the hyper-parameters defined above, we use the RJMCMC setup discussed above to explore the resulting posterior distribution. We run the Markov chain for 2 500 000 iterations, and study trace plots of different scalar quantities to evaluate the convergence and mixing properties of the simulated Markov chain. Figure 7 shows trace plots of the first 25 000 iterations for the number of interactions and for the logarithm of the posterior density. From these two and the other trace plots we have studied, we conclude that the simulated chain has converged at least within the first 10 000 – 15 000 iterations. As an extra precaution we discard the first 25 000 iterations when estimating posterior properties.

To study the posterior distribution we first estimate, for each of the 34 a priori potential neighbors in τ_0 , the posterior probability for $v \in \tau_0$ to be a neighbor. To estimate this we simply use the fraction of simulated models where v is in the template sequential neighborhood

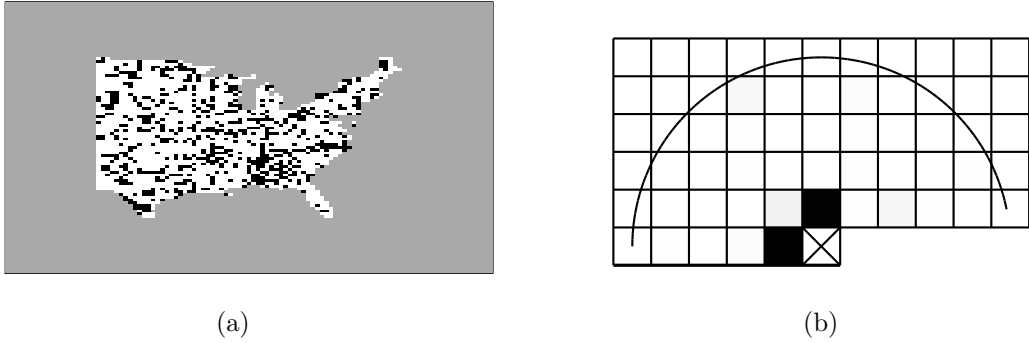


Figure 6: Cancer mortality map example: (a) Observed cancer mortality map. Black and white nodes represent counties with high and low cancer mortality rates, respectively. The nodes added to the lattice to reduce the boundary effects of the Markov mesh model is shown in gray. (b) Map of estimated a posteriori marginal probabilities for each node $v \in \tau_0$ to be a neighbor. A gray-scale is used to visualize the probabilities, where black and white represents one and zero, respectively.

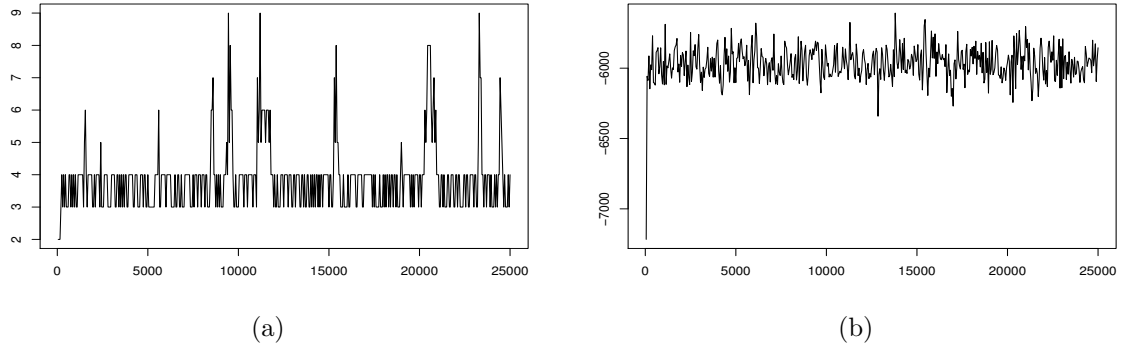












Figure 7: Cancer mortality map example: Trace plots of the first 25 000 iterations of the RJMCMC run. (a) Number of interactions $|\Lambda|$, (b) logarithm of the posterior density $\log [f(\tau, \Lambda, \{\theta(\lambda) : \lambda \in \Lambda\}, x_u | x_o)]$

Table 1: Cancer mortality map example: Table with the top 10 a posteriori most likely interactions and their estimated posterior probabilities.

Interaction					
Probability	1.0000	0.9998	0.9902	0.5452	0.0489
Interaction					
Probability	0.0303	0.0280	0.0274	0.0270	0.0251

τ . The result is shown in Figure 6(b), where we use a grayscale to visualize the probabilities. Nodes $(0, -1)$ and $(-1, 0)$ have high estimated posterior probabilities, equal to 0.999819 and 0.990577, respectively. The third and fourth most probable neighbor nodes are $(-1, -1)$ and $(-1, 2)$, where the estimated probabilities are 0.049388 and 0.030353, respectively. From the data set shown in Figure 6(a), we see that the dependence between neighbor nodes seems to be quite weak, so the low number of simulated neighbors should come as no surprise.

Next we correspondingly estimate the posterior probabilities for each possible interaction to be included in the model. Table 1 shows the top 10 a posteriori most likely interactions and the corresponding estimated probabilities. We see that the first four interactions have high posterior probabilities while the others have low probabilities. In addition, the four most likely interactions only include the high probability neighbor nodes $(0, -1)$ and $(-1, 0)$.

We also estimate the a posteriori marginal distributions for the parameter values $\theta(\cdot)$ corresponding to the four high probable interactions. Note that some of the interactions do not exist in some of the simulated models, but the $\theta(\cdot)$ value is still well defined and can be computed as discussed in Section 2.3. Figure 8 depicts the histograms of the simulated parameter values $\theta(\cdot)$. From the simulation we also estimate the posterior probability for each of the possible models. The two most probable models are shown in Figure 9. These two models have posterior probabilities as high as 0.475 and 0.381 while the remaining probability mass is spread out on a very large number of models.

Finally, we generate realizations from simulated Markov mesh models. Figure 10 contains realizations simulated from four randomly chosen models simulated in the Markov chain (after the specified burn-in). As in Figure 6(a), showing the observed data set, black and white nodes v represent $x_v = 1$ and 0, respectively. Comparing the realizations with the data set in Figure 6(a), we can get a visual impression of to what degree the simulated models have captured the dependence structure in the data set. Table 2 shows the estimated posterior correlation structure based on a large number of realizations when the nodes are numbered in the lexicographical order (left) and when they instead are numbered right-to-left and top-to-bottom (right). In the lags $(\pm 1, \pm 1)$ we can see a clear effect from the node ordering, but

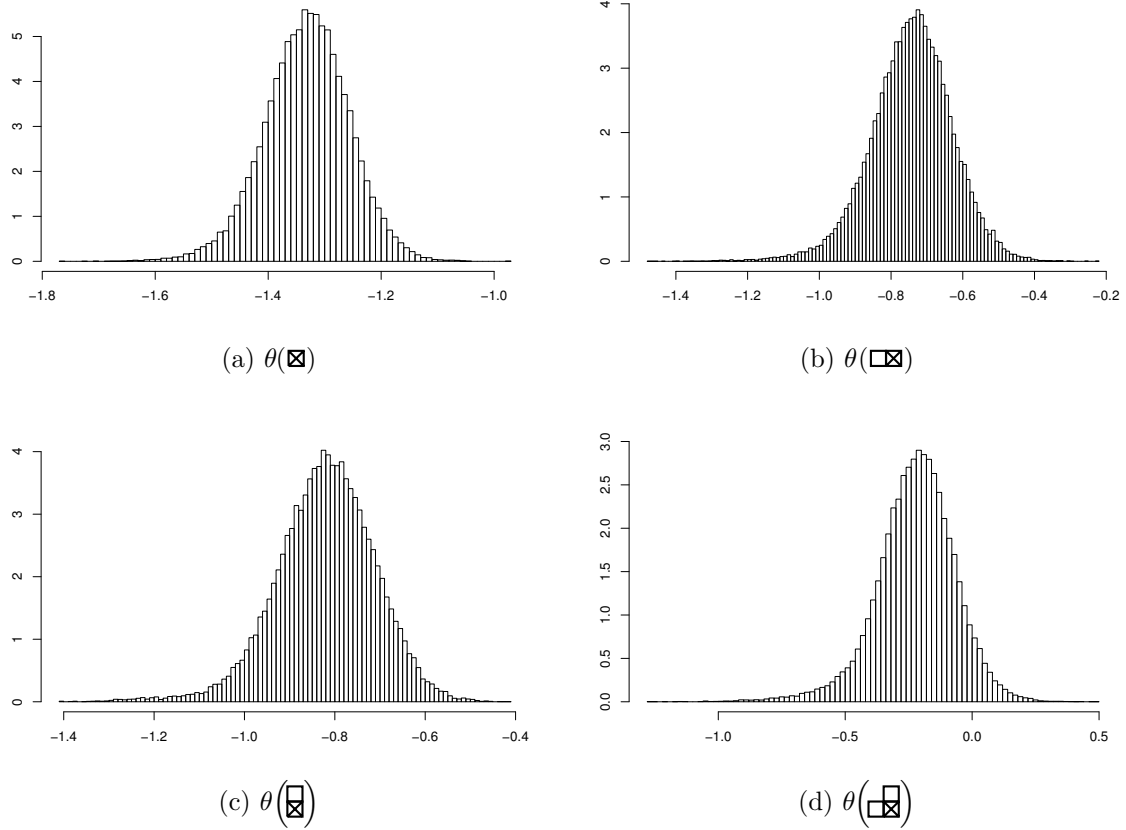


Figure 8: Cancer mortality map example: Histograms of the simulated parameter values $\theta(\cdot)$ for the top four a posteriori most likely interactions.

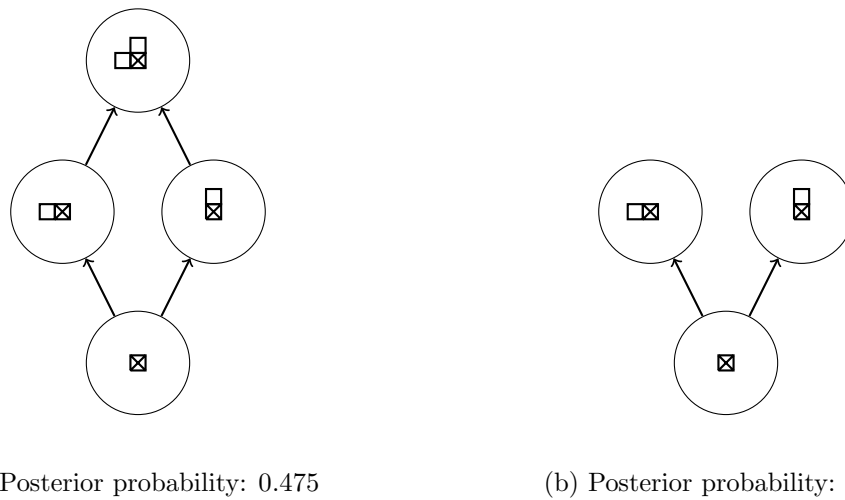


Figure 9: Cancer mortality map example: The two a posteriori most likely models and the corresponding estimated posterior probabilities.

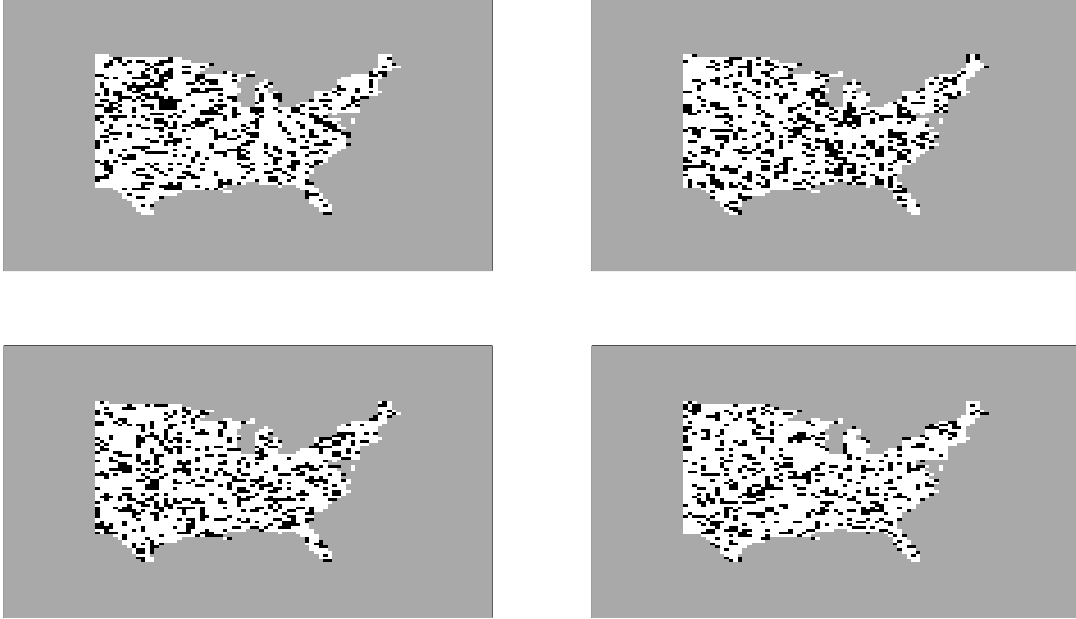


Figure 10: Cancer mortality map example: Four Markov mesh model realizations where the models used are randomly sampled from all models simulated in the RJMCMC (after the specified burn-in). The color coding is the same as in Figure 6(a).

Table 2: Cancer mortality map example: Estimated spatial posterior correlation function when (left) the nodes are numbered in the lexicographical order, and (right) the nodes are numbered right to left and top to down.

	-2	-1	0	1	2		-2	-1	0	1	2
-2	0.002	0.006	0.017	0.004	0.001	-2	0.003	0.005	0.017	0.006	0.002
-1	0.005	0.028	0.124	0.014	0.002	-1	0.005	0.033	0.122	0.024	0.005
0	0.013	0.106	1.000	0.106	0.013	0	0.012	0.097	1.000	0.097	0.012
1	0.002	0.014	0.124	0.028	0.005	1	0.005	0.024	0.122	0.033	0.005
2	0.001	0.004	0.017	0.006	0.002	2	0.002	0.006	0.017	0.005	0.003

one should note that for both node orderings the correlations become stronger in the SE-NW direction than in the NE-SW direction. To study the effect of conditioning on zero values outside the extended lattice and in particular to check whether or not the unobserved boundary we have included is sufficiently large to remove the effect of the boundary, we estimate also the posterior mean function by averaging over many realizations from the simulated Markov mesh models. Close to the boundary of the extended lattice we can clearly see the effect of conditioning to zero, but inside the observed area the estimated mean function is constant except for some small Monte Carlo error. We thereby conclude that the unobserved area we have included is sufficiently large to eliminate the border effect.

To study further the small scale properties of the simulated scenes, we consider the 16 possible configurations in a 2×2 block of nodes. For each of these configurations, we find in a realization the fraction of such blocks that has the specified configuration. By repeating this for a large number of realizations we estimate the posterior distribution for the fraction of 2×2 blocks with a specified configuration in a realization. This distribution should be compared with the corresponding fraction in the observed data set. Figure 11 shows the estimated density for each of the 16 configurations. The corresponding fractions for the observed data set are marked by vertical dotted lines. Note that for most of these distributions the corresponding fractions for the observed data set are centrally located in the distribution. The exceptions are (g) and partly (i) and (j), where the observed quantity is more in the tail of the distribution. The large scale properties of the simulated scenes can be studied correspondingly by choosing one or more univariate function of a scene. It is of course possible to define infinitely many such functions, and unless specific functions is of special interest for an application it is not clear what functions to pick. We are not studying this any further here.

5.2 Sisim data set

In this example we reconsider a data set previously studied in Stien and Kolbjørnsen (2011). The scene, shown in Figure 12(a), is simulated by the sequential indicator simulation procedure (Journel, 1982; Deutsch and Journel, 1998) and it is a much used example scene in the geostatistical community. We name the data set "sisim". The sisim scene is represented on a 121×121 lattice. To reduce the boundary effects of the Markov mesh model we again include unobserved nodes around the observed area, shown as gray in Figure 12(a).

Again adopting the Markov mesh and prior models defined in Sections 2 and 3 and the hyper-parameters defined above, we use the RJMCMC setup discussed above to explore the resulting posterior distribution. For this data set each iteration of the algorithm requires more computation time than in the cancer mortality map data, so we run the Markov chain for only 1 250 000 iterations. To evaluate the convergence properties of the simulated Markov chain, we study trace plots of different scalar quantities in the same way as in Section 5.1.

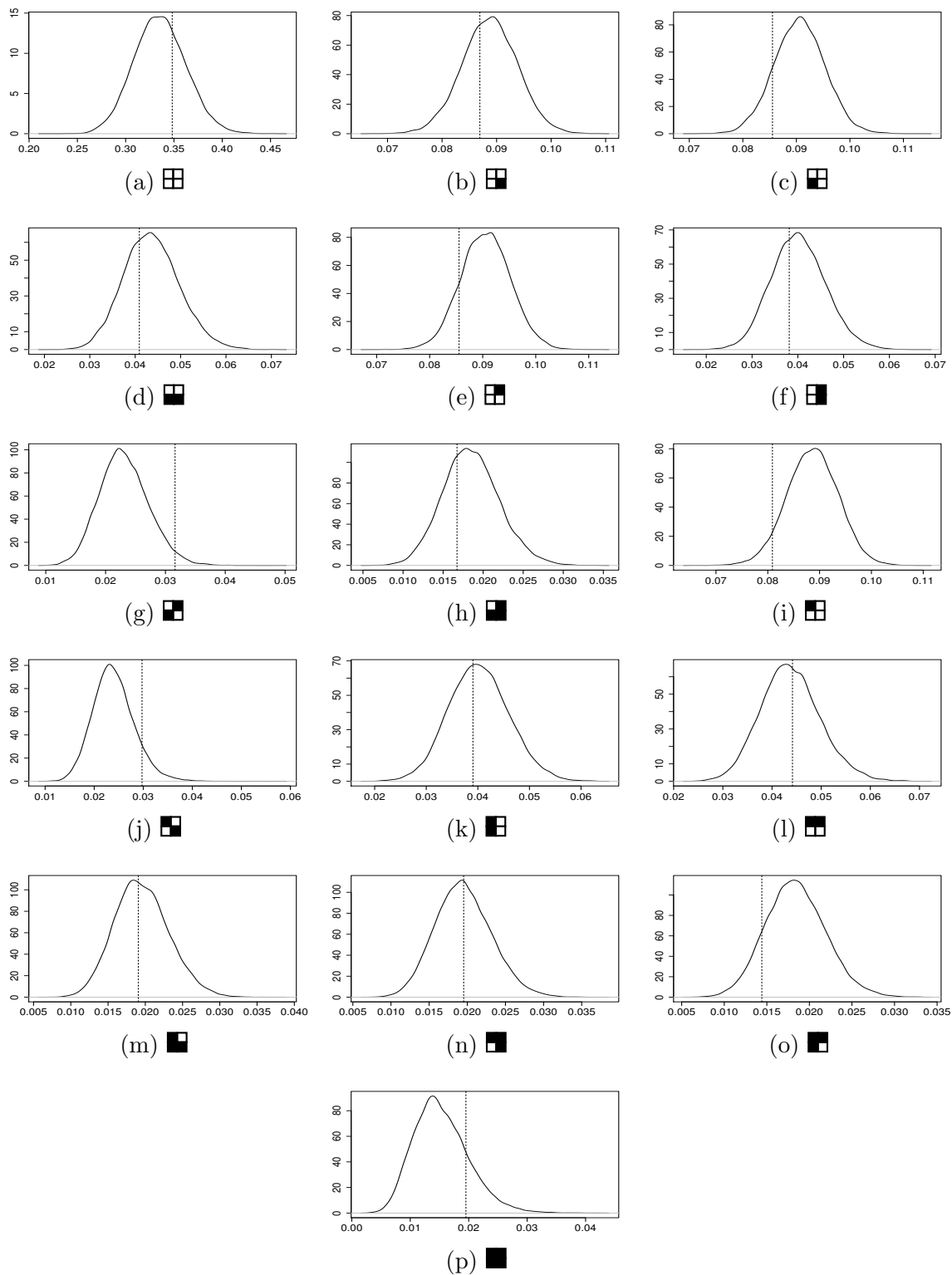


Figure 11: Cancer mortality map example: Estimated a posteriori marginal densities for each of the possible 16 configurations in a 2×2 block of nodes. Corresponding values computed from the cancer map data set is shown as a vertical dotted line. The configuration corresponding to an estimated density is shown below each figure, where black and white nodes represent one and zero, respectively.

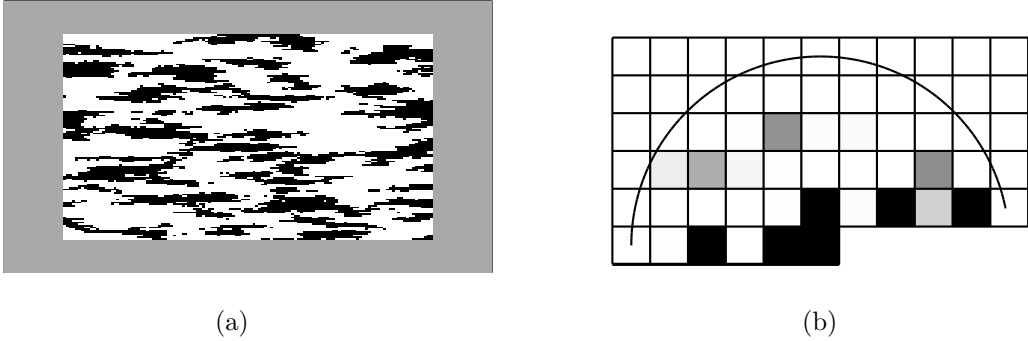


Figure 12: Sisim data set example: (a) Given scene. Nodes added to the lattice to reduce the boundary effects of the Markov mesh model is shown in gray. (b) Map of estimated a posteriori probabilities for each node $v \in \tau_0$ to be a neighbor. A gray-scale is used to visualize the probabilities, where black and white represents one and zero, respectively.

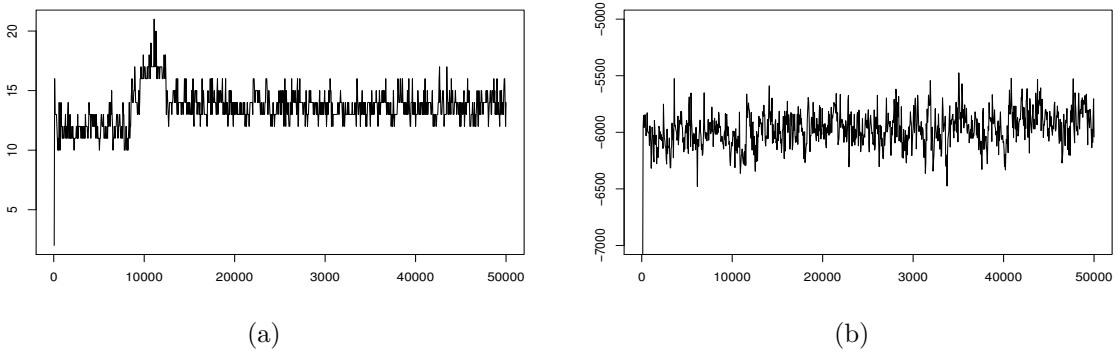


Figure 13: Sisim data set example: Trace plots of the first 50 000 iterations of the RJMCMCM run. (a) Number of interactions $|\Lambda|$, (b) logarithm of the posterior density $\log [f(\tau, \Lambda, \{\theta(\lambda) : \lambda \in \Lambda\}, x_u | x_o)]$.

Figure 13 shows trace plots of the first 50 000 iterations for the number of interactions and for the logarithm of the posterior density. At first glance at these two and the other trace plots we have studied, we asserted that the simulated chain had converged at least within the first 30 000 – 40 000 iterations. As an extra precaution we discarded the first 250 000 iterations when estimating posterior properties.

As in Section 5.1, we estimate the posterior probability for $v \in \tau_0$ to be in the template sequential neighborhood τ . The results are shown in Figure 12(b), where we use a grayscale to visualize the probabilities. There are five nodes whose estimated posterior probabilities are essentially equal to 1, and these are $(0, -1)$, $(-1, 0)$, $(-1, 2)$, $(0, -3)$ and $(-1, 4)$. Four more nodes have estimated posterior probabilities higher than 0.1. These are $(-2, 3)$, $(-3, -1)$, $(-2, -3)$ and $(-1, 3)$ with estimated probabilities 0.444608, 0.425779, 0.323181 and 0.182879, respectively. It is interesting to note the spatial locations of the high probability nodes. At

Table 3: Sisim data set example: The top 20 a posteriori most likely interactions and their estimated posterior probabilities.

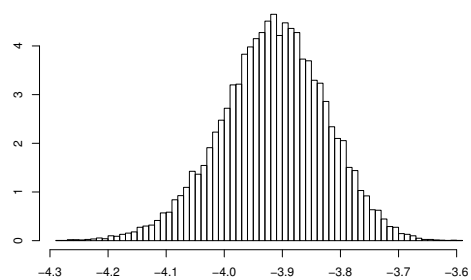
Interaction	⊠	□⊠	⊠	⊠□	□ ⊠
Probability	1.0000	1.0000	1.0000	1.0000	1.0000
Interaction	□ □⊠	⊠ □	□ ⊠ □	□⊠ □	□ □⊠ □
Probability	1.0000	1.0000	0.8572	0.8525	0.8484
Interaction	⊠ □	⊠ □	□	□	□⊠
Probability	0.7351	0.4446	0.4258	0.3232	0.1949
Interaction	□ ⊠	⊠ □	□ ⊠ □	□ □⊠	□ ⊠ □
Probability	0.1882	0.1829	0.1794	0.1531	0.1260

least for a part of the area every second node is chosen as a neighbor with high probability. To understand this effect, we must remember that the values of two nodes that are lying next to each other are highly correlated, so one would not gain much extra information by including both of them in the template sequential neighborhood. Moreover, the prior prefers parsimonious models, which we obtain by not including too many nodes in the template sequential neighborhood.

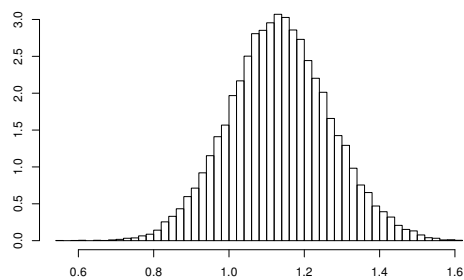
Next, as for the cancer mortality map data set, we correspondingly estimate the posterior probabilities for each possible interaction to be included in the model. Table 3 shows the top 20 a posteriori most likely interactions and corresponding estimated probabilities. We see that many interactions have high posterior probabilities.

We also estimate the a posteriori marginal distributions for the parameter values $\theta(\cdot)$ corresponding to the top eight most likely interactions. Figure 14 depicts the histograms of the simulated parameter values $\theta(\cdot)$. From the simulation we also estimate the posterior probability for each of the possible models. The most probable model is shown in Figure 15. This model has posterior probability equal to 0.13802. The remaining probability mass is spread out on a very large number of models.

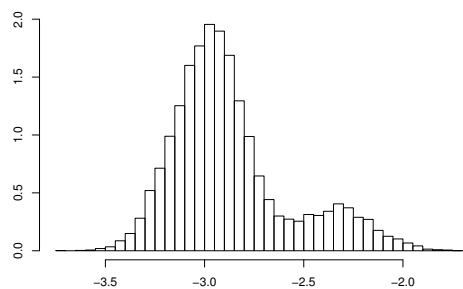
As in the cancer mortality data set example, we also now generate realizations from the simulated Markov mesh models. Figure 16 contains realizations simulated from four randomly chosen models simulated in the Markov chain (after the specified burn-in). As in Figure 12(a), showing the observed data set, black and white nodes v represent $x_v = 1$ and 0, respectively. Using a large number of such realizations we also for this data set estimate the spatial mean



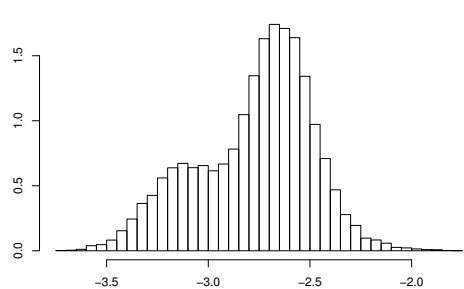
(a) $\theta(\boxtimes)$



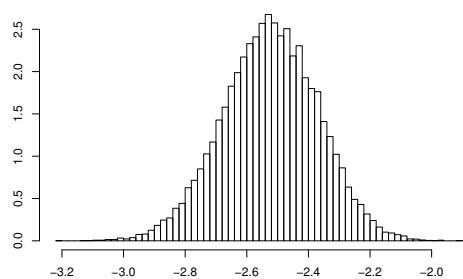
(b) $\theta(\square\boxtimes)$



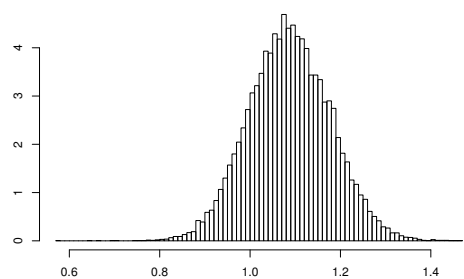
(c) $\theta(\square\boxtimes)$



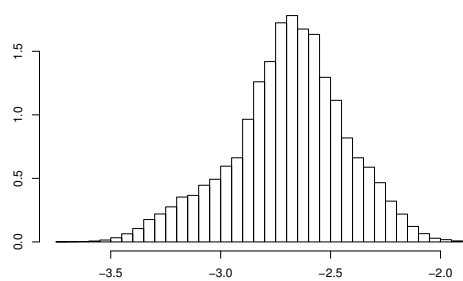
(d) $\theta(\boxtimes\square\square)$



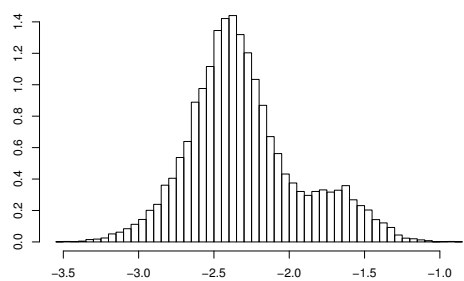
(e) $\theta(\square\boxtimes)$



(f) $\theta(\square\boxtimes\square)$



(g) $\theta(\boxtimes\square\square)$



(h) $\theta(\square\boxtimes\square\square)$

Figure 14: Sisim data set example: Histograms of the simulated parameter values $\theta(\cdot)$ for the top eight a posteriori most likely interactions.

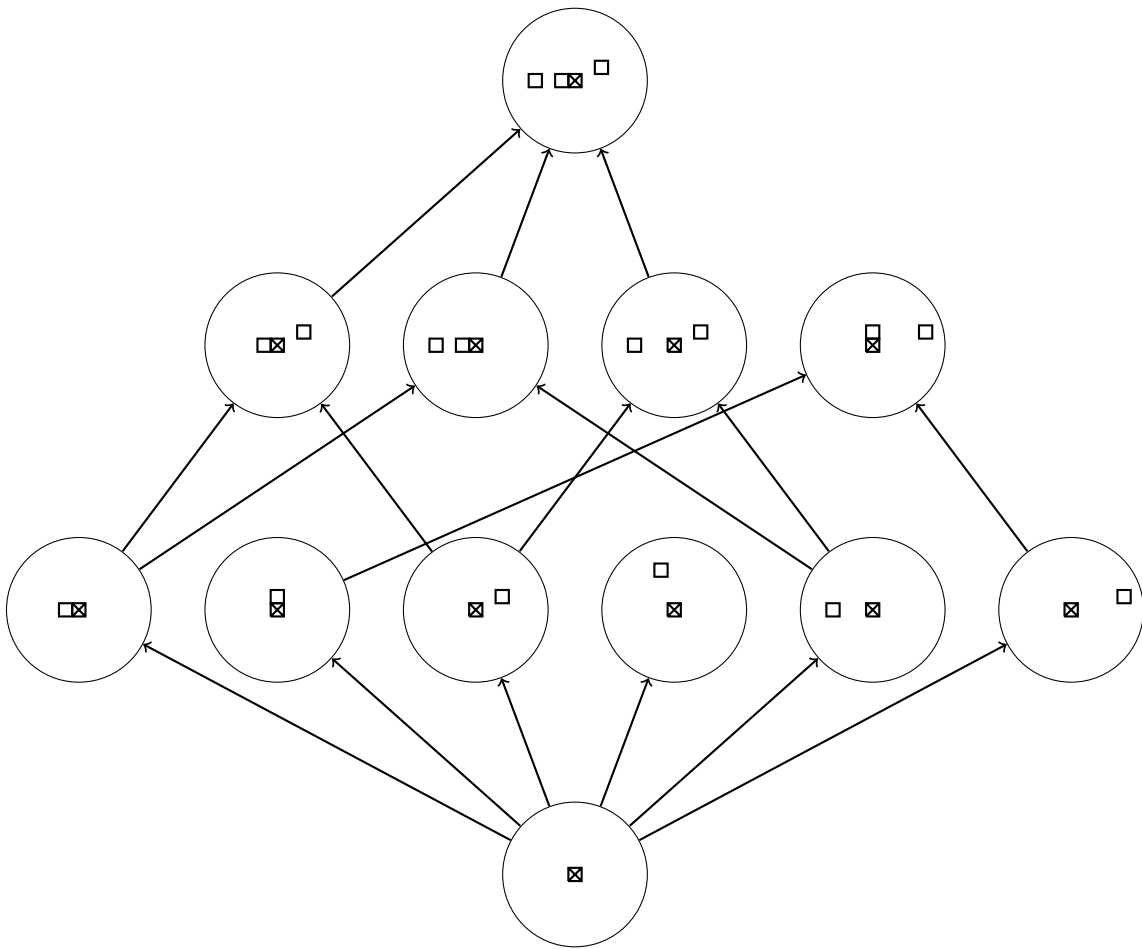


Figure 15: Sisim data set example: The a posteriori most likely model. The estimated posterior probability for this model is 0.13802.

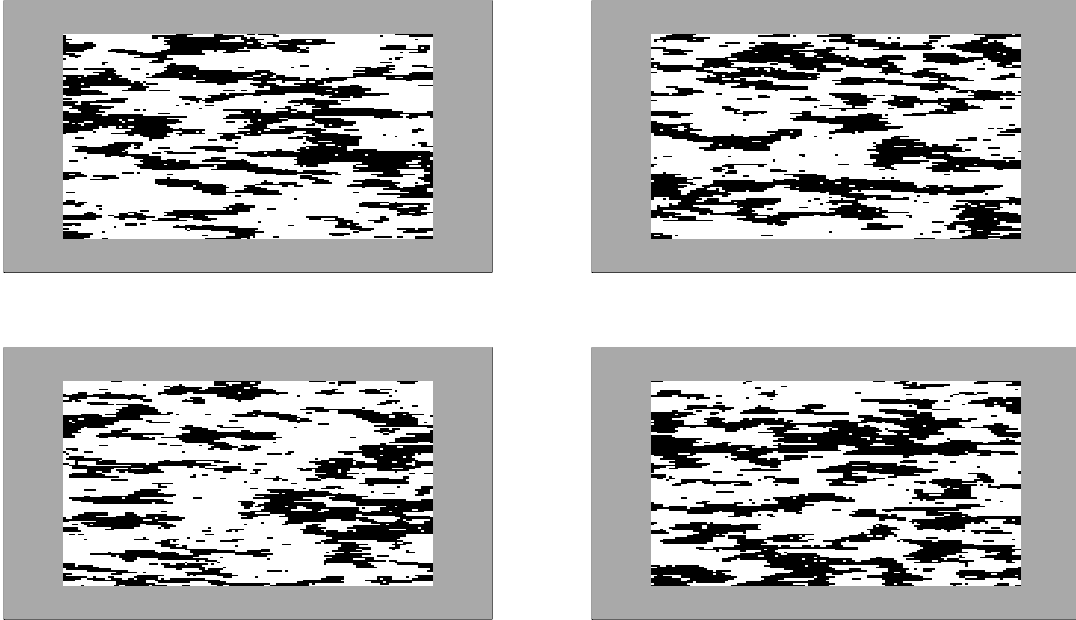


Figure 16: Sisim data set example: Four Markov mesh model realizations where the models used are randomly sampled from all models simulated in the RJMCMC (after the specified burn-in). The color coding is the same as in Figure 12a.

function. As for the cancer mortality data set example, the effect of conditioning to zero outside the extended lattice can only be seen in the unobserved area added to the original observed lattice. Again we also estimate the distribution of values in a 2×2 block of nodes. Figure 17 shows the estimated density for each of the 16 configurations. The corresponding fractions for the observed data set are marked by vertical dotted lines. Note that for most of these distributions, the corresponding fractions for the observed data set are centrally located in the distribution. The exceptions are (c), and partly (e), (f), (i) and (j), where the observed quantities are more in the tail of the distribution.

In the cancer mortality data set example, essentially all of the posterior probability mass was concentrated in a few models. In the sisim data set example, the probability mass is spread out on a very large number of models. In particular, as also discussed above, the most probable model has a posterior probability estimated to be as low as 0.13802. Using the simulated models to understand the posterior model distribution is then more difficult. As a first step in describing the posterior model distribution, our focus here is on whether it has one or several modes. To do this we first need to define what we should mean by a mode in this complicated model space. We start by defining two models to be neighbors if one of them can be obtained from the other by including one extra interaction. Thus, our proposal distribution in Section 4.2, proposing to change the set of active interactions, is

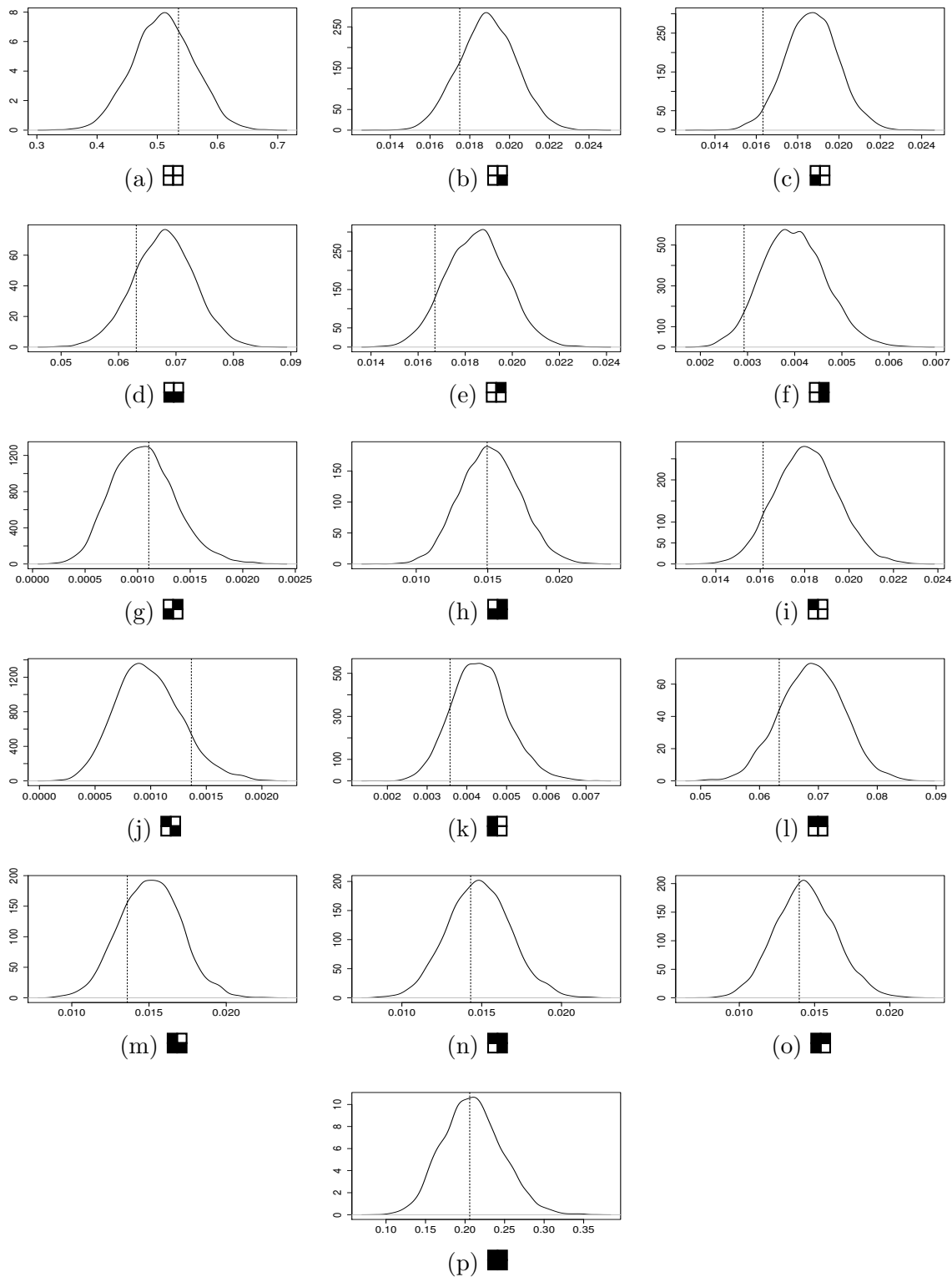


Figure 17: Sisim data set example: Estimated posterior marginal densities for each of the possible 16 configurations in a 2×2 block of nodes. Corresponding values computed from the sisim data set is shown as a vertical dotted line. The configuration corresponding to an estimated density is shown below each figure, where black and white nodes represent one and zero, respectively.

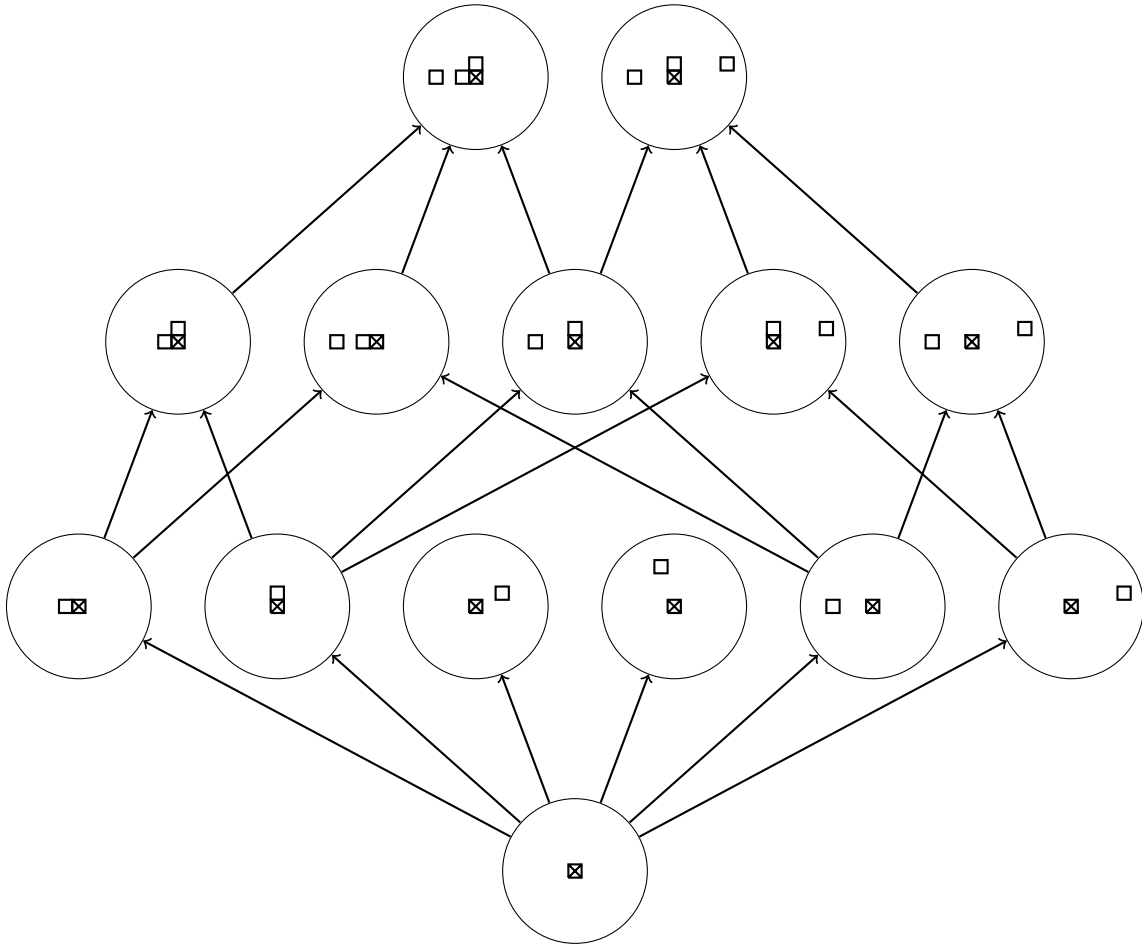


Figure 18: Sisim data set example: The estimated a posteriori most probable model in the second cluster of models.

always generating a potential new model that is a neighbor of the current model. To explore whether we have several modes in our posterior distribution, we first subsample the simulated Markov chain, keeping a realization every 50 iterations after the burn-in period. This leave us with 20 000 realizations. From these we first find the most frequent model, visualized in Figure 15, and then all neighbor models to this most probable model, all neighbor models to the neighbors, and so on until the process stops. The sum of the estimated posterior probabilities of the models in the resulting cluster of models is 0.80755, giving a clear indication that the posterior model distribution have more than one mode. To find a second mode we limit the attention to the simulated models that was not included in the first cluster of models and repeat the process. Thus, we first find the a posteriori most probable model not included in the first model cluster. This model is shown in Figure 18. Then we find all neighbors of this model, all neighbors of the neighbors and so on. The estimated posterior probability in this

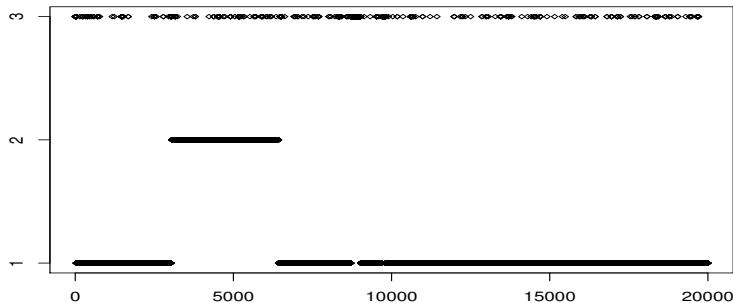


Figure 19: Sisim data set example: Trace plot visited clusters for the subsampled models. On the y -axis 1 and 2 represent the first and second clusters of models found, respectively, and 3 represents all remaining models.

second cluster of models is 0.146563. Thus, these two first clusters contain more than 95% of the simulated models, and we therefore choose not to search for a third cluster. Knowing that we have two important clusters or modes it is natural to reconsider the convergence and mixing properties of our Markov chain. Figure 19 shows a trace plot of the visited clusters for the subsampled models, where 1 and 2 on the y -axis represent the first and second clusters found, respectively, and 3 represents all remaining models. We then see that the second cluster is in fact visited only once, giving a clear indication of poor mixing. We should thereby not trust the estimated probabilities for the two clusters given above, but that the chain is first moving from the first cluster to the second and thereafter back again clearly shows that both of them have a significant posterior probability mass.

6 Closing remarks

In this article we propose a prior distribution for a binary Markov mesh model. The specification of a Markov mesh model has three parts. First a sequential neighborhood is specified, next the parametric form of the conditional distributions is defined, and finally we assign values to the parameters. We formulate prior distributions for all these three parts. To favor parsimonious models, our prior in particular assigns positive prior probabilities for some interaction parameters to be exactly zero. A corresponding prior formulation has previously been proposed for Markov random fields (Arnesen and Tjelmeland, 2017). The advantage of using it for a Markov mesh model is that an explicit and easy to compute expression is available for the resulting posterior distribution, whereas the posterior based on a Markov random field will include the computationally intractable normalizing constant of the Markov random field.

To sample from the resulting posterior distribution when conditioning on an observed scene, we adopt the RJMCMC setup. We propose an algorithm based on the combination

of two proposal distributions, a Gibbs proposal for the parameter values and a reversible jump proposal changing the sequential neighborhood and parametric form of the conditional distributions.

To explore the performance of the specified prior distribution and the corresponding RJMCMC posterior simulation algorithm, we consider two scenes. The first is an observed cancer mortality map data with small spatial coupling between neighboring nodes. For this scene the RJMCMC algorithm converges quickly and has good mixing properties. Most of the posterior mass ends up in models with only two nodes in the sequential neighborhood. The second scene we tried is a frequently used scene in the geostatistical community. It has more spatial continuity than the first scene. The convergence of the RJMCMC algorithm becomes much slower when conditioning on this scene. In particular the posterior seems to have at least two modes and the mixing between the modes is slow. Our simulation results indicate that the a posteriori most likely model has six nodes in the sequential neighborhood and the conditional distributions has a parametric form with as much as twelve parameters. This shows that the specified prior is flexible in that the model complexity favored by the corresponding posterior adapts to the the complexity of the scene conditioned on.

In this article we have focused on binary Markov mesh models and thereby binary scenes. Our strategy for prior specification and posterior simulation, however, can easily be extended to a situation with more than two colors. The main challenge in this generalization does not lie in the specification of the prior, but is computational in that one should expect the convergence and mixing of a corresponding RJMCMC algorithm to be slower for a multi-color model. A direction for future research is therefore to improve the proposal distributions to obtain better convergence and mixing for the RJMCMC algorithm, both in the binary and multi-color cases. In particular we think a promising direction here is to define an MCMC algorithm where several Metropolis–Hastings proposals can be generated in parallel and where the proposals may add and remove more than just one interaction relative to the current model.

References

- Abend, K., Harley, T., and Kanal, L. (1965). “Classification of binary random patterns.” *IEEE Transactions on Information Theory*, 11, 538–544.
- Arnesen, P. and Tjelmeland, H. (2017). “Prior specification of neighbourhood and interaction structure in binary Markov random fields.” *Statistics and Computing*, 27, 737–756.
- Austad, H. and Tjelmeland, H. (2017). “Approximate computations for binary Markov random fields and their use in Bayesian models.” *Statistics and Computing*, 27, 1271–1292.

- Cressie, N. and Davidson, J. (1998). “Image analysis with partially ordered Markov models.” *Computational Statistics and Data Analysis*, 29, 1–26.
- Cucula, L. and Marin, J.-M. (2013). “Bayesian inference on a mixture model with spatial dependence.” *Journal of Computational and Graphical Statistics*, 22, 584–597.
- Deutsch, C. and Journel, A. (1998). *GSLIB: Geostatistical Software Library*. 2nd ed. Oxford: Oxford University Press.
- 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. (2013). “Evidence and Bayes factor estimation for Gibbs random fields.” *Journal of Computational and Graphical Statistics*, 22, 518–532.
- Friel, N., Pettitt, A. N., Reeves, R., and Wit, E. (2009). “Bayesian inference in hidden Markov random fields for binary data defined on large lattices.” *Journal of Computational and Graphical Statistics*, 18, 243–261.
- Gilks, W. R. (1992). “Derivative-free adaptive rejection sampling for Gibbs sampling.” In *Bayesian Statistics 4*, eds. J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, 641–649. Oxford University Press, Oxford, UK.
- Grabisch, M., Marichal, J. L., and Roubens, M. (2000). “Equivalent representations of set functions.” *Mathematics of Operations Research*, 25, 157–178.
- Green, P. J. (1995). “Reversible jump MCMC computation and Bayesian model determination.” *Biometrika*, 82, 711–732.
- Hammer, P. L. and Holzman, R. (1992). “Approximations of pseudo-Boolean functions; applications to game theory.” *Methods and Models of Operation Research*, 36, 3–21.
- Hammer, P. L. and Rudeanu, S. (1968). *Boolean Methods in Operation Research and Related Areas*. Berlin: Springer.
- Heikkinen, J. and Högmänder, H. (1994). “Fully Bayesian approach to image restoration with an application in biogeography.” *Applied Statistics*, 43, 569–582.
- Higdon, D. M., Bowsher, J. E., Johnsen, V. E., Turkington, T. G., Gilland, D. R., and Jaszczak, R. J. (1997). “Fully Bayesian estimation of Gibbs hyperparameters for emission computed tomography data.” *IEEE Transactions on medical imaging*, 16, 516–526.
- Hurn, M., Husby, O., and Rue, H. (2003). “A tutorial on image analysis.” In *Spatial statistics and computational methods*, ed. J. Møller, vol. 173 of *Lecture Notes in Statistics*, 87–139. Springer.

- Journal, A. (1982). “The indicator approach to estimation of spatial distributions.” In *17th APCOM Symposium Proceedings*. Society of Mining Engineers.
- Kindermann, R. and Snell, J. L. (1980). *Markov random fields and their applications*. Providence, R.I.: American Mathematical Society.
- Lauritzen, S. (1996). *Graphical Models*. Oxford: Clarendon Press.
- Liang, F. (2010). “A double Metropolis–Hastings sampler for spatial models with intractable normalizing constants.” *Journal of Statistical Computation and Simulation*, 80, 1007–1022.
- Liang, F., Liu, C., and Carroll, R. (2011). *Advanced Markov Chain Monte Carlo Methods: Learning from Past Samples*. Wiley.
- Riggan, W. B., Creason, J. P., Nelson, W. C., Manton, K. G., Woodbury, M. A., Stallard, E., Pellom, A. C., and Beaubier, J. (1987). *U.S. Cancer Mortality Rates and Trends, 1950–1979*. Vol. IV (U.S. Government Printing Office, Washington, DC: Maps, U.S. Environmental Protection Agency).
- Sherman, M., Apanasovich, T. V., and Carroll, R. J. (2006). “On estimation in binary autologistic spatial models.” *Journal of Statistical Computation and Simulation*, 76, 167–179.
- Stien, M. and Kolbjørnsen, O. (2011). “Facies modeling using a Markov mesh model specification.” *Mathematical Geosciences*, 43, 611–624.
- Stoehr, J., Pudlo, P., and Cuccala, L. (2015). “Adaptive ABC model choice and geometric summary statistics for hidden Gibbs random fields.” *Statistics and Computing*, 25, 129–141.

A Log-concavity of the full conditional for α

In this appendix we prove that the full conditional $f(\alpha|\tau, \Lambda, \{\theta(\lambda) + \alpha\Delta(\lambda) : \lambda \in \Lambda\}, x)$ defined in (32) is log-concave, so that we can use adaptive rejection sampling to generate samples from it. Defining $g(\alpha) = \ln[f(\alpha|\tau, \Lambda, \{\theta(\lambda) + \alpha\Delta(\lambda) : \lambda \in \Lambda\}, x)]$ and using (32) we have

$$\begin{aligned}
 g(\alpha) &= \ln[f(\{\theta(\lambda) + \alpha\Delta(\lambda) : \lambda \in \Lambda\}|\tau, \Lambda)] \\
 &\quad + \ln[f(x|\tau, \Lambda, \{\theta(\lambda) + \alpha\Delta(\lambda) : \lambda \in \Lambda\})] + C,
 \end{aligned}
 \tag{43}$$

where C is the logarithm of the normalizing constant in (32). Inserting expressions for the prior and likelihood in (29) and (13), respectively, we get

$$\begin{aligned}
g(\alpha) = & \sum_{\lambda \in \Lambda} \left[c(\lambda) + \theta(\lambda) + \alpha \Delta(\lambda) - 2 \ln \left(1 + e^{\theta(\lambda) + \alpha \Delta(\lambda)} \right) - \frac{(\theta(\lambda) + \alpha \Delta(\lambda))^2}{2\sigma^2} \right] \\
& + \sum_{v \in \chi} [x_v (\theta(\xi(x) \cap (\tau \oplus v)) + \alpha \Delta(\xi(x) \cap (\tau \oplus v))) \\
& \quad - \ln \left(1 + e^{\theta(\xi(x) \cap (\tau \oplus v)) + \alpha \Delta(\xi(x) \cap (\tau \oplus v))} \right)] + C.
\end{aligned} \tag{44}$$

Grouping terms of the same functional form, we get

$$\begin{aligned}
g(\alpha) = & C_0 + C_1 \alpha - \frac{1}{2\sigma^2} \sum_{\lambda \in \Lambda} (\theta(\lambda) + \alpha \Delta(\lambda))^2 - 2 \sum_{\lambda \in \Lambda} \ln \left(1 + e^{\theta(\lambda) + \alpha \Delta(\lambda)} \right) \\
& - \sum_{v \in \chi} \ln \left(1 + e^{\theta(\xi(x) \cap (\tau \oplus v)) + \alpha \Delta(\xi(x) \cap (\tau \oplus v))} \right),
\end{aligned} \tag{45}$$

where

$$C_0 = C + \sum_{\lambda \in \Lambda} \theta(\lambda) + \sum_{v \in \chi} x_v \theta(\xi(x) \cap (\tau \oplus v)) \text{ and } C_1 = \sum_{\lambda \in \Lambda} \Delta(\lambda) + \sum_{v \in \chi} \Delta(\xi(x) \cap (\tau \oplus v)) \tag{46}$$

are constants as a function of α . The second derivative of the constant and linear terms in (45) are of course zero. Since the coefficients of the quadratic terms are all negative, the second derivative of all of these are less or equal to zero, and unless $\Delta(\lambda)$ equals zero for all $\lambda \in \Lambda$ the second derivative of the sum of these terms is even strictly less than zero. The remaining terms in (45) all have the same functional form as a function of α , namely

$$h(\alpha) = -a \ln \left(1 + e^{b+c\alpha} \right), \tag{47}$$

a term in the sum over $\lambda \in \Lambda$ has $a = 2$, $b = \theta(\lambda)$ and $c = \Delta(\lambda)$, whereas a term in the sum over $v \in \chi$ has $a = 1$, $b = \theta(\xi(x) \cap (\tau \oplus v))$ and $c = \Delta(\xi(x) \cap (\tau \oplus v))$. To prove that the second derivative of all of these terms are negative, it is thereby sufficient to show that $h''(\alpha) < 0$ for all $a > 0$ and $\alpha, b, c \in \mathbb{R}$. Simple differentiation gives

$$h''(\alpha) = -\frac{ac^2 e^{b+c\alpha}}{(1 + e^{b+c\alpha})^2}. \tag{48}$$

Thus, $h''(\alpha) < 0$ for all $a > 0$ and $\alpha, b, c \in \mathbb{R}$, and thereby $g(\alpha)$ is concave and the full conditional $f(\alpha | \tau, \Lambda, \{\theta(\lambda) + \alpha \Delta(\lambda) : \lambda \in \Lambda\}, x)$ is log-concave.

B Jacobian determinant for the proposal in Section 4.2.1

The Jacobi determinant for our removing an active interaction from Λ proposal is $\det(A)$, where A is defined by (39). The exact form of the matrix A depends on how we define the

vectors θ and θ^* used in (39). The vector θ should contain the set of current parameters $\{\theta(\lambda) : \lambda \in \Lambda\}$, but so far we have not specified what order to use when arranging this set of parameters into the vector θ . Correspondingly, we have not specified what order to use when arranging the set of potential new parameters $\{\theta^*(\lambda) : \lambda \in \Lambda^*\}$ into the vector θ^* . However, even if the elements of A depend on how we construct θ and θ^* , the absolute value of the determinant of A is the same for all arrangements of θ and θ^* . To find $\det(A)$ we arrange the vector θ so that parameters corresponding to lower order interactions come first. The first element of the vector θ is thereby $\theta(\emptyset)$, thereafter follows parameters corresponding to the first order interactions $\{\theta(\lambda) : \lambda \in \Lambda, |\lambda| = 1\}$ (in an arbitrary order), then all parameters corresponding to second order interactions $\{\theta(\lambda) : \lambda \in \Lambda, |\lambda| = 2\}$ (again in an arbitrary order), and so on. We arrange θ^* correspondingly, parameters corresponding to lower order interactions come first.

As also touched on in Section 4.2.1, the transformation in (39) can be done in three steps. First θ is transformed into a vector β of the corresponding current interaction parameters $\{\beta(\lambda) : \lambda \in \Lambda\}$. This relation is given in (15) and is in particular linear so we can write

$$\beta = A_1 \theta. \quad (49)$$

Arranging also the vector β so that lower order interactions comes first, it is easy to see from (15) that A_1 is a lower triangular matrix with all diagonal elements equal to one. Thus $\det(A_1) = 1$. The second step in the transformation is to use (36) to define a vector β^* containing the set of potential new interaction parameters $\{\beta^*(\lambda) : \lambda \in \Lambda^*\}$. As the proposal is to remove an interaction, the number of elements in β^* is one less than the number of elements in β . To obtain a one-to-one relation as required in the reversible jump setup, we include the current value $\beta(\lambda^*)$ in a vector together with β^* . We let $\beta(\lambda^*)$ be the last element in the vector and we arrange also the vector β^* so that lower order interaction parameters come first. As the relation in (36) is linear we can then write

$$\begin{bmatrix} \beta^* \\ \beta(\lambda^*) \end{bmatrix} = A_2 \beta, \quad (50)$$

where the elements of the square matrix A_2 is defined by (36). To find the determinant of A_2 , let r denote the number of elements in β before $\beta(\lambda^*)$, so that element number $r + 1$ in β is $\beta(\lambda^*)$. From (36) it then follows that A_2 has the block structure,

$$A_2 = \begin{bmatrix} I_{r \times r} & A_2^{12} \\ 0_{(|\Lambda|-r) \times r} & A_2^{22} \end{bmatrix}, \quad (51)$$

where $I_{r \times r}$ is the $r \times r$ identity matrix, A_2^{12} is an $r \times (|\Lambda| - r)$ matrix, $0_{(|\Lambda|-r) \times r}$ is a $(|\Lambda| - r) \times r$ matrix of only zeros, and A_2^{22} is the $(|\Lambda| - r) \times (|\Lambda| - r)$ permutation matrix where the elements $(i, i + 1)$ for $i = 1, \dots, |\Lambda| - r - 1$ and $(|\Lambda| - r, 1)$ equals one and all other elements are zero.

Thereby we have $\det(A_2) = \det(I_{r \times r}) \cdot \det(A_2^{22}) = \det(A_2^{22})$, and as A_2^{22} is a permutation matrix its determinant is plus or minus one. Thus, $|\det(A_2)| = 1$. The third step in the transformation from θ to θ^* is to use (14) to transform the vector of potential new interaction parameters, β^* , to a corresponding vector θ^* of potential new parameter values. As the relation in (14) is also linear, we can write

$$\begin{bmatrix} \theta^* \\ \beta(\lambda^*) \end{bmatrix} = A_3 \begin{bmatrix} \beta^* \\ \beta(\lambda^*) \end{bmatrix}, \quad (52)$$

where the elements of the matrix A_3 is defined by (14). Recalling that we have arranged the elements in both θ^* and β^* so that parameters corresponding to lower order interactions come first, it is easy to see from (14) that A_3 is an upper triangular matrix with all diagonal elements equal to one. Thus $\det(A_3) = 1$. Setting the three steps in the transformation together we have $A = A_1 A_2 A_3$ and thereby $|\det(A)| = |\det(A_1)| \cdot |\det(A_2)| \cdot |\det(A_3)| = 1$.