N-fold way simulated tempering for pairwise interaction point processes

Wilfrid S. Kendall [a], Yuan Shen [b,1], Elke Thönnes [b,*,1]

[a] *Department of Statistics, University of Warwick*

[b] *Centre for Scientific Computing and Department of Statistics, University of Warwick*

Abstract

Pairwise interaction point processes with strong interaction are usually difficult to sample. We discuss how Besag lattice processes can be used in a simulated tempering MCMC scheme to help with the simulation of such processes. We show how the N-fold way algorithm can be used to sample the lattice processes efficiently and introduce the N-fold way algorithm into our simulated tempering scheme. To calibrate the simulated tempering scheme we use the Wang-Landau algorithm.

Key words: Pairwise interaction point processes, Besag lattice processes, MCMC, simulated tempering, N-fold way algorithm, Wang-Landau algorithm
1991 MSC: 60J10, 68U05, 60G55, 60D05

1 Introduction

Pairwise interaction point processes are common models for random point pattern. However, they are usually not amenable to analytic examination and therefore require simulation-based inference. Markov chain Monte Carlo (MCMC) methods produce Markov chains that can be used to sample such processes. However, for models with strong interaction, these Markov chains often suffer from high autocorrelation and so the resulting estimates have high asymptotic variances. To improve mixing of these chains (and thus reduce the autocorrelation) methods like simulated tempering are being used. In this paper we develop a variant of simulated tempering based on the use of Besag

* Corresponding author. Present address: Department of Statistics, University of Warwick, Coventry CV4 7AL, UK.

[1] Supported by EPSRC first grant GR/S52780/01.

lattice processes. These define point processes via the joint distribution of point counts in cells dividing the sampling window. We can sample these lattice processes using an N-fold way algorithm, a method originally introduced for Markov random fields, which reduces computation time for Markov chains with long waiting times between transitions from one state to another.

The paper is structured as follows. First we introduce pairwise interaction point processes and Besag lattice processes. Then we provide a review on MCMC methods and ways of improving these. We then discuss our N-fold way approach to simulated tempering and illustrate it on the example of a Strauss process. The last section discusses the results thus obtained.

2 Point processes and Besag lattice processes

Point processes are models for random pattern of points. They are often defined by a density with respect to a Poisson point process on a bounded window W. For simplicity, we assume that $W = [0,1]^2$, the unit square window. In the following we consider pairwise interaction point processes on W which have the following density with respect to a unit rate Poisson process:

$$f(\mathbf{x}) \quad = \quad \alpha\lambda^{n(\mathbf{x})} \prod_{i<j} g(||x_i - x_j||) \qquad \mathbf{x} \subseteq W. \tag{1}$$

Here $n(\mathbf{x})$ is the number of points in $\mathbf{x}$ and $||\cdot||$ denotes the Euclidean distance in $\mathbb{R}^2$. The parameter λ is positive and α denotes the normalizing constant of the density. If the interaction function g satisfies $0 \leq g(d) < 1$, then the point process exhibits inhibitory interaction, that is it produces regular point pattern. For example, if the interaction function is given by

$$g(d) \quad = \quad \begin{cases} \gamma & \text{if } d < R \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

where $0 \leq \gamma < 1$, then the resulting density

$$\pi(\mathbf{x}) \quad = \quad \alpha\lambda^{n(\mathbf{x})}\gamma^{S_R(\mathbf{x})} \tag{3}$$

specifies the Strauss process with interaction range R [21]. Here $S_R(\mathbf{x})$ counts the number of neighbour pairs in the point pattern $\mathbf{x}$, that is the number of point pairs less than the distance R apart. As $\gamma < 1$ the Strauss process favours point pattern where only a few point pairs are neighbours. If $\gamma = 0$, then f describes a hard core process in which points are at least a minimum distance R apart from each other.

In [2] the authors show that the distribution of a purely inhibitory pairwise interaction process can be seen as the limit distribution of a sequence of auto-Poisson processes. Suppose we subdivide the sampling window into a regular grid of m equal sized square cells $C_r, r = 1, \ldots, m$. Let ξ_i be the centroid, $\mathbf{x}_i$ the point pattern and n_r the number of points in cell C_r. To define the distribution of an auto-Poisson process we specify the probability of observing the cell counts $\mathbf{n} = (n_1, \ldots, n_m)$, where n_r is the number of points in cell C_r, as proportional to

$$\prod_{1 \le r \le m} \frac{(\lambda/m)^{n_r}}{n_r!} \prod_{r<s} g(||\xi_r - \xi_s||)^{n_r n_s}.$$

Now, given the point count in each cell we can produce a point pattern on W by generating a binomial process for each cell, that is, the locations of the n_r points are independent and uniformly distributed in cell C_r. The resulting point pattern X has a density with respect to a unit rate Poisson process given by

$$f_m(\mathbf{x}) \quad \propto \quad \lambda^{n(\mathbf{x})} \prod_{r<s} g(||\xi_r - \xi_s||)^{n_r n_s}. \tag{4}$$

Now, [2] show that as the cell area $1/m$ tends to zero we have that $f_m(\mathbf{x})$ tends to the density $f(\mathbf{x})$ defined in (1).

Besag lattice processes such as the auto-Poisson process or the auto-logistic process were originally considered for pseudo-likelihood estimation [2] and also can be derived for marked point processes [11]. They can be simulated easily using a Gibbs sampler as in Section 3.2; for a perfect Gibbs Sampler see [17]. For the Strauss process the auto-Poisson approximation is given by

$$f_m(\mathbf{x}) \quad \propto \quad \lambda^{n(\mathbf{x})} \gamma^{S_R^B(\mathbf{x})} \qquad \text{where } S_R^B(\mathbf{x}) = \sum_{r<s:||\xi_r - \xi_s|| < R} n_r n_s \tag{5}$$

In the following we will call this process the *Strauss lattice process*.

Because within-cell interactions are neglected in the standard auto-Poisson process, its realisations can look very different from the corresponding Markov point process. We propose the use of a modified Besag lattice process where the area of interaction is still determined by the centroids of the cell but within-cell interactions are introduced. We achieve this by defining

$$h_m(\mathbf{x}) \quad \propto \quad \lambda^{n(\mathbf{x})} \Big(\prod_{r<s} g(||\xi_r - \xi_s||)^{n_r n_s} \Big) \Big(\prod_r g(0)^{\frac{n_r(n_r-1)}{2}} \Big).$$

Then we still have that $h_m(\mathbf{x}) \to f(\mathbf{x})$ as $m \to \infty$. For the Strauss process we define the following modified Strauss lattice process through the interaction

function

$$g_m(d) \quad = \quad \begin{cases} \gamma \text{ for } d = 0 \\ \beta \text{ for } 0 < d < R \\ 1 \text{ otherwise} \end{cases}$$

leading to the density

$$h_m(\mathbf{x}) \quad \propto \quad \lambda^{n(\mathbf{x})} \beta^{S_R^B(\mathbf{x})} \gamma^{\frac{n_i(n_i-1)}{2}} \tag{6}$$

where $S_R^B(\mathbf{x})$ is defined as in (5). Here $0 < \beta < 1$ and $0 < \gamma < 1$. Assume that as $m \to \infty$ the interaction function $g_m(d)$ tends to the interaction function of a Strauss process as defined in (2). Then it can be shown easily that as $m \to \infty$ the density $h_m(\mathbf{x}) \to \pi(x)$ where $\pi(x)$ is the density of the Strauss process defined in (3).

3 Markov chain Monte Carlo algorithms

Sampling of Markov point processes usually requires Markov chain Monte Carlo (MCMC) methods. In the following we give a short review of the most common algorithms.

3.1 Metropolis-Hastings algorithm

The Metropolis-Hastings algorithm developed in [9] is often used for point processes and, like all Metropolis-Hastings type algorithms, it consists of proposing a new state and accepting or rejecting the proposal with the appropriate probability. If the state is accepted the chain moves to the proposed state, otherwise the chain remains in its current state. The probability is chosen such that the invariant distribution of the resulting chain coincides with the target distribution.

Suppose the current state of the chain is $X_t = \mathbf{x}$. With probability $p(\mathbf{x})$ the birth of a point ξ, which is sampled according to a density $b(\mathbf{x}, \cdot)$, is proposed. The birth is accepted with probability $\min\{1, \alpha(\mathbf{x}, \xi)\}$ in which case we set $X_t = \mathbf{x} \cup \{\xi\}$. Alternatively, with probability $1 - p(\mathbf{x})$ a point $\eta \in \mathbf{x}$ is sampled according to a probability $d(\mathbf{x}, \eta)$ and its death proposed. The death proposal is accepted with probability $\min\{1, \alpha(\mathbf{x} \backslash \eta, \eta)^{-1}\}$. The acceptance probabilities are specified by the Metropolis-Hastings ratio

$$\alpha(\mathbf{x}, \xi) \quad = \quad \lambda(\mathbf{x}, \xi) \, \frac{1 - p(\mathbf{x} \cup \{\xi\})}{p(\mathbf{x})} \, \frac{d(\mathbf{x} \cup \{\xi\}, \xi)}{b(x, \xi)}$$

where

$$\lambda(\mathbf{x},\xi) \quad = \quad \begin{cases} \frac{f(\mathbf{x}\cup\{\xi\})}{f(\mathbf{x})} & \text{for } f(\mathbf{x})>0 \\ 0 & \text{otherwise} \end{cases}$$

is the Papangelou conditional intensity of the target process with density f. A common choice is to let $b(\mathbf{x},\xi)$ be the uniform density on the sampling window W. For death proposals a point is chosen at random from the current configuration. Note that in pairwise interaction processes with strong inhibitory interaction the Papangelou conditional intensity $\lambda(x,\xi)$ can be very small in large regions of the sampling window. This leads to a high rejection rate of births if the location of proposed points is sampled uniformly on the sampling window.

For a perfect simulation version of the above algorithm see [15] and for a comparison with birth-death processes see [4,5].

3.2 The Gibbs sampler

To simulate the auto-Poisson process we will use a Gibbs sampler. For a more detailed description of Gibbs sampling see for example [7]. The Gibbs sampler applies to multivariate distributions and samples successive states from full conditional distributions.

Consider the modified Strauss lattice process and let N_r be the cell count in cell C_r. The conditional probability $q(k|\mathbf{n}_{(-r)})$ of $N_r = k$ given the point counts $\mathbf{n}_{(-r)}$ in all other cells is proportional to

$$\frac{\mu_r^k}{k!}\,\gamma^{\frac{k(k-1)}{2}}. \tag{7}$$

Here

$$\mu_r \quad = \quad \frac{\lambda}{m}\prod_{s\sim r}\beta^{n_s}$$

where the product is over all R-close cells of cell C_r and n_s is the point count in cell C_s. (Cell C_s is R-close to C_r if $||\xi_r - \xi_s|| < R$). Using the full conditional distributions of cell counts we easily can sample the modified Strauss lattice process using a random-scan Gibbs sampler. Suppose the current state of the chain is $X_t = \mathbf{x}$ where $\mathbf{x} = (x_1,\ldots,x_m)$ is a point pattern with point counts $\mathbf{n} = (n_1,\ldots,n_m)$. To produce X_{t+1} proceed as follows:

(1) Choose at random a cell index $r \in \{1,\ldots,m\}$.
(2) Sample k with probability $q(k|\mathbf{n}_{(-r)})$.
(3) Sample a Binomial point pattern $\mathbf{y}_r$ with k points on cell C_r.
(4) Now X_{t+1} is the point pattern produced from $\mathbf{x}$ by replacing the point pattern $\mathbf{x}_r$ in cell C_r with the point pattern $\mathbf{y}_r$.

4 Improving and accelerating MCMC algorithms

While MCMC is a powerful technique for examining complex probability distributions efficiency can be a bottleneck. Various methods have been developed to improve efficiency and we review the following selection of methods that we will be exploiting in our context:

(1) the N-fold way algorithm,
(2) simulated tempering and
(3) the Wang-Landau algorithm.

For point processes with strong interaction the standard Metropolis-Hastings algorithm described in Section 3.1 or the Gibbs Sampler in Section 3.2 often experience long waiting times between moves from one state in the state space to another. The N-fold way algorithm reduces computation time by observing the target Markov chain only at times when there is a change in its state and recording the waiting between such transitions. To implement the algorithm we need to be able to sample these waiting times and transitions efficiently. As we describe in Section 4.1 this be done for the Gibbs sampler chain for the (modified) Besag lattice processes. (In general it is not possible to define an efficient N-fold way algorithm for pairwise interaction processes as it requires a partitioning of the sampling window into regions with constant Papangelou conditional intensity).

In order to exploit the computational saving produced by the N-fold way algorithm we use a simulated tempering scheme to combine N-fold way sampling of the modified Besag lattice process with standard Metropolis-Hastings sampling for the pairwise interaction process. Simulated tempering is a method that exploits faster mixing Markov chains defined on the same state space to help the slowly mixing target chain escape from local modes. More details are given in Section 4.2. To be able to apply simulated tempering we need to compute ratios of normalizing constants of the relevant invariant distributions. As this cannot be done analytically for our problem numerical approximations have to be found. The Wang-Landau algorithm is a clever way of avoiding such computation by using an adaptive algorithm that determines these quantities during the run-time of the simulated tempering algorithm.

4.1 The N-fold way algorithm

Because the Gibbs sampler is based on full conditional distributions, high correlation between the components of the multivariate target distribution leads to a slowly mixing Gibbs chain. For the Strauss lattice process this happens when there is strong repulsion between points. In this case the parameter μ_r

will be very small for a large number of cells. This leads to a large number of empty cells which often are updated to the empty set, thus leading to no change in the state of the chain. This problem can be alleviated by using the N-fold way algorithm [3]. This method is based on the idea of sub-sampling a Markov chain at times when there is a change in the state of the chain. Estimation using the sub-sampled chain is then based on weighted averages where the weighting is according to the lifetime of a state. While this does not change the mixing properties of the chain itself it does reduce the computing time needed to produce a sample of a fixed size.

In more detail, the N-fold way algorithm on a discrete state space works as follows. Firstly, given the current state $X_t = \mathbf{x}$ of the chain the probability $p(\mathbf{x})$ of a transition in the next time step to a state different from $\mathbf{x}$ is determined. Let $L_{\mathbf{x}}$ be a Geometrically distributed random variable with parameter $p(\mathbf{x})$ which we call the lifetime of state $\mathbf{x}$. We set $X_{t+k} = \mathbf{x}$ for $k < L_{\mathbf{x}}$ and then sample $X_{t+L_{\mathbf{x}}}$ from its conditional distribution given that $X_{t+L_{\mathbf{x}}} \neq \mathbf{x}$ and $X_{t+L_{\mathbf{x}}-1} = \mathbf{x}$.

For the modified Strauss lattice process an N-fold way algorithm can be derived as follows. Cell C_r is picked for updating with probability $1/m$. A change in the point pattern $\mathbf{x}_r$ in cell C_r always occurs unless the cell is empty and updated to the empty configuration. Assume cell C_r is empty, then the probability of the cell being updated to an empty configuration is given by $q(0|\mathbf{n}_{(-r)})$ as defined in (7). Thus given the current state $X_t = \mathbf{x}$ the probability $p(\mathbf{x})$ of a transition to a point pattern different to $\mathbf{x}$ at time $t+1$ is equal to

$$p(\mathbf{x}) \quad = \quad 1 - \frac{1}{m} \sum_{r:\mathbf{x}_r=\emptyset} q(0|\mathbf{n}_{(-r)}).$$

Hence state $\mathbf{x}$ is given a lifetime $L_{\mathbf{x}}$ which is Geometrically distributed with parameter $p(\mathbf{x})$. The state $X_{t+L_{\mathbf{x}}}$ is now produced as follows.

(1) Pick cell C_r with probability

$$\frac{1}{p(\mathbf{x})}\left(\frac{1}{m}\,\mathbf{1}_{[\mathbf{x}_r\neq\emptyset]} + \frac{1-q(0|\mathbf{n}_{(-r)})}{m}\,\mathbf{1}_{[\mathbf{x}_r=\emptyset]}\right).$$

(2) (a) If cell C_r is not empty then sample k with probability $q(k|\mathbf{n}_{(-r)})$.
(b) If cell C_r is empty then sample $k > 0$ with probability

$$q(k|\mathbf{n}_{(-r)})/(1-q(0|\mathbf{n}_{(-r)})).$$

(3) Replace the point pattern $\mathbf{x}_r$ in C_r by a Binomial point pattern $\mathbf{y}_r$ on C_r consisting of k points.

Note that an N-fold way algorithm is also useful when using Besag lattice processes for maximum pseudo-likelihood estimation. The asymptotic variance

of the estimator can be reduced by using a Rao-Blackwell approach [4]. Let $x_1, \ldots, x_n$ be the states sampled by the N-fold way algorithm and $L_{x_1}, \ldots, L_{x_n}$ the corresponding observed lifetimes. To estimate $\mathbb{E}_\pi(f(X))$ we can use the standard ergodic average which is equivalent to the lifetime weighted average

$$\bar{f}_1 \quad = \quad \frac{1}{\sum_{t=1}^n L_{x_t}} \sum_{t=1}^n f(x_t) L_{x_t}$$

A Rao-Blackwell-type estimate can be produced by replacing the observed lifetimes L_{x_t} by their expectation $1/p(x_t)$ and so the following estimator has smaller asymptotic variance than $\bar{f}_1$:

$$\bar{f}_2 \quad = \quad \frac{1}{\sum_{t=1}^n 1/p(x_t)} \sum_{t=1}^n f(x_t)/p(x_t).$$

4.2 Simulated tempering

Simulated tempering [10,19] is an MCMC approach that aims at improving mixing of the target MCMC sampler by exploiting faster mixing samplers with stationary distribution different to the target distribution.

Let $\pi_1(x) = \pi(x)$ be the unnormalized target density and let $\pi_2(x), \ldots, \pi_M(x)$ be a set of unnormalized auxiliary densities. Generally π_i is in some sense "flatter" than π_{i+1} and so an MCMC sampler with stationary density π_i tends to explore the state space better. For example, we might choose

$$\pi_i(x) \quad = \quad \Big(\pi(x)\Big)^{\frac{1}{1+\lambda(i-1)}}, \qquad i = 1, \ldots, M; \qquad \lambda > 0.$$

In this paper we will explore a sequence of auxiliary distributions that are based on a coarser auto-Poisson approximation to the target distribution.

Simulated tempering samples a chain $(X, I)_{t \geq 0}$ with stationary distribution

$$\pi(x, i) \quad \propto \quad c_i \pi_i(x), \qquad i \in \{1, \ldots, M\}$$

and inference on $\pi(x)$ can be done by simply retaining values (x, i) where $i = 1$. Let the current state of the chain be $(X_t, I_t) = (x, i)$. One iteration of simulated tempering consists of an update of X_t according to the sampler with stationary distribution π_i followed by an update of the index I_t. In the following we call I_t the level. To perform an update of $I_t = i$ we propose a new index j with user-defined probability q_{ij}. The proposed index is accepted with probability

$$\alpha(x, i, j) \quad = \quad \min\left\{1, \frac{c_j \pi_j(x)\ q_{ji}}{c_i \pi_i(x)\ q_{ij}}\right\}. \tag{8}$$

The constants $c_i, i = 1, \ldots, M$, are chosen such that the chain divides its time roughly equally among the M levels. This is ensured if c_i is equal to the normalizing constant of π_i. As these normalizing constants usually are not available approximations need to be found before the simulated tempering scheme is run. A common method used is reverse logistic regression, see [8].

Successive auxiliary densities π_i and π_{i+1} should be sufficiently similar to allow the simulated tempering chain to move from one level to another. On the other hand π_M should be sufficiently different from π so that the corresponding chain is fast mixing. Literature suggests a spacing that leads to an average acceptance rate for moves between neighbouring levels of about 20% to 40%.

4.3 The Wang-Landau algorithm

The Wang-Landau algorithm [22,1] is an adaptive MCMC approach based on a sequence of stochastic approximations of the target density $\pi(x)$ by mixture distributions of the form

$$\pi_n(x) \quad = \quad \sum_{i=1}^{M} \frac{\pi(x)}{\theta_n(i)} \mathbf{1}_{[x \in \mathcal{S}_i]}, \qquad n \in \mathbb{N}, \tag{9}$$

where $\mathbf{1}$ denotes the indicator function. Here $\mathcal{S}_1, \ldots \mathcal{S}_M$, is a disjoint partition of the state space $\mathcal{S}$ of π which is chosen such that the mass $\int_{\mathcal{S}_i} \pi(x)dx$ is roughly equal for all $i \in \{1, \ldots, M\}$. The random sequences $\theta_n(i), i \in \{1, \ldots, M\}$, for $n \in \mathbb{N}$ are defined as

$$\theta_n(i) = \Phi_n(i) / \sum_{j=1}^{M} \Phi_n(j) \qquad \text{where} \tag{10}$$

$$\Phi_n(i) = \Phi_{n-1}(i)(1 + \gamma_{n-1} \mathbf{1}_{[X_n \in \mathcal{S}_i]}) \text{ and} \tag{11}$$

$$\Phi_0(i) \quad \text{is set to some user-defined value.} \tag{12}$$

Here $(\gamma_n)_{n \in \mathcal{N}}$ is a positive, non-increasing sequence that we define later and $X = (X_n)_{n \in \mathbb{N}}$ is a controlled Markov chain with limit distribution π. This controlled Markov chain is evolved as follows. Let $P_n(x, \cdot)$ be a transition kernel with stationary distribution $\pi_n(x)$ as defined in (9). Given the current state $X_n = x$ we sample X_{n+1} from $P_n(x, \cdot)$ and then update $\theta_n(i)$ for $i \in \{1, \ldots, M\}$ as defined in equations (10)-(12). Note that $\Phi_n(i)$ may be interpreted as an occupation measure of $\mathcal{S}_i$. Recall that the partition of $\mathcal{S}$ is chosen such that in equilibrium each partition set has equal mass and so asymptotically the chain will spend equal time in each partition set. As the occupation measure $\Phi_n(i)$ increases the mixture distribution π_n and thus the stationary distribution of X_n is adapted such that the probability of the chain leaving $\mathcal{S}_i$ is increased.

To ensure that the controlled Markov chain has the correct asymptotic distribution π the sequence $(\gamma_n)_{n\in\mathbb{N}}$ needs to converge to zero. It should converge slowly enough to avoid the algorithm becoming stuck but not too slowly as the rate of convergence of the controlled Markov chain X depends on the rate of convergence of $(\gamma_n)_{n\in\mathbb{N}}$ towards zero. A major contribution of the Wang-Landau algorithm is a method of adaptively decreasing γ_n according to the following mechanism. At random times $\tilde{\tau}_1 < \tilde{\tau}_2 < \ldots$ we set

$$\gamma_{\tilde{\tau}_k+1} \quad = \quad (1+\gamma_0)^{\frac{1}{2^k}} - 1$$

for some initial value $1 < \gamma_0 + 1 < e$. Between these random times γ_n remains constant, that is

$$\gamma_n \quad = \quad \gamma_{\tilde{\tau}_k+1} \quad \text{for } \tilde{\tau}_k < n \leq \tilde{\tau}_{k+1}.$$

The random times $\tilde{\tau}_n$ are chosen to be times at which the occupation measures $\Phi(n)$ become approximately equal. Wang and Landau [22] suggest to measure this as follows. Equal occupation measures are assumed if

$$\min_{i=1,\ldots,M} \theta_n(i) \quad \geq \quad c\,\frac{1}{M}\sum_{j=1}^{M}\theta_n(j)$$

where c is some constant close to one, say 0.8. Atchade and Liu [1] recommend to use times $\tau_n = \min\{a, \tilde{\tau}_n\}$ for some large integer a such that $n^\lambda \gamma_n \leq K$ for constants $K < \infty$ and $\lambda > \frac{1}{2}$. This is to ensure that the time-varying transition kernels P_n stabilize and thus the algorithm converges.

The Wang-Landau algorithm can also be applied to simulated tempering [1]. Here the partition of $\mathcal{S} = \cup_{i=1}^{M}\{i\}\times\mathcal{X}$ is given by $\mathcal{S}_i = \{i\}\times\mathcal{X}$, $i \in \{1,\ldots,M\}$. Suppose the current state of the chain is $(X_t, I_t) = (x, i)$. A proposal to update the level I_t to state j then has acceptance probability

$$\alpha_n(x,i,j) \quad = \quad \min\left\{1, \frac{\Phi_n(j)\pi_j(x)\ q_{ji}}{\Phi_n(i)\pi_i(x)\ q_{ij}}\right\}.$$

In this case we have $\theta_n(i) \to 1/c_i$ as $n \to \infty$ and thus the Wang-Landau algorithm avoids the need to compute approximations to the normalizing constants prior to running the simulated tempering algorithm.

For illustration we describe below a Wang-Landau simulated tempering scheme for the Strauss process with density

$$\pi_1(\mathbf{x}) \quad = \quad \lambda_1^{n(\mathbf{x})}\ \gamma_1^{S_R(\mathbf{x})}.$$

The auxiliary densities are Strauss processes with weaker interaction, that is

$$\pi_i(\mathbf{x}) \quad = \quad \lambda_i^{n(\mathbf{x})}\ \gamma_i^{S_R(\mathbf{x})} \qquad i \in \{1,\ldots,\pi_M\}$$

where $\lambda_1 > \lambda_2 > \cdots > \lambda_M$ and $\gamma_1 < \gamma_2 < \cdots \gamma_M$. The interaction radius R is equal for all M densities.

Suppose the current state of the Wang-Landau simulated tempering chain is $(X_t, I_t) = (x, i)$, then we sample (X_{t+1}, I_{t+1}) as follows:

(1) With probability $1/2$ propose a birth. Alternatively, propose a death.
   (a) If we propose a birth we sample a point ξ uniformly on W. With probability
$$\min\left\{1, \gamma_i^{S_R(\mathbf{x}\cup\{\xi\})-S_R(\mathbf{x})} \frac{\lambda_i}{(n(\mathbf{x})+1)}\right\}$$
   we set $X_{t+1} = \mathbf{x} \cup \{\xi\}$. Alternatively, we set $X_{t+1} = \mathbf{x}$.
   (b) If we propose a death and $\mathbf{x} = \emptyset$ we set $X_{t+1} = \emptyset$. If $X_t \neq \emptyset$ then we sample at random a point $\xi \in \mathbf{x}$. With probability
$$\min\left\{1, \gamma_i^{S_R(\mathbf{x}\setminus\{\xi\})-S_R(\mathbf{x})} \frac{n(\mathbf{x})}{\lambda_i}\right\}$$
   we set $X_{t+1} = \mathbf{x}\setminus\{\xi\}$. Alternatively, we set $X_{t+1} = \mathbf{x}$.
(2) With probability $1/2$ we sample $j = \min\{i+1, M\}$. Alternatively we set $j = \max\{i-1, 1\}$. Then with probability
$$\min\left\{1, \frac{\Phi_t(j)\ \lambda_j^{n(\mathbf{x})}\gamma_j^{S_R(\mathbf{x})}}{\Phi_t(i)\ \lambda_i^{n(\mathbf{x})}\gamma_i^{S_R(\mathbf{x})}}\right\}.$$
   we set $I_{t+1} = j$. Alternatively, we set $I_{t+1} = i$.
(3) Finally, for $k \in \{1, \ldots, M\}$ we update $\Phi_t(k)$ to $\Phi_{t+1}(k)$ as defined in (11).

5 N-fold way simulated tempering for the Strauss process

5.1 Simulated tempering using lattice processes

In the following we consider a simulated tempering approach with one auxiliary density π_2 that defines a modified auto-Poisson distribution. In principle this simulated tempering approach could be extended to include several auxiliary densities with auto-Poisson models defined on coarsening grids. Such an approach would produce a multi-resolution type algorithm. Multi-resolution or multi-scale methods have been used very successfully for lattice models such as Markov random fields, see for example [20,14,13,18,12,16]. However to the best of our knowledge such methods have not been used for continuous models such as the point process models that we are examining here. Also note that by having auxiliary densities defined on an increasingly coarse lattice structure we are able to introduce a grid that is so coarse that the algorithm

will update the locations of several points at once. However, more auxiliary densities will lead to the target chain being visited less often. Furthermore, while chains whose equilibrium distribution is defined on a coarser grid will tend to mix better, the computational advantages of using the N-fold way is lost as the expected waiting time in a state becomes shorter. We found it computationally more efficient to consider only one auxiliary density and use it to exploit the advantages of the N-fold way algorithm.

As an example we consider as target density the (unnormalized) density π_1 of the Strauss process:

$$\pi_1(\mathbf{x}) \quad = \quad \lambda^{n(\mathbf{x})}\gamma^{s_R(\mathbf{x})}.$$

Now π_2 is the unnormalized density of a modified Strauss lattice process, that is

$$\pi_2(\mathbf{x}) \quad = \quad \lambda^{n(\mathbf{x})}\beta^{S_R^B(\mathbf{x})}\prod_r \gamma^{\frac{n_r(n_r-1)}{2}}.$$

In our experiments we chose the parameters of the modified Strauss lattice process such that the number of cells m was as small as possible but still produced a sufficiently high average acceptance rate for moves between levels.

5.2 Wang-Landau simulated tempering, the N-fold way

For the modified Strauss lattice model on a fine grid we observe slow mixing of the corresponding Gibbs Sampler when the inhibitory interaction is very strong. Thus we will use the N-fold way algorithm to save computing time. We described the standard N-fold way algorithm to sample a modified auto-Poisson process in Section 4.1. We now augment this algorithm for the Wang-Landau simulated tempering approach described in Section 4.3.

Recall that the simulated tempering scheme produces a chain $(X, I)_{t\geq 0}$ with two components: $(X)_{t\geq 0}$ which describes the current point pattern and $(I)_{t\geq 0}$ which describes the level. An iteration within the simulated algorithm scheme then consist of an update of the point pattern followed by an update of the level. In the Wang-Landau approach to simulated tempering the transition kernels for updates of the level change with time. This may be taken account of by sampling the lifetime $L^t_{\mathbf{x},i}$ of the state $(\mathbf{x}, i)$ entered by the chain at time t not from a Geometric distribution but from the probability mass function given by

$$\mathbb{P}(L^t_{\mathbf{x},i} = k) \quad = \quad \prod_{j=0}^{k-2}\Big(1 - p_{t+j}(\mathbf{x}, i)\Big)\, p_{t+k-1}(\mathbf{x}, i). \tag{13}$$

Here

$$p_t(\mathbf{x}, i) \quad = \quad 1 - \mathbb{P}\Big((X_t, I_t) = (\mathbf{x}, i) \,\Big|\, (X_{t-1}, I_{t-1}) = (\mathbf{x}, i)\Big), \qquad t \in \mathbb{N},$$

and can be computed as follows. We have $X_t = X_{t-1}$ if at time $t-1$ we pick an empty cell and update it to an empty cell. Given $(X_{t-1}, I_{t-1}) = (\mathbf{x}, i)$ this occurs with probability

$$q(\mathbf{x}, i) \quad = \quad \frac{1}{m} \sum_{r: x_r = \emptyset} q_i(0|n(-r)).$$

Here $q_i(k|n(-r))$ is the conditional probability mass function of the number of points in cell C_r given the point counts $n_{(-r)}$ in all other cells for the modified Strauss lattice process with density $\pi_i(\mathbf{x})$.

Next, we derive the probability of $I_t = I_{t-1}$ given that $X_t = X_{t-1}$ and $(X_{t-1}, I_{t-1}) = (x, i)$. Recall that q_{ij} is the probability of proposing to move from level i to level j. Also recall that the probability of accepting the proposal j at time t given $X_t = x$ and $I_{t-1} = i$ is given by

$$\alpha_t(\mathbf{x}, i, j) \quad = \quad \min\left\{1, \frac{\Phi_t(j)\pi_j(\mathbf{x})\ q_{ji}}{\Phi_t(i)\pi_i(\mathbf{x})\ q_{ij}}\right\}.$$

Then

$$\begin{aligned} 1 - p_t(\mathbf{x}, i) &= \mathbb{P}\Big((X_t, I_t) = (\mathbf{x}, i) \;\Big|\; (X_{t-1}, I_{t-1}) = (\mathbf{x}, i)\Big) \\ &= q(\mathbf{x}, i)\Big[q_{i,i-1}[1 - \alpha_t(\mathbf{x}, i, i-1)] + q_{i,i+1}[1 - \alpha_t(\mathbf{x}, i, i+1)]\Big] \end{aligned}$$

This defines the distribution of the lifetime $L^t_{\mathbf{x},i}$.

We now describe the Wang-Landau simulated tempering scheme with one auxiliary density, although the algorithm can easily be extended to a scheme with more than one auxiliary density. Recall that

$$\pi_1(\mathbf{x}) \quad = \quad \lambda_1^{n(\mathbf{x})} \gamma_1^{s_R(\mathbf{x})}$$

and

$$\pi_2(\mathbf{x}) \quad = \quad \lambda_2^{n(\mathbf{x})} \beta_2^{S_R^B(\mathbf{x})} \prod_r \gamma_2^{\frac{n_r(n_r-1)}{2}}.$$

As mentioned earlier, N-fold way simulation is only feasible for the Markov chain with invariant density π_2. We augment the state space of the simulated tempering chain by an additional variable that keeps track of the lifetimes. Choose a starting value $(X_0, I_0, Z_0) = (x_0, 1, 1)$ for the chain and set $\Phi_0(1) = \Phi_0(2) = 1$. Suppose the current state of the chain is $(X_t, I_t, Z_t) = (x, i, z)$. We then proceed as follows:

(1) If $i = 1$ then perform the following steps:
   (a) With probability $1/2$ propose a birth. Alternatively, propose a death.

(i) If we propose a birth we sample a point ξ uniformly on W. With probability

$$\min\left\{1,\ \gamma_1^{S_R(\mathbf{x}\cup\{\xi\})-S_R(\mathbf{x})}\ \frac{\lambda_1}{(n(\mathbf{x})+1)}\right\}$$

we set $X_{t+1}=\mathbf{x}\cup\{\xi\}$. Alternatively, we set $X_{t+1}=\mathbf{x}$.

(ii) If we propose a death and $\mathbf{x}=\emptyset$ we set $X_{t+1}=\emptyset$. If $X_t\neq\emptyset$ then we sample at random $\xi\in\mathbf{x}$. With probability

$$\min\left\{1,\ \gamma_1^{S_R(\mathbf{x}\backslash\{\xi\})-S_R(\mathbf{x})}\ \frac{n(\mathbf{x})}{\lambda_1}\right\}$$

we set $X_{t+1}=\mathbf{x}\backslash\{\xi\}$. Alternatively, set $X_{t+1}=\mathbf{x}$.

(b) With probability

$$\min\left\{1,\ \frac{\Phi_t(2)\ \lambda_2^{n(\mathbf{x})}\beta_2^{S_R^B(\mathbf{x})}\prod_r\gamma_2^{\frac{n_r(n_r-1)}{2}}}{\Phi_t(1)\ \lambda_1^{n(\mathbf{x})}\gamma_1^{S_R(\mathbf{x})}}\right\}.$$

we set $I_{t+1}=2$. Alternatively, set $I_{t+1}=1$.

(c) For $k\in\{1,2\}$ update $\Phi_t(k)$ to $\Phi_{t+1}(k)$ as defined in (11).

(d) If $I_{t+1}=1$ set $Z_{t+1}=t+1$. Alternatively, if $I_{t+1}=2$, sample the lifetime $L_{x,2}^{t+1}=L$ according to the probability mass function defined in (13) and set $Z_{t+1}=t+L$.

(2) If $i=2$ then perform the following steps:

(a) If $Z_t>t$ set $(X_{t+1},I_{t+1},Z_{t+1})=(X_t,I_t,Z_t)$. For $k\in\{1,2\}$ update $\Phi_t(k)$ to $\Phi_{t+1}(k)$ as defined in (11).

(b) If $Z_t=t$ then proceed as follows:

(i) With probability $q(x,i)$ set $X_{t+L}=\mathbf{x}$. Alternatively set $X_{t+L}=\mathbf{y}$ where $\mathbf{y}\neq\mathbf{x}$ is sampled as follows. Choose cell C_r with probability

$$\frac{1}{1-q(\mathbf{x},i)}\left(\frac{1}{m}\mathbf{1}_{[x_r\neq\emptyset]}+\frac{1-q_i(0|\mathbf{n}_{(-r)})}{m}\mathbf{1}_{[x_r=\emptyset]}\right).$$

If cell C_r is not empty sample k with probability $q_i(k|\mathbf{n}_{(-r)})$. If cell C_r is empty sample $k>0$ with probability

$$q_i(k|\mathbf{n}_{(-r)})/(1-q_i(0|\mathbf{n}_{(-r)}).$$

Then $\mathbf{y}$ is the point pattern produced by replacing in $\mathbf{x}$ the pattern x_r with a Binomial pattern on cell C_r with k points.

(ii) If $X_{t+1}=\mathbf{x}$ then set $I_{t+1}=2$ with probability

$$\frac{\alpha_t(\mathbf{x},2,2)}{\alpha_t(\mathbf{x},2,1)+\alpha_t(\mathbf{x},2,2)}$$

and $I_{t+1} = 1$ otherwise. If $X_{t+1} = \mathbf{y} \neq \mathbf{x}$ then perform a standard update of the level. Set $I_{t+1} = 1$ with probability $0.5\alpha_t(\mathbf{y}, 2, 1)$ and set $I_{t+1} = 2$ otherwise.

(iii) If $I_{t+1} = 1$ set $Z_{t+1} = t + 1$. If If $I_{t+1} = 2$ sample the lifetime $L_{x,2}^{t+1} = L$ according to the probability mass function defined in (13) and set $Z_{t+1} = t+L$. For $k \in \{1, 2\}$ update $\Phi_t(k)$ to $\Phi_{t+1}(k)$ as defined in (11).

6 Experimental Results

6.1 Assessing performance

The standard approach to assess the performance is to examine the auto-correlation function of a summary statistic of the point pattern. Commonly used summary statistics are the number of points, the number of neighbour pairs and the average first coordinate of the points. We additionally propose to use a summary statistic more specific to spatial point pattern: the multivariate K-function [6]. Given two point patterns x and y, the multivariate K-function essentially looks at the number of points in pattern x that lie close to a point in y. More formally, the multivariate K-function of a point process X and a point process Y is given by

$$\begin{aligned}\lambda_Y K_{X,Y}(r) = \mathbb{E}(&\text{number of points in } Y \text{ within distance } r \\ &\text{of an arbitrary point in } X) \quad = \quad \lambda_X K_{Y,X}(r),\end{aligned}$$

where λ_X and λ_Y are the intensity of X and Y respectively. An estimate of the K-function can be computed as follows:

$$K(r, x, y) \quad = \quad \frac{|W|}{n(x)n(y)} \sum_{x_i \in x} \sum_{y_j \in y} \mathbf{1}_{[||x_i - y_j|| \leq r]}.$$

We use periodical boundary conditions to account for edge effects. If x and y are independent then $K(r, x, y) = \pi r^2$. To stabilize the variance, it is common to consider the so-called L-function which is defined as

$$L(r, x, y) \quad = \quad \sqrt{K(r, x, y)/\pi} - r$$

Under the hypothesis of independence L-function is constant zero. Thus to measure the dependence between successive point configurations sampled by our MCMC algorithm we compute

$$\hat{L}(\tau, r) \quad = \quad \frac{1}{N - \tau} \sum_{k=1}^{n-\tau} L(r, x_k, x_{k+\tau}),$$

where x_t is the observed state of the Markov chain at time n. We expect these estimates to decay to zero as τ increases subject to sampling fluctuations. To decide when the L-function has converged to zero we calculate an upper confidence envelope.. The envelope is obtained by randomly translating the first point pattern x_k and estimating the L-function for the translated pattern and $x_{k+\tau}$. This procedure is repeated independently 99 times. The 100(1-α)% confidence upper envelope is then given by the pointwise $100(1-\alpha)$th percentile of the 99 L-function estimates. For more information on using confidence envelopes and related Monte Carlo tests, see [6].

6.2 Example: A Strauss process

In the following we consider the Strauss process as an example case. However, our methods apply to any Markov point process that can be approximated by an auto-Poisson or auto-logistic lattice process. We examine sampling from a Strauss process on the window $W = [0, 2.5]^2$ with parameters

$$\lambda = 1000, \qquad \gamma = 10^{-5}, \qquad \text{and} \qquad R = 0.45.$$

This is a process with very strong inhibitory interaction which is difficult to sample and so a good test case for our method. The average number of points for this process is about 20 and Figure 1 shows a sample.

As discussed earlier we use one auxiliary density $\pi_2(\mathbf{x})$ which describes a modified Strauss lattice process. The parameter of the auxiliary density are chosen as follows:

$$m = 32^2, \quad \lambda = 1000 \quad \beta = 10^{-5} \quad \gamma = 10^{-5} \quad R = 0.45.$$

These parameters lead to an average acceptance rate of 2% for moves between levels. This is lower than the values recommended in the literature and using a finer grid would increase the average acceptance rate. However, a finer grid makes the sampling process computationally more expensive. More importantly it produces a Markov chain that is slower mixing as it effectively updates smaller region in the sampling space. In simulation experiments we found that using a finer grid produced little improvement in terms of mixing at a significantly higher computational expense.

For comparison we also run a standard simulated tempering scheme based on auxiliary densities that describe Strauss process with increasingly weaker inhibitory interaction. The parameters of the auxiliary densities are as follows:

$$
\begin{array}{lll}
\lambda_2 = 600 & \gamma_2 = 0.002 & R_2 = 0.45, \\
\lambda_3 = 380 & \gamma_3 = 0.0066 & R_3 = 0.45, \\
\lambda_4 = 315 & \gamma_4 = 0.02 & R_4 = 0.45, \\
\lambda_5 = 210 & \gamma_5 = 0.05 & R_5 = 0.45, \\
\lambda_6 = 65 & \gamma_6 = 0.1 & R_6 = 0.45, \\
\lambda_7 = 30 & \gamma_7 = 0.22 & R_7 = 0.45, \\
\lambda_8 = 12.5 & \gamma_8 = 0.45 & R_8 = 0.45, \\
\lambda_9 = 7.2 & \gamma_9 = 0.66 & R_9 = 0.45, \\
\lambda_{10} = 3.35 & \gamma_{10} = 1 & R_{10} = 0.45,
\end{array}
$$

The above parameters were chosen such that the average acceptance rate of moves between levels lie between 20% and 40%. The hottest chain (sampling the Strauss process with the weakest interaction) is in fact a Poisson process which can be sampled directly.

6.3 Results and Discussion

Both simulated tempering schemes were run for 5×10^6 iterations. Then the cold chain was subsampled to yield a timeseries of length 10^4. Figure 2 shows the auto-correlation function of the number of points in the sub-sampled cold chain for both the N-fold way simulated tempering scheme and the standard simulated tempering algorithm. Figures 3 and 4 show the L-function for both schemes for length scales $r = 0.05$ and $r = 0.1$ respectively.

First note that the auto-correlation function of the number of points suggests negligible auto-correlation at smaller lags than the L-functions. This provides evidence to the fact that a high -level summary statistic like the number of points is unlikely to provide an adequate description of the auto-correlation of the chain. Similary, we found the number of neighbour pairs not a useful summary statistic as our target distribution has such strong repulsoin that neighbour pairs hardly occur. Thus we would recommend the use of the L-function which is specifically designed to detect "correlation" between point patterns.

Recall that r denotes the radius of the neighbourhood in which the L-function considers point counts. For $r = 0.05$ the L-function displays negligible auto-correlation at lag $\tau = 200$ for both simulation schemes, see Figure 3. For larger length scales $r = 0.1$ N-fold way simulated tempering compares less favourable as the L-function indicates negligible auto-correlation at about $\tau = 200$ in contrast to $\tau = 150$ in the competing scheme, see Figure 4. However, this comes at the cost of having 10 auxiliary densities in the standard scheme compared to only two auxiliary densities in the N-fold way simulated tempering schedule.

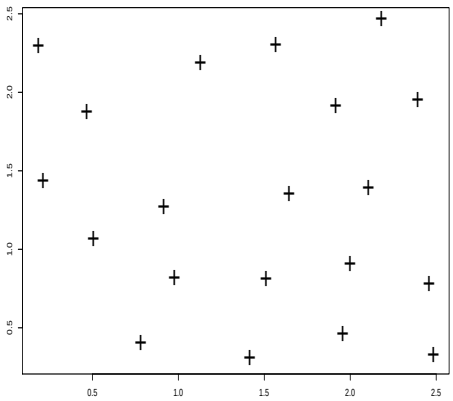

Fig. 1. A sample of a Strauss process on the window $[0, 2.5]^2$ with parameters $\lambda = 1000, \gamma = 10^{-5}$ and $R = 0.45$.

The fact that the L-function takes longer to decay to zero is probably due to the gridsize. In our scheme the grid has square cells of sidelength 0.78125. Therefore the neighbour count $S_R^B(\mathbf{x})$ of a point pattern $\mathbf{x}$ does not change if a point $\xi \in \mathbf{x}$ changes its location within the same cell.

We restricted our simulation experiments to simulated tempering and did not exploit a parallel computing based implementation. The fact that the N-fold way scheme relies on a smaller number of chains will be less of an advantage if using parallel tempering implemented on multiple processing units.

Finally, we considered a Strauss process whose interaction function g is a step function. The interaction in a (modified) Besag lattice process is based on cell centres. For the (modified) Strauss lattice process this leads to existence of point patterns $\mathbf{x}$ such that $S_R(\mathbf{x}) = 0$ but $S_R^B(\mathbf{x}) > 0$ and vice versa. This, in turn, leads to an increased rejection rate when moving from one level to another. We would expect this problem to be less pronounced for a smooth interaction function, thus leading to a better mixing N-fold way simulated tempering algorithm.

References

[1] Y.P. Atchade and J. Liu. The Wang-Landau algorithm for Monte Carlo computation in general state spaces. Technical Report, 2004.

[2] J. Besag, R. Milne and S. Zachary. Point process Limits of lattice processes. Journal of Applied Probability, 19:210-216,1982.

[3] A.B. Bortz, M.H. Kalos and J.L. Lebowitz. A new algorithm for Monte Carlo

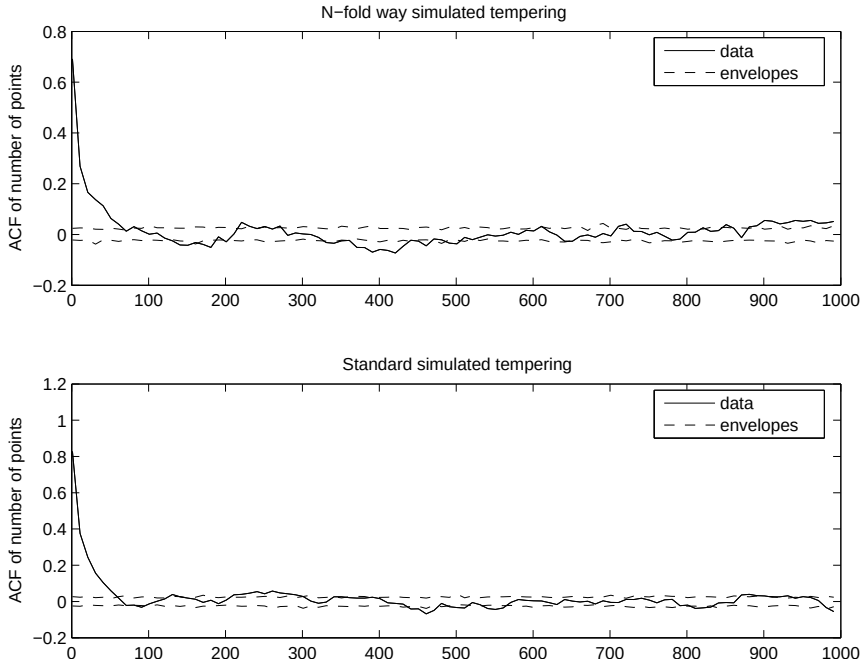

Fig. 2. The auto-correlation function for the number of points for the two simulated tempering schemes.

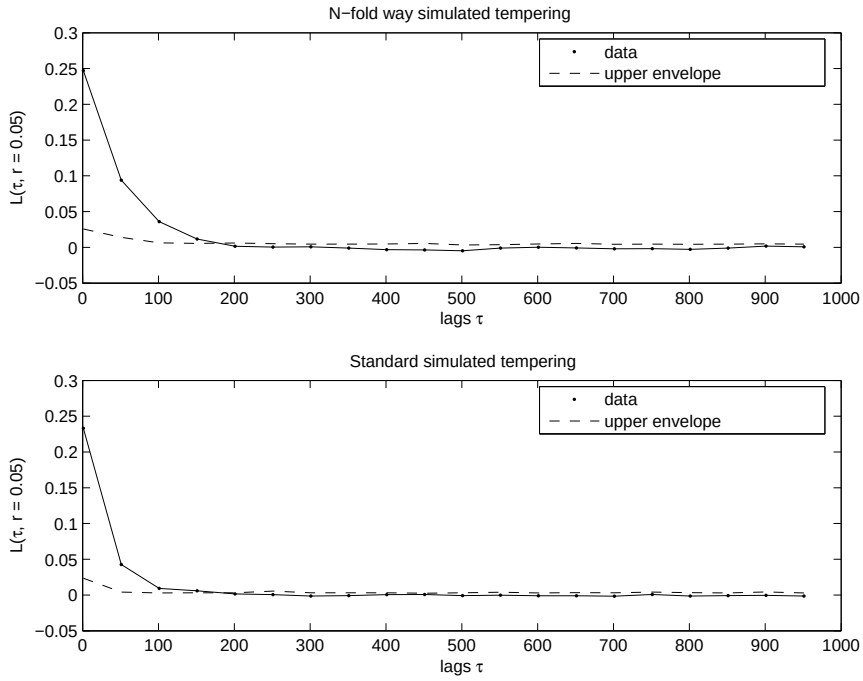

Fig. 3. The L-function at length scale $r = 0.05$ for both simulated tempering schemes. The dashed line is the 99%-confidence envelope.

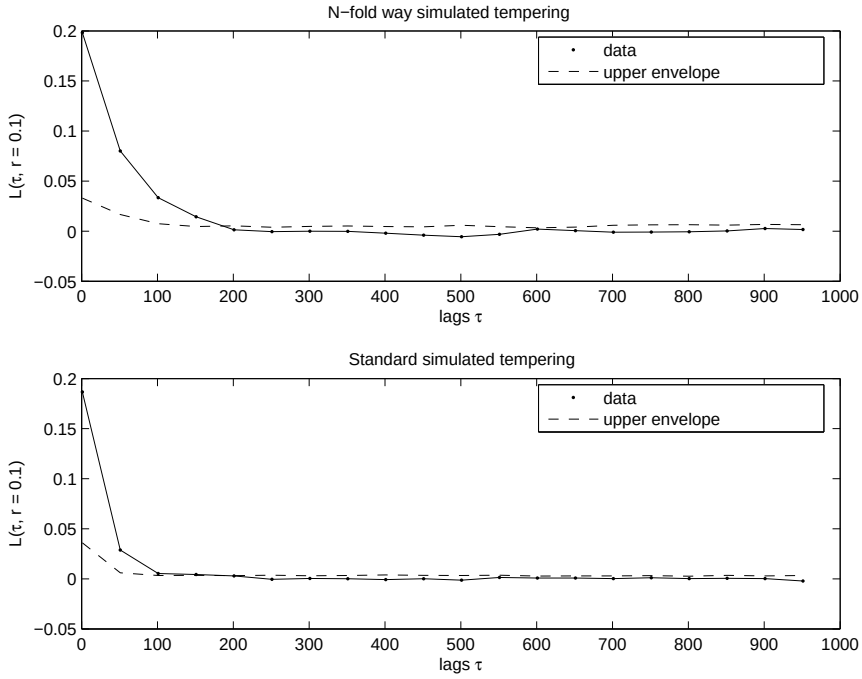

Fig. 4. The L-function at length scale $r = 0.1$ for both simulated tempering schemes. The dashed line is the 99%-confidence envelope.

simulation of Ising spin systems. Journal of Computational Physics, 17:10–18, 1975.

[4] O. Cappé, C.P. Robert and T. Rydén. Reversible jump MCMC converging to birth-and-death MCMC and more general continuous time samplers. Journal of the Royal Statistical Society, Series B,65:679-700, 2002.

[5] P. Clifford and G. Nicholls. Comparison of birth-and-death and Metropolis-Hastings Markov chain Monte Carlo for the Strauss process. Department of Statistics, Oxford University. Manuscript, 1994.

[6] P.J. Diggle. Statistical Analysis of Spatial Point Patterns. Academic Press. New York, 1983.

[7] A.E. Gelfand and A.F.M. Smith. Sampling based approaches to calculating marginal densities. Journal of the American Statistical Association, 85:398–409. 1990.

[8] C. Geyer. Markov chain Monte Carlo maximum likelihood. In P. Keramidas, editor, Computer Science and Statistics, Proceedings of the 23rd Interface, 156-163. Interface Foundation, Fairfax Station, 1991.

[9] C. Geyer and J. Møller. Simulation procedures and likelihood inference for spatial birth and death processes. Scandinavian Journal of Statistics, 21:359-373. 1994.

[10] C.J. Geyer and E.A. Thompson. Annealing Markov Chain Monte Carlo with applications to ancestral inference. Journal of the American Statistical Association, 90:909-920.1995.

[11] M. Goulard, P. Grabarnik and A. Särkkä. Parameter estimation for marked Gibbs point processes through the pseudo-likelihood method. Scandinavian Journal of Statistics, 23:365–379,1996.

[12] D. Higdon, H. Lee and Z. Bi. A Bayesian approach to characterizing uncertainty in inverse problems using coarse and fine scale information. IEEE Transactions on Signal Processing, 50:389-399, 2002.

[13] M.A. Hurn and C. Jennison. A study of simulated annealing and a revised cascade algorithm for image reconstruction. Statistics and Computing. 5:175-190, 1995.

[14] D. Kandel, E. Domany and A. Brandt, Simulations without critical slowing down: Ising and three-state Potts models. Physical Review, Series B. 40: 330-344, 1989.

[15] W. S. Kendall and J. Møller. Perfect simulation using dominating processes on ordered spaces, with application to locally stable point processes. Advances in Applied Probability, 32: 844-865, 2000.

[16] W.S. Kendall and R.G. Wilson. Ising models and multi-resolution quad trees. Advances in Applied Probability. 35: 96-122, 2003.

[17] M.N.M. van Lieshout and R.S. Stoica. Perfect simulation for marked point processes. To appear in Computational Statistics and Data Analysis, 2006.

[18] J. Liu and C. Sabatti. Generalized Gibbs Sampler and Multigrid Monte Carlo for Bayesian Computation. Biometrika, 87:353-369, 2000.

[19] E. Marinari and G. Parisi. Simulated tempering: a new Monte Carlo scheme. Europhysics Letters, 19: 451-458, 1992.

[20] A. Sokal. Monte Carlo Methods in Statistical Mechanics: Foundations and New Algorithms. Cours de Troisieme Cycle de la Physique en Suisse Romande. 1989

[21] D.J. Strauss. A model for clustering. Biometrika, 63:467-475, 1975.

[22] F. Wang and D.P. Landau. Efficient, multiple-range random walk to calculate the density of states. Physical Reviews Letters, 86: 2050-2053. 2001.