NORGES TEKNISK-NATURVITENSKAPELIGE UNIVERSITET

A FAST AND EXACT SIMULATION ALGORITHM FOR GENERAL GAUSSIAN MARKOV RANDOM FIELDS

by

Håvard Rue

PREPRINT
STATISTICS NO. 8/1999

NORWEGIAN UNIVERSITY OF SCIENCE AND TECHNOLOGY
TRONDHEIM, NORWAY

This report has URL http://www.math.ntnu.no/preprint/statistics/1999/S8-1999.ps
Håvard Rue has homepage: http://www.math.ntnu.no/~hrue
E-mail: hrue@math.ntnu.no
Address: Department of Mathematical Sciences, Norwegian University of Science and Technology, N-7491 Trondheim, Norway.

A Fast and Exact Simulation Algorithm for General Gaussian Markov Random Fields

HÅVARD RUE

DEPARTMENT OF MATHEMATICAL SCIENCES

NTNU, NORWAY

FEBRUARY 23, 1999

SUMMARY

This paper presents a fast and exact simulation algorithm for a general Gaussian Markov Random Field (GMRF) defined on a lattice. For a 5×5 neighborhood, the algorithm has initialization cost of $4n^2$ flops where n is the number of sites in the lattice and exact samples are then available using $4n\sqrt{n}$ flops. Conditioned samples on k constraints costs only $3nk$ extra flops. The algorithm is easy to generalize to non-rectangular lattices, compute the log-likelihood of the sample nearly for free and runs very efficient on the computer.

An exact and fast simulation algorithm for unconditional and conditional GMRF additional to easy access to the log-likelihood, makes GMRF (even more) computational convenient. GMRF has a potential use when modeling non-stationary spatial fields and is commonly used in hierarchical spatial or spatio-temporal Bayesian models analyzed using Markov chain Monte Carlo.

KEYWORDS: Conditional Autoregressive models, Gaussian Markov random field, Markov chain Monte Carlo.

ADDRESS FOR CORRESPONDENCE: H. Rue, Department of Mathematical Sciences, The Norwegian University for Science and Technology, N-7491 Trondheim.

E-MAIL: Havard.Rue@math.ntnu.no

ACKNOWLEDGMENTS: The author thanks Arne Martinsen, Håkon Tjelmeland and Erik Bølviken for fruitful and stimulating discussions, and the EU-TMR project on Spatial Statistics (ERB-FMRX-CT960095) for support and inspiration.

1 INTRODUCTION

This paper presents a fast and exact algorithm for simulating a general Gaussian random fields with a Markov property (GMRF), also known as conditional auto-regressions. Besag & Kooperberg (1995) give references to GMRF in various applications. The algorithm depends only on the conditional independence structure induced by the Markov property and no assumptions (apart from positive definiteness) are required on the interaction terms. On a $n_1 \times n_1$ lattice with n variables and a 5×5 neighborhood, the algorithm has initialization cost of $4n^2$ flops and then exact samples are available using only $4n\sqrt{n}$ flops. Conditioning on k observed variables or linear combinations of variables, only costs $3nk$ extra flops. The simulation algorithm can be extended to GMRF defined on non-rectangular lattices with essentially the same computational cost. Further, the likelihood can be computed for the generated sample nearly for free.

The presented simulation algorithm makes general GMRF even more computational convenient in important applications. Hierarchical spatial and spatio-temporal Bayesian models often require simulation based inference using Markov chain Monte Carlo (MCMC) and a GMRF is a natural choice for one layer in the model due to its Markov properties, see for example Wikle, Berliner & Cressie (1998). A fast and exact sampling algorithm for general GMRF will be useful as it may allow for fast and large block updates of the GMRF in the MCMC algorithm. Further, the inclusion of external covariates into spatial and spatio-temporal Gaussian models through the covariance function is troublesome. Using hierarchical modeling and GMRFs, covariates can be introduced directly into the conditional densities while preserving its Markov property. The presented simulation algorithm can then be used to produce exact unconditional and conditional samples. Since the normalization constant is readily available and the simulation algorithm is general, GMRF is a natural candidate to model non-stationary spatial fields, a topic yet to be explored.

The plan of the paper is as follows. After some preliminary definitions in Section 2, we present our algorithm in Section 3 and discuss generalizations in Section 4. We present an example in Section 5 and conclude in Section 6.

2 PRELIMINARIES

Define the Gaussian random field (GRF) $\boldsymbol{x} = \{x_{ij}, (i,j) \in \Lambda\}$ on the $n = n_1 \times n_1$ lattice Λ. Denote by $\pi(\boldsymbol{x})$ the Gaussian joint density of $\boldsymbol{x}$ and assume $\boldsymbol{x}$ has zero mean. The GRF has a Markov property iff

$$\pi(x_{ij} \mid x_{kl}, (k,l) \in \Lambda \backslash_{ij}) = \pi(x_{ij} \mid x_{kl}, (k,l) \in \partial_{ij})$$

where ∂_{ij} is the neighborhood of (i,j). A GRF with a Markov property is a Gaussian Markov random field (GMRF). Typical, ∂_{ij} is a set of neighboring sites, for example

$$\partial_{ij} = \{(k,l) \neq (i,j) \ : \ \|(k,l) - (i,j)\| \leq d\}$$

where $\|\cdot\|$ is some norm. Common neighborhoods in the nearest (in Euclidean norm) have 4, 8 and 12 sites, or the nearest 24 using the maximum-norm. We are mainly interested in the neighborhood defined as the 5×5 window centered at each site, but assume a general $(2h+1)\times(2h+1)$ neighborhood throughout the paper.

Let $\boldsymbol{Q}$ be the $n \times n$ precision matrix (the inverse covariance matrix) for $\boldsymbol{x}$. A Markov property of a GRF induce a band-structure in the precision matrix, as follows:

$$x_{ij} \perp x_{kl} \mid \boldsymbol{x}\backslash_{ij,kl} \quad \Longleftrightarrow \quad Q_{ij,kl} = 0, \tag{1}$$

meaning that x_{ij} and x_{kl} are conditional independent given the others iff the corresponding element in $\boldsymbol{Q}$ is zero. Thus for a 5×5 neighborhood system, (1) implies that only 25 elements in $Q_{ij,kl}$ are non-zero for each ij, and these corresponds to x_{ij}, $x_{i+1,j}$, $x_{i+2,j}$, $x_{i,j-1}$ and $x_{i,j-2}$ and so on. Appropriate changes at the boundary is needed. Hence, $\boldsymbol{Q}$ is a sparse block pentadiagonal matrix or a band matrix. We will from now treat $\boldsymbol{Q}$ as a band matrix with bandwidth $b_w = hn_1 + h$ (although most of the bands are zero), as the appropriate numerical algorithms needed in the simulation algorithms is designed for band matrices.

3 The Algorithm

Based on the preliminaries in Section 2 we can now derive our exact simulation algorithm. We first start by noting that computing the Cholesky factorization of the band matrix $\boldsymbol{Q}$ (with bandwidth b_w)

$$\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T \tag{2}$$

is a standard problem in numerics. By Theorem 4.3.1 in Golub & van Loan (1996), $\boldsymbol{L}$ has lower bandwidth b_w (and is zero above the diagonal) and can be computed in nb_w^2 flops, see Golub & van Loan (1996, Sec. 4.3.5) for details and algorithm. Note that $\boldsymbol{L}$ require nb_wU bytes of storage where U is the number of bytes needed to store one float. In the next step we let $\boldsymbol{z}$ be a vector of independent standard Gaussian variates, and note by direct calculation that $\boldsymbol{x}$ defined by

$$\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{z} \tag{3}$$

has mean zero and the desired covariance matrix $\boldsymbol{Q}^{-1}$. Again, solving (3) is a standard problem in numerics and the solution can be computed in $2nb_w$ flops using band back-substitution, see again Golub & van Loan (1996, Sec. 4.3.2) for details and algorithm.

We need to compute the band Cholesky factorization in (2) only once, and repeated samples are then available by solving (3).

To implement the algorithm, we make use of high-quality algorithms `dpbtf2` (or `dpbtrf`) and `dtbsv` in the public-domain `Lapack`-library written in Fortran (Anderson et al. 1995) to compute the band Cholesky decomposition in (2) and solve by back-substitution the linear system in (3), respectively.

Note that the speed of the algorithm depend on the size of the neighborhood only though the band-width. Further, there are no assumptions of the coefficients in the $\boldsymbol{Q}$-matrix (apart from being positive definite), so the algorithm is general.

I agree that the algorithm seems rather trivial, but (unfortunately) sometimes solutions to problems are!

4 GENERALIZATIONS OF THE ALGORITHM

I will now discuss some generalizations of the simulation algorithm to conditional simulation, simulation on a non-rectangular lattice, and the computation of the log-likelihood.

4.1 CONDITIONAL SIMULATION

I consider only the general case where we want to generate samples from π conditioned on $\boldsymbol{A}\boldsymbol{x}+\boldsymbol{\epsilon} = \boldsymbol{b}$ where $\boldsymbol{A}$ is a $k \times n$ matrix with rank k, but note that simplifications for special cases exists. Let the Gaussian noise $\boldsymbol{\epsilon}$ has zero mean and known covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{\epsilon}}$, and define $\widetilde{\boldsymbol{x}} = (\widetilde{\boldsymbol{x}}_1^T, \widetilde{\boldsymbol{x}}_2^T)^T = (\boldsymbol{x}^T, (\boldsymbol{A}\boldsymbol{x}+\boldsymbol{\epsilon})^T)^T$ where $\widetilde{\boldsymbol{x}}$ have dimension $n+k$. Let $\widetilde{\boldsymbol{\Sigma}}$ denote the covariance matrix of $\widetilde{\boldsymbol{x}}$. It is easy to simulate $\widetilde{\boldsymbol{x}}$ unconditionally by sampling $\boldsymbol{x}$ and $\boldsymbol{\epsilon}$, and then use the definition $\widetilde{\boldsymbol{x}} = (\boldsymbol{x}^T, (\boldsymbol{A}\boldsymbol{x}+\boldsymbol{\epsilon})^T)^T$. A conditional sample of $\widetilde{\boldsymbol{x}}_1 \mid \widetilde{\boldsymbol{x}}_2 = \boldsymbol{b}$ can be obtained using the following identity

$$\widetilde{\boldsymbol{x}}_1 \mid \widetilde{\boldsymbol{x}}_2 = \boldsymbol{b} \quad \stackrel{d}{=} \quad \widetilde{\boldsymbol{x}}_1 - \widetilde{\boldsymbol{\Sigma}}_{12}\widetilde{\boldsymbol{\Sigma}}_{22}^{-1}(\widetilde{\boldsymbol{x}}_2 - \boldsymbol{b}) \tag{4}$$

where $\widetilde{\boldsymbol{x}}$ on the right hand side is an unconditional sample. A direct calculation shows that (4) is correct. Eq. (4) is commonly refered to as conditioning using Kriging, see Cressie (1993, Sec. 3.6.2). By expanding the right hand side of (4), we obtain

$$\boldsymbol{x} - \boldsymbol{Q}^{-1}\boldsymbol{A}^T\left(\boldsymbol{A}\boldsymbol{Q}^{-1}\boldsymbol{A}^T + \boldsymbol{\Sigma}_{\boldsymbol{\epsilon}}\right)^{-1}(\boldsymbol{A}\boldsymbol{x} + \boldsymbol{\epsilon} - \boldsymbol{b})\,. \tag{5}$$

We need to compute $\boldsymbol{v} = \boldsymbol{Q}^{-1}\boldsymbol{A}^T$, which requires solving $\boldsymbol{Q}\boldsymbol{v}_i = (\boldsymbol{A}^T)_i$ for each of the k columns. Since the band Cholesky decomposition is already available as $\boldsymbol{Q} = \boldsymbol{L}\boldsymbol{L}^T$, we solve $\boldsymbol{Q}\boldsymbol{v}_i = (\boldsymbol{A}^T)_i$ by a forward substitution $\boldsymbol{L}\boldsymbol{u}_i = (\boldsymbol{A}^T)_i$ and a back-substitution $\boldsymbol{L}^T\boldsymbol{v}_i = \boldsymbol{u}_i$.

The remaining parts of (5) are now straight forward. To save work, we compute $\boldsymbol{v}$ and then $\boldsymbol{A}\boldsymbol{v}$ (using $4knb_w$ flops) and invert the symmetric $k \times k$ matrix (using k^3 flops) only once. For each sample we need to compute $\boldsymbol{A}\boldsymbol{x} + \boldsymbol{\epsilon} - \boldsymbol{b}$, pre-multiply with the matrix found in the initialization, negate and add $\boldsymbol{x}$ (in total $3nk$ flops).

4.2 NON-RECTANGULAR LATTICE

I will now relax the assumption that $\boldsymbol{x}$ is defined on a rectangular lattice. Suppose the region of interest is a non-rectangular lattice Λ. The precision matrix will not have the required band-structure required for the fast simulation algorithm. However, define Λ' as the smallest rectangular lattice covering Λ and extend $\boldsymbol{x}$ to $\boldsymbol{x}'$ by adding independent standard Gaussian variates for each site in $\Lambda' - \Lambda$, then the precision matrix for $\boldsymbol{x}'$ will have the required band structure. Further, $\boldsymbol{x}'$ restricted to Λ have the desired marginal density.

4.3 Likelihood

Likelihood based inference and testing, computation of acceptance rates in MCMC from GMRF, all depends on the log-likelihood which is easily available for general GMRF. First, if we simulate a sample $\boldsymbol{x}$, then note that $\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x}$ equals $\boldsymbol{z}^T\boldsymbol{z}$, and $|\boldsymbol{Q}|^{1/2}$ equals $\prod_i L_{ii}$. Hence, the log-likelihood for the sample $\boldsymbol{L}^T\boldsymbol{x} = \boldsymbol{z}$ becomes

$$\text{log-likelihood}(\boldsymbol{x}) = -\frac{n}{2}\log 2\pi + \sum_i \log(L_{ii}) - \frac{1}{2}\boldsymbol{z}^T\boldsymbol{z}$$

and the computation of the log-likelihood does not require (nearly) any extra computational costs compared to sampling $\boldsymbol{x}$. If $\boldsymbol{x}$ is given, we need to compute the alternative expression $\boldsymbol{x}^T\boldsymbol{Q}\boldsymbol{x}$. Using the argument as in Section 4.2, we can compute the log-likelihood for a general lattice.

5 An Example

To give an idea of the CPU costs for the algorithm, I will present results for my current implementation of the algorithm on a Sun Ultra 2 Model 1296 with a 296 MHz SUNW UltraSPARC-II processor and a 100×100 lattice.

With a 3×3 neighborhood the algorithm used 0.92 seconds for the first sample and then produced iid samples using 0.07 seconds each. (All timings include the evaluation of the log-likelihood for the sample.) When I increased the neighborhood to 5×5, the algorithm used 2.80 and 0.11 seconds, and for a 7×7 neighborhood 6.00 and 0.15 seconds.

Then I added 100 linear conditions requiring the sum of each of the 100 columns is zero. The first evaluation of the conditioning took 14.3, 19.9 and 25.8 seconds for 3×3, 5×5 and 7×7 neighborhoods, respectively. Then the following samples used only 0.08 seconds in the conditioning, for all three neighborhoods.

Figure 1 shows an unconditioned sample using a 5×5 neighborhood with coefficients chosen to approximate a Gaussian field with correlation function $\exp(-3d^2/20^2)$, where d is the Euclidean distance in pixels. The GMRF fits the Gaussian field surprisingly well and demonstrates that GMRF is indeed appropriate as a computational attractive approximate to Gaussian fields, refer to Tjelmeland & Rue (1999) for details and discussion along these lines.

My current implementation makes about 140 and 37 Megaflops on a 5×5 neighborhood, computing (mainly) the Cholesky-decomposition and solving the band linear system, respectively. To compare the numbers, the plain loop $y_i = y_i + \beta x_i$, makes on the same computer 180 and 195 Megaflops in C and Fortran compiled with SUN compilers using options `-fast -xO5`, respectively.

6 DISCUSSION

We have in this paper demonstrated that Gaussian Markov random fields can be sampled exactly using $\mathcal{O}(n\sqrt{n})$ flops with an initial costs $\mathcal{O}(n^2)$, and that a very efficient implementation of the sampler is possible using the `Lapack`-library. Further, the log-likelihood for the sample is easily available, the algorithm is trivial to extend to non-rectangular lattices, and we can obtain conditional samples efficiently using Kriging.

The example in Figure 1 shows that a GMRF is indeed capable of approximating a Gaussian fields with (relative) long correlation length, see Tjelmeland & Rue (1999) for further details.

The only negative aspect of the algorithm is the memory-requirement of $\mathcal{O}(n\sqrt{n})$. This prohibits the use of the algorithm for large lattices. For a 256×256 lattice, the algorithm require about 268 MBytes of memory (using 8 Bytes to store a float), still feasible for my computer, but a 512×512 lattice requires about 2.1 GBytes which is too much. Rue, Marthinsen & Husby (1999) solves this problem by developing a split and conquer algorithm which runs with near optimal speedup on parallel computers, where the memory requirement is $\mathcal{O}(n)$ and the computational costs is about $\mathcal{O}(n^{1.8})$. However, their algorithm is intended for large fields and runs much slower for problems where the current algorithm is feasible, see Rue et al. (1999) for details.

I end this paper by pointing out that Lavine (1998) also reports an $\mathcal{O}(n^2)$ (unconditional) simulation algorithm for GMRF with 4 nearest neighbors and invariant interactions (the elements in the $\boldsymbol{Q}$-matrix) by running a special Bayesian dynamic model. However, for his case we can do even better by making use of the fast block-tridiagonal solver of Swarztrauber (1974) and derive (essentially) an $\mathcal{O}(n\log(n))$ using the split and conquer algorithm in Rue et al. (1999). However, algorithms for special cases in terms of values of the interactions are usually of less interest compared to a general algorithm.

REFERENCES

ANDERSON, E., BAI, Z., BISCHOF, C., DEMMEL, J., DONGARRA, J., CROZ, J. D., GREENBAUM, A., HAMMARLING, S., MCKENNEY, A., OSTROUCHOV, S., & SORENSEN, D. (1995). *LAPACK Users' Guide*, 2 edn, Philadelphia: Society for Industrial and Applied Mathematics.

BESAG, J. & KOOPERBERG, C. (1995). On conditional and intrinsic autoregression, *Biometrika* 82(4): 733–746.

CRESSIE, N. A. C. (1993). *Statistics for spatial data*, 2 edn, John Wiley, New York.

GOLUB, G. H. & VAN LOAN, C. F. (1996). *Matrix Computations*, 3 edn, Johns Hopkins University Press, Baltimore.

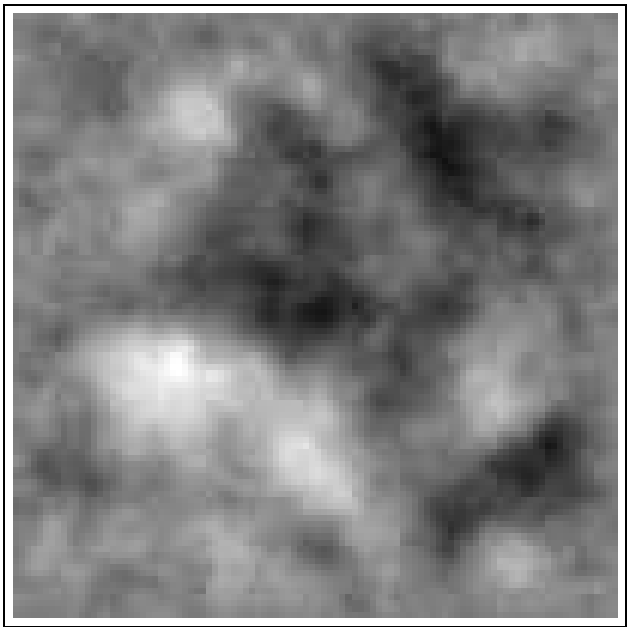

FIGURE 1: An exact sample from a 100×100 GMRF with a 5×5 neighborhood, where the coefficients are chosen to fit a Gaussian field with correlation function $\exp(-3d^2/20^2)$.

LAVINE, M. (1998). Another look at conditionally Gaussian Markov random fields, *Bayesian Statistics: Proceedings of the Sixth International Meeting Held in Valencia (Spain)*. to appear.

RUE, H., MARTHINSEN, A. & HUSBY, E. G. (1999). A split and conquer algorithm for exact simulation of general Gaussian Markov random fields, *Statistics no. 11*, Department of Mathematical Sciences, Norwegian University of Science and Technology, Trondheim, Norway.

SWARZTRAUBER, P. N. (1974). A direct method for the discrete solution of separable elliptic equations, *SIAM Journal of Numerical Analysis* 11(6): 1136–1150.

TJELMELAND, H. & RUE, H. (1999). Approximating Gaussian fields with Gaussian Markov random fields, *in progress*, Department of Mathematical Sciences, Norwegian University of Science and Technology, Trondheim, Norway.

WIKLE, C. K., BERLINER, L. M. & CRESSIE, N. A. (1998). Hierarchical Bayesian space-time models, *Environmental and Ecological Statistics* 5(2): 117–154.