Simulation of a Gaussian random vector: A propagative approach to the Gibbs sampler

C. Lantuéjoul, N. Desassis and F. Fouedjio

MinesParisTech

"Kriging and Gaussian processes for computer experiments"
CHORUS Workshop, 30.04.14

Outline

Presentation of the problem

– reminder and notation

– running Gibbs sampler on large Gaussian random vectors

Propagative version of the Gibbs sampler

– two remarks

– basic algorithm

– possible extensions

Applications

– simulation of a nonstationary Gaussian random field

– conditional simulation of a Cox process

Presentation of the problem

Reminder and notation

Let $Y_S = (Y_s, s \in S)$ be a standardized Gaussian random vector with covariance matrix C.

(Simple) kriging:

If $s \in S$ and $A \subset S$. The kriging of Y_s on Y_A is a linear estimator $Y_s^A = \sum_{\alpha \in A} \lambda_\alpha Y_\alpha$. The weights λ_α are chosen so as to minimize the estimation variance $Var\{Y_s - Y_s^A\}$. They are solutions of the linear system

$$\sum_{\beta \in A} \lambda_\beta C_{\alpha,\beta} = C_{\alpha,s} \qquad \alpha \in A$$

The estimation variance is

$$\sigma_s^{2A} = Var\{Y_s - Y_s^A\} = 1 - \sum_{\alpha \in A} \lambda_\alpha C_{\alpha,s}$$

Conditional distribution:

$Y_s | Y_A = y_A$ is normally distributed. More precisely, we have

$$Y_s | Y_A = y_A \sim \mathcal{N}\left(y_s^A, \sigma_s^{2A}\right)$$

Simulation of a Gaussian random vector using the Gibbs sampler

Problem:

Simulate the Gaussian vector Y_S iteratively.

Algorithm:

(i) reset y_S^c;

(ii) put $y_S^n = y_S^c$;

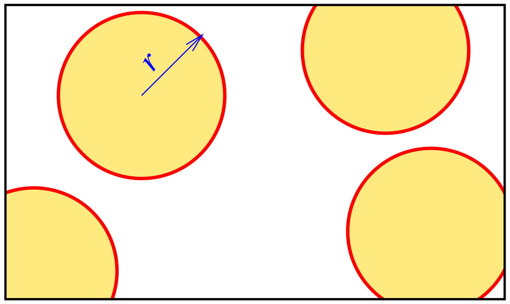
(iii) select $s \sim \mathcal{U}(S)$ *and generate* $y_s^n \sim \mathcal{N}\left(y_s^{S\setminus s}, \sigma_s^{2S\setminus s}\right)$;

(iv) put $y_S^c = y_S^n$ *and goto (ii).*

Problem:

When S is large, the kriging matrices cannot be inverted.

Simulation of a Gaussian random vector using the Gibbs sampler; approximate algorithm



To illustrate the approach, the components of S are depicted as the nodes of a square grid, which makes it possible to consider kriging neighbourhoods as balls $B_{s,r}$.

Algorithm:

(i) reset y_S^c;

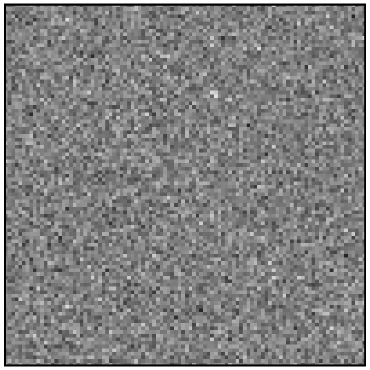
(ii) put $y_S^n = y_S^c$;

(iii) select $s \sim \mathcal{U}(S)$ *and generate* $y_s^n \sim \mathcal{N}\left(y_s^{B_{s,r}\setminus s}, \sigma_s^{2B_{s,r}\setminus s}\right)$;

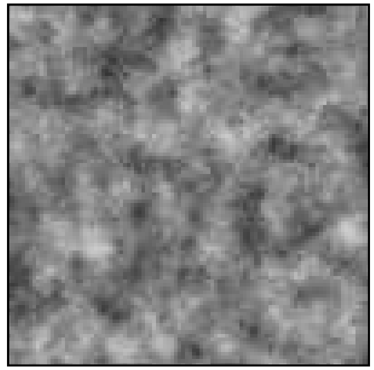
(iv) put $y_S^c = y_S^n$ *and goto (ii).*

Example (10000 components), spherical covariance (range 10)

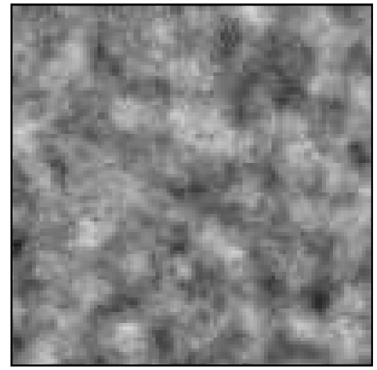
Neighbourhood radius of 15



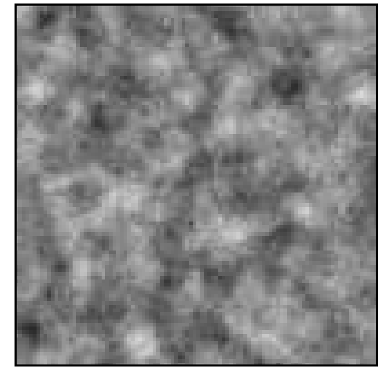
0



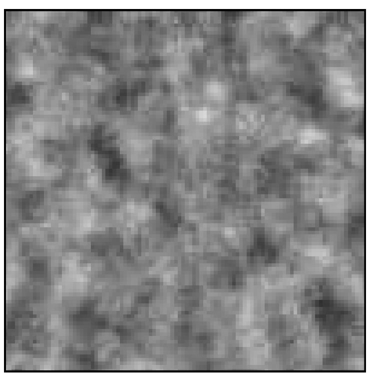
200



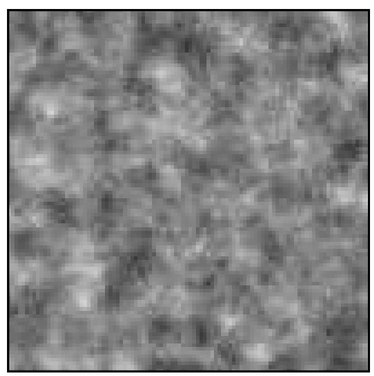
400



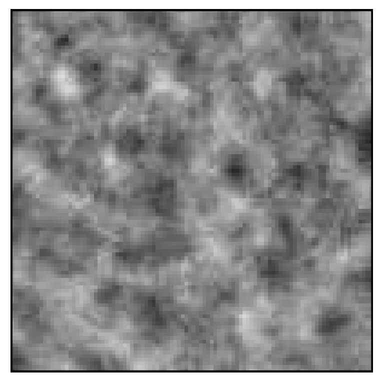
600



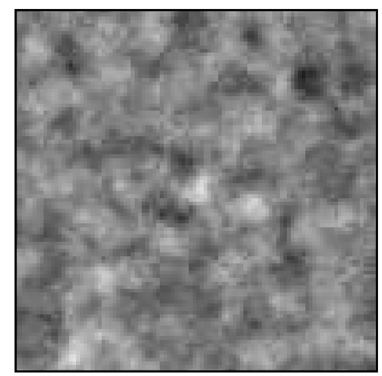
800



1000



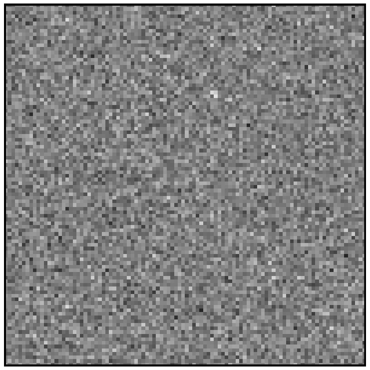
1200



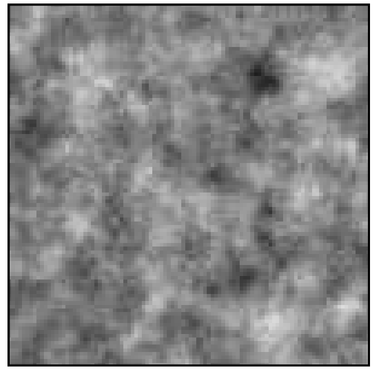
1400

Example (10000 components), spherical covariance (range 10)

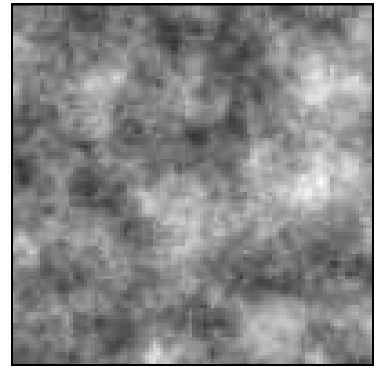
Neighbourhood radius of 5



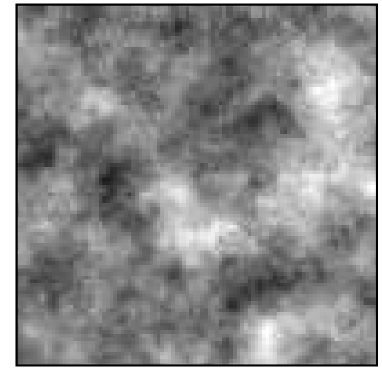
0



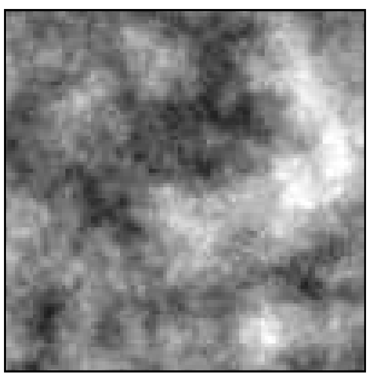
200



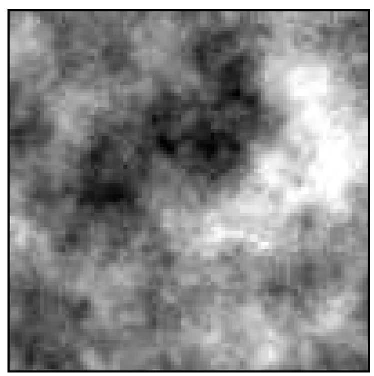
400



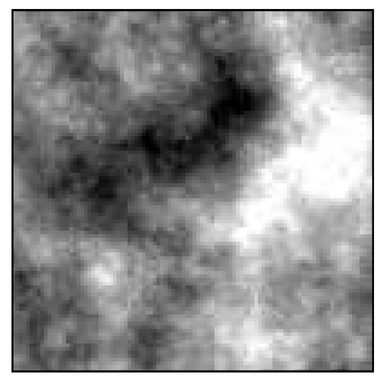
600



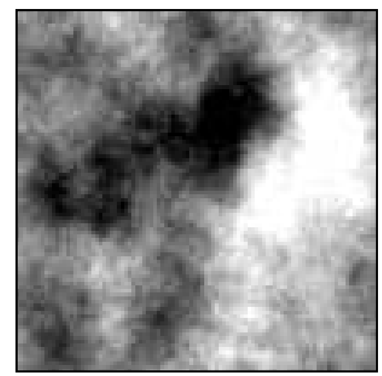
800



1000

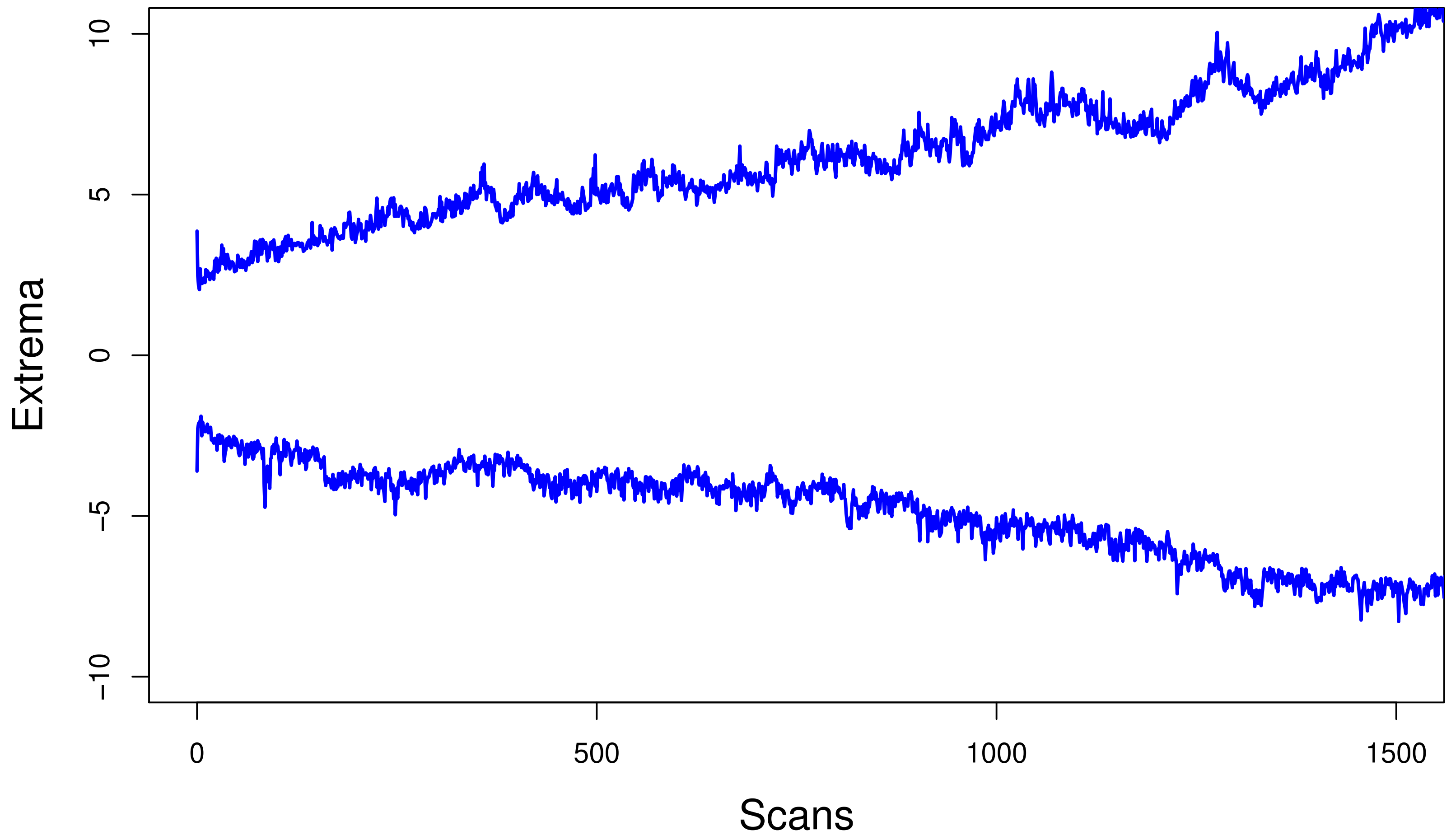


1200



1400

Extrema



Gibbs sampler
for Gaussian random vectors
A propagative version

Two useful remarks

Remark 1:

Let $Y_S \sim \mathcal{N}(0, C)$. The kriging estimate $y_s^{S\backslash s}$ and its estimation variance $\sigma_s^{2S\backslash s}$ can be expressed using the inverse C^{-1} of matrix C:

$$y_s^{S\backslash s} = -\sum_{t \neq s} \frac{C_{st}^{-1}}{C_{ss}^{-1}} y_t \qquad\qquad \sigma_s^{2S\backslash s} = \frac{1}{C_{ss}^{-1}}$$

Remark 2:

Galli and Gao (2001): If $X = C^{-1}Y$, then $X \sim \mathcal{N}(0, C^{-1})$.

Consequence:

As $(C^{-1})^{-1} = C$, the Gibbs sampler can be run exactly on X.

Idea:

Apply the exact Gibbs sampler to X and transport the results to Y using the relation $Y = CX$.

Running the Gibbs sampler on X

Remark:

$$X_s^{S\backslash s} = -\sum_{t \neq s} \frac{C_{st}}{C_{ss}} X_t = -\sum_{t \neq s} C_{st} X_t = X_s - Y_s$$

$$Var\{X_s - X_s^{S\backslash s}\} = \frac{1}{C_{ss}} = 1$$

Algorithm:

(i) reset x_S^c;

(ii) put $x_S^n = x_S^c$;

(iii) select $p \sim \mathcal{U}(S)$ *and generate* $x_p^n \sim \mathcal{N}(x_p^c - y_p^c, 1)$;

(iv) put $x_S^c = x_S^n$ *and goto (ii).*

Definition:

The point p is called a pivot.

Transporting the Gibbs sampler to Y

If $x_p^n \sim \mathcal{N}(x_p^c - y_p^c, 1)$, then $x_p^n = x_p^c - y_p^c + u$ with $u \sim \mathcal{N}$. Then

$$y_s^n = \sum_t C_{st} x_t^n = \sum_{t \neq p} C_{st} x_t^c \; + \; C_{sp} x_p^n$$

$$= y_s^c - C_{sp} x_p^c \; + \; C_{sp}(x_p^c - y_p^c + u) = y_s^c + C_{sp}(u - y_p^c)$$

Taking $s = p$, we obtain $y_p^n = u$, which finally gives

$$y_s^n = y_s^c + C_{sp}(y_p^n - y_p^c) \qquad s \in S$$

Algorithm:

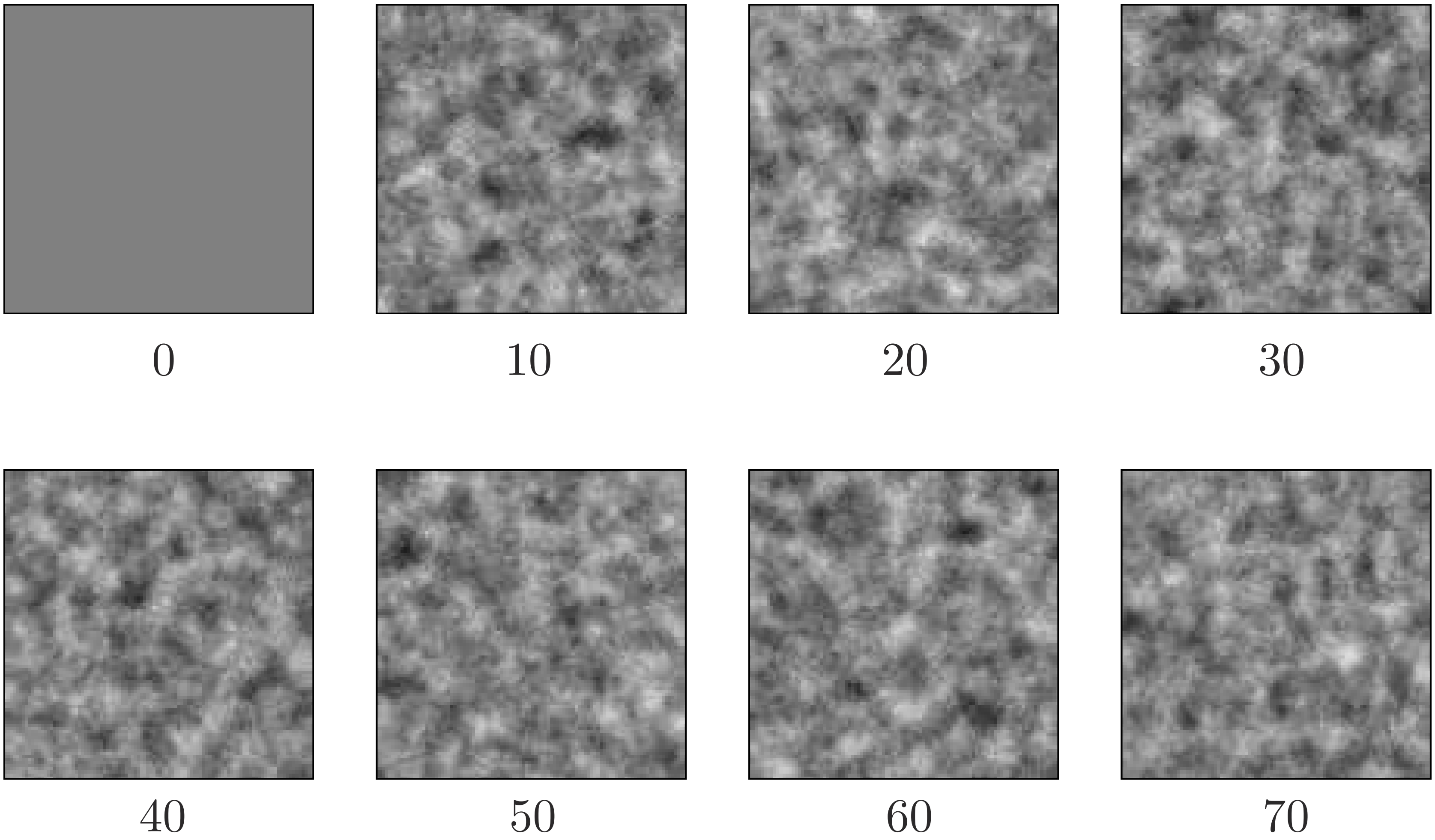
(i) reset y_S^c;

(ii) select $p \sim \mathcal{U}(S)$ *and generate* $y_p^n \sim \mathcal{N}$;

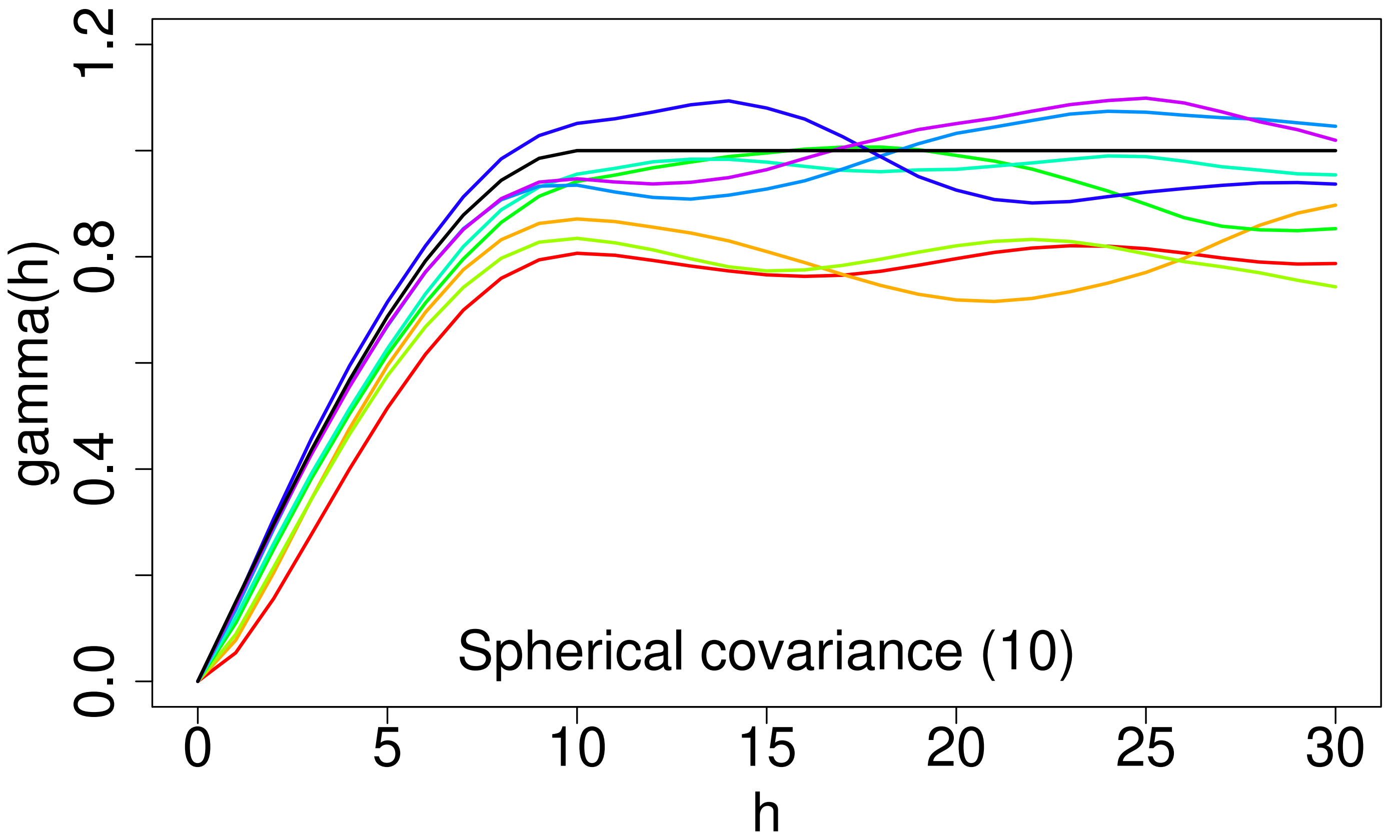
(iii) put $y_s^n = y_s^c + C_{sp}(y_p^n - y_p^c)$ *for each* $s \in S$;

(iv) put $y_S^c = y_S^n$ *and goto (ii).*

Example of a vector with 10000 components
Spherical covariance function with range 10



Variograms of simulations



Simulation variograms obtained after 1, 2, 3, 5, 7, 10, 15 and 20 scans. In black, the variogram model.

Generalizations

Generating the pivot value

y_p^n can be generated as a function of y_p^c, e.g. $y_p^n \sim \mathcal{N}(ry_p^c, 1-r^2)$. It is sometimes convenient to draw y^n close to y^c.

Blocking strategy

$$y_s^n = y_s^c + C_{sp}(y_p^n - y_p^c) \qquad \Longleftrightarrow \qquad y_s^n - C_{sp}y_p^n = y_s^c - C_{sp}y_p^c$$

Thus, kriging residuals are preserved by the propagative approach of the Gibbs sampler.

This remark remains valid when the pivot p is replaced by a family P of pivots.

(i) put $y^c = 0$;
(ii) select P *at random in* S *and generate* $y_P^n \sim \mathcal{N}(0, C_{PP})$;
(iii) put $y_s^n = y_s^c + y_s^{n,P} - y_s^{c,P}$ *for each* $s \in S$;
(iv) put $y^c = y^n$, *and goto (ii).*

Application
Simulation of a non-stationary Gaussian random field

Simulation of a nonstationary Gaussian random field

Let Z be a nonstationary Gaussian random field. It can be written as $m + \sigma Y$, where Y is a (non necessarily stationary), standardized Gaussian random field.

Problem:

Generate Z in the discrete domain S.

Notation:

Let C be the covariance matrix of Y_S.

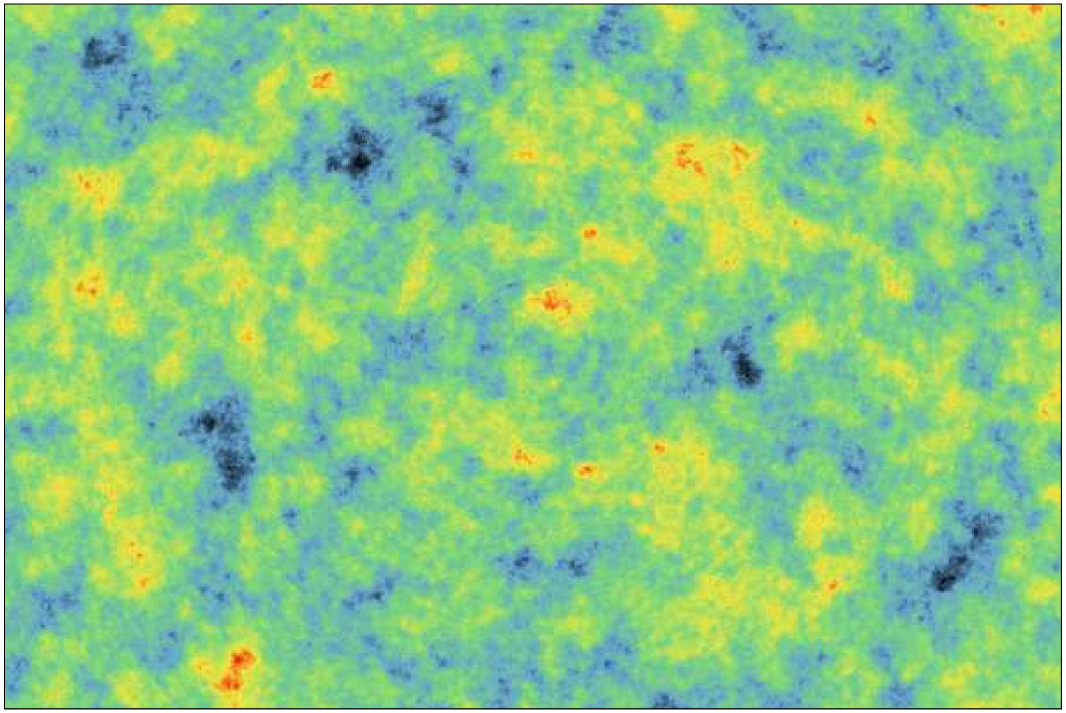
Algorithm:

(i) generate $y_S \sim \mathcal{N}(0, C)$*;*

(ii) put $z_s = m_s + \sigma_s y_s$ *for each* $s \in S$*.*

Example: point source model

$$C_{s,t} = \exp\bigl(-\lambda|s-t|\bigr)\ \exp\bigl(-\mu\bigl|e^{-\nu|s|} - e^{-\nu|t|}\bigr|\bigr)$$



$m = 0$ and $\sigma = 1$. $\lambda = 0.05$, $\mu = 5$ and $\nu = 0.0025$.
Simulation field 600×400.

Application

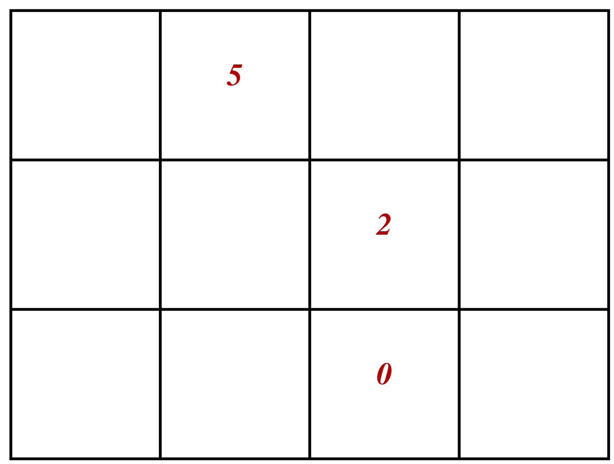
Conditional simulation of a Cox process

Motivation

A piece of land is covered with trees. It is partitioned into congruent plots.

The number of trees is known on a number of plots.

We want to predict the number of trees on each plot.

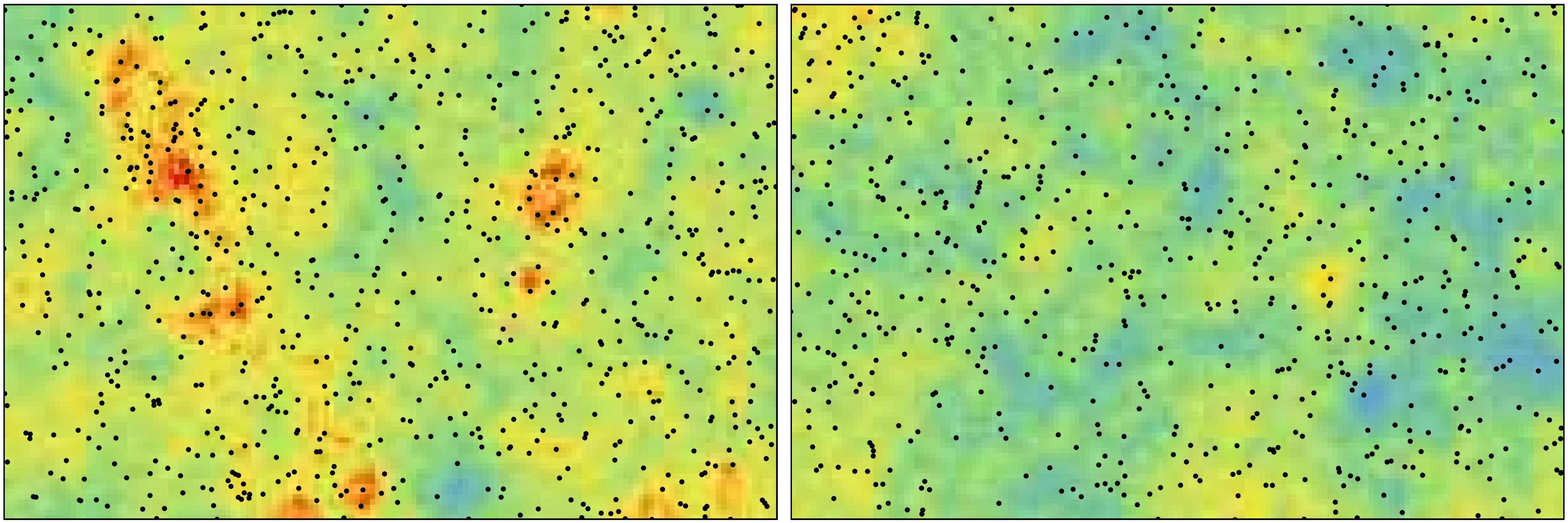


This can be done by averaging a set of conditional simulations. The spatial distribution of trees is modelled by a Cox process.

Cox process

Definition:

A Cox process is a Poisson point process with random intensity function.



Remark:

The random intensity function is also called potential.

Properties of the Cox process

Let $N_S = (N_s, s \in S)$ be the number of trees on all plots, and let Z_S be their potential.

Mean value:

$$E\{N_s\} = E\{Z_s\}$$

Covariance between plots:

$$Cov\{N_s, N_t\} = \begin{cases} Cov\{Z_s, Z_t\} & \text{if } s \neq t \\ Var\{Z_s\} + E\{Z_s\} & \text{if } s = t \end{cases}$$

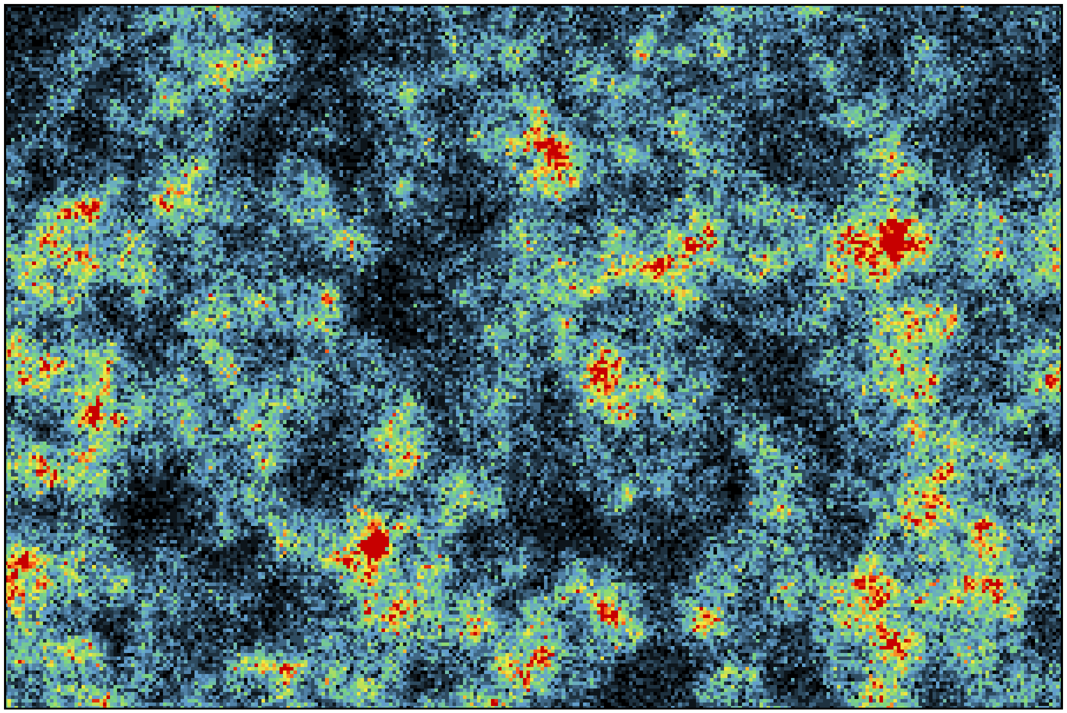
Conditional distribution:

Given the potential, the numbers of trees on differents plots are conditionally independent.

$$P\{N_S = n_S | Z_S = z_S\} = \prod_{s \in S} \exp(-z_s) \frac{z_s^{n_s}}{n_s!}$$

Cox-lognormal process

The potential is lognormally distributed, i.e. $Z_S = \exp(\mu + \sigma Y_S)$, where Y_S is a standardized Gaussian random vector (covariance matrix C).



$\mu = 1.364$ and $\sigma = 0.7$. $C_{s,t} = \exp(-|s-t|/15)$.
Simulation field 300×200 unit plots.

Conditional simulation of a Cox-lognormal process

Problem:

Generate N_S given $N_A = n_A$ for some $A \subset S$.

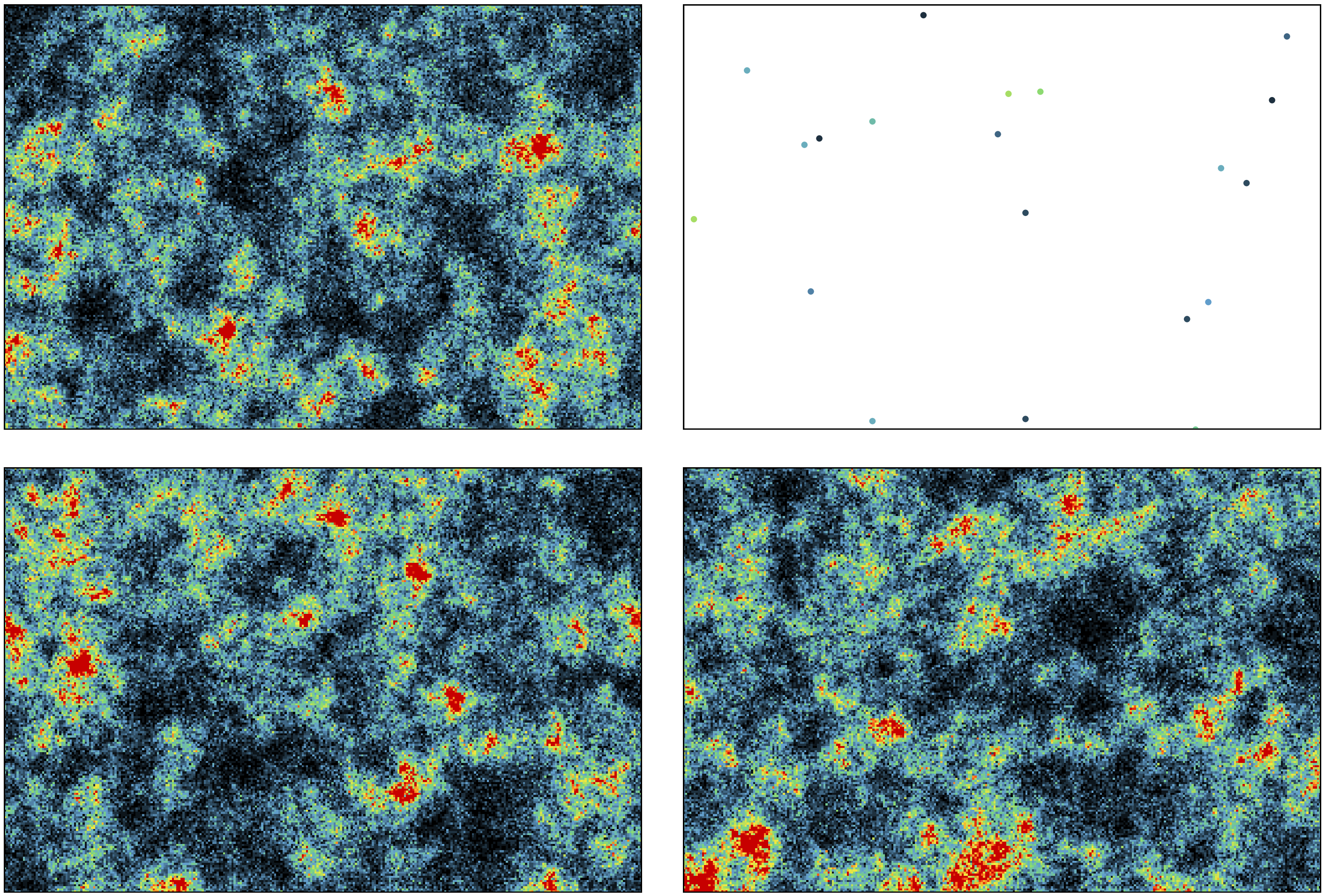
Idea:

At each iteration, candidate values are proposed for y_S using the propagative version of the Gibbs sampler. They are accepted or rejected using Metropolis-Hastings criterion.

Algorithm:

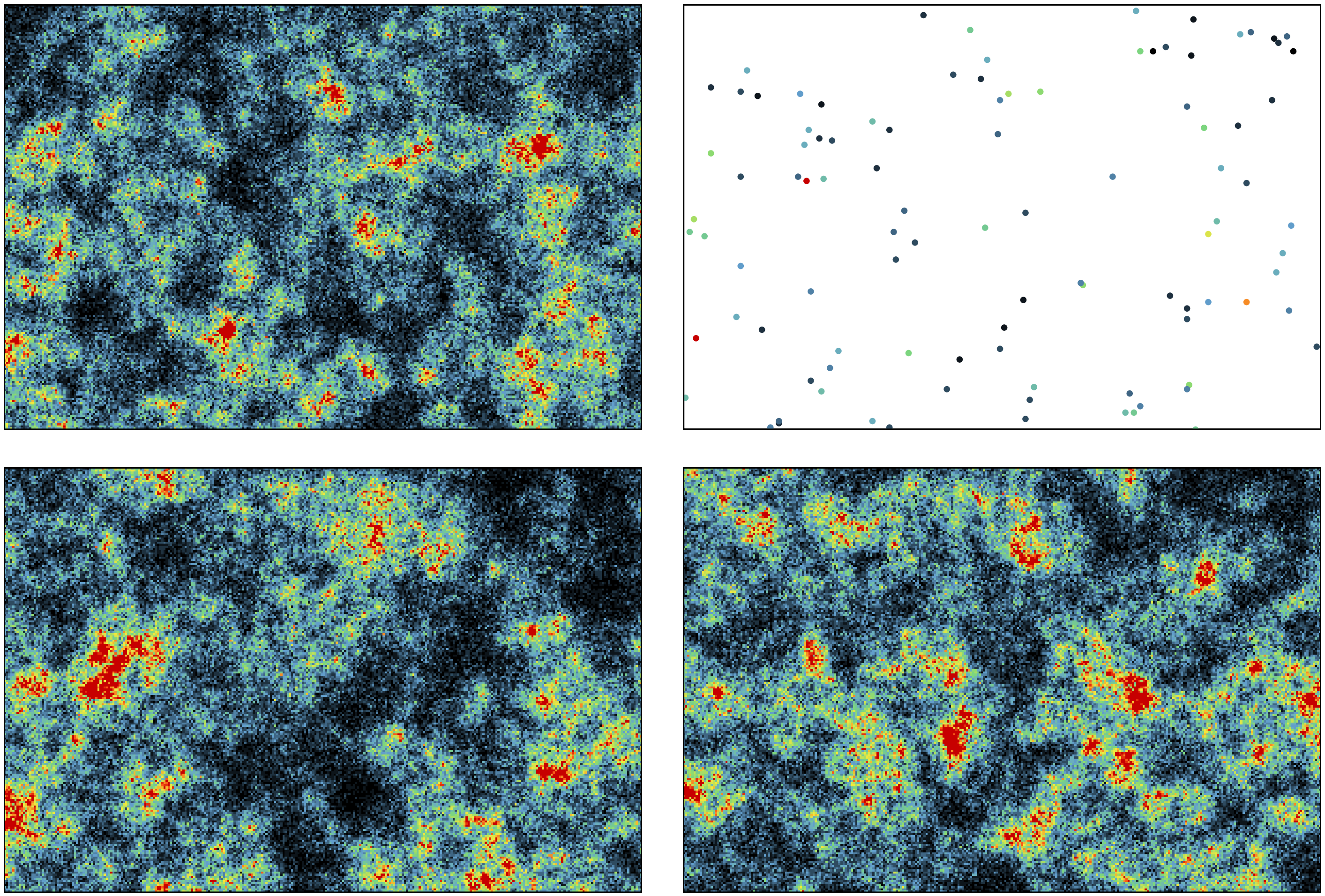
(i) put $y_S^c = 0$;
(ii) generate $p \sim \mathcal{U}(S)$ *and* $y_p^n \sim \mathcal{N}$;
(iii) put $z_A^c = e^{\mu + \sigma y_A^c}$, $z_A^n = z_A^c \, e^{\sigma C_{Ap}(y_p^n - y_p^c)}$, *and generate* $u \sim \mathcal{U}$;
(iv) if $u > p(n_A | z_A^n)/p(n_A | z_A^c)$, *then goto (ii);*
(v) put $y_S^c = y_S^c + C_{Sp}(y_p^n - y_p^c)$;
(vi) goto (ii).

20 conditioning data points



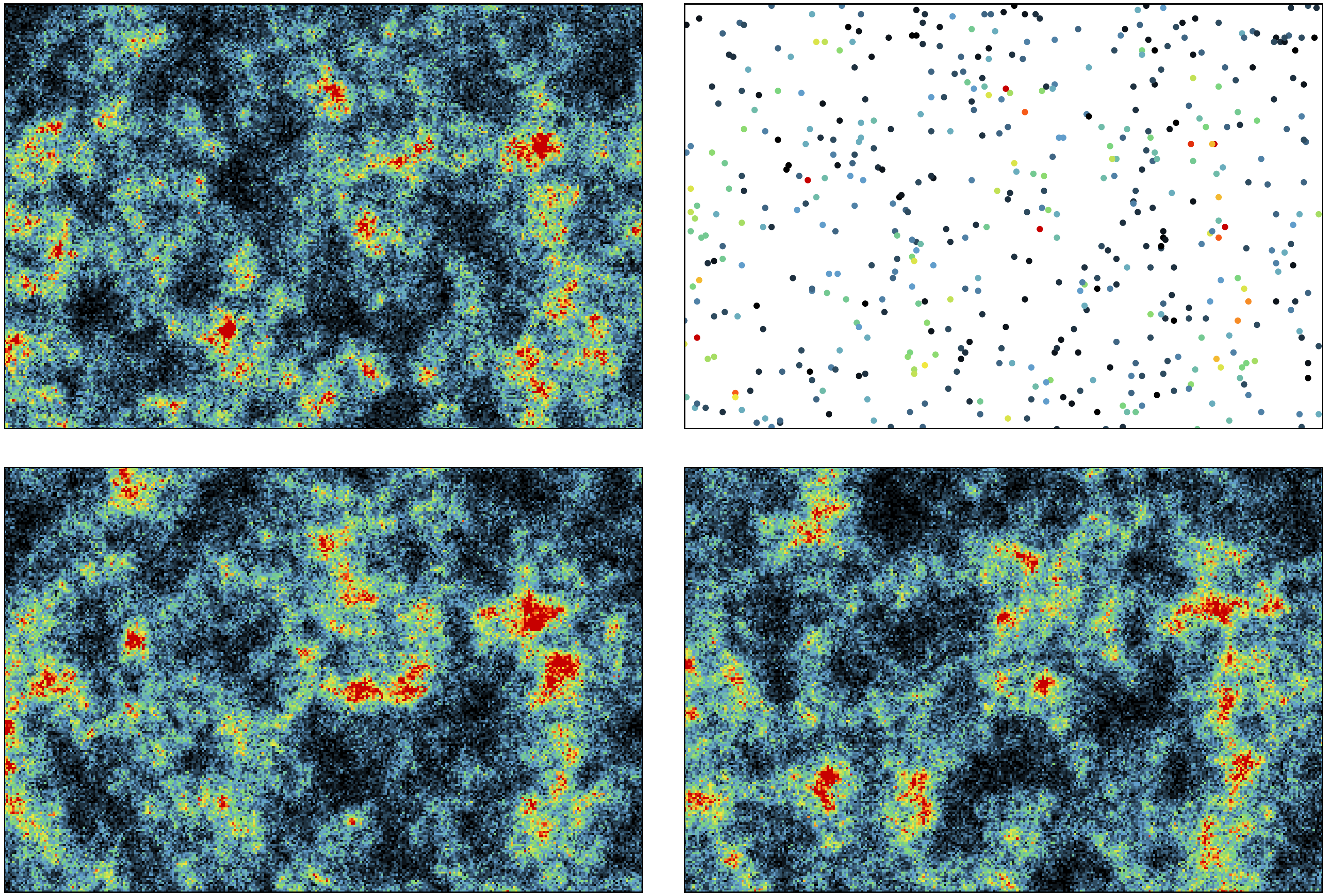
"Reality" (TL), conditioning data points (TR), and two conditional simulations (BL and BR)

100 conditioning data points



"Reality" (TL), conditioning data points (TR), and two conditional simulations (BL and BR)

500 conditioning data points



"Reality" (TL), conditioning data points (TR), and two conditional simulations (BL and BR)

Conclusions

An iterative algorithm, derived from the Gibbs sampler, has been proposed for simulating Gaussian random vectors.

At each iteration, a component is selected and randomly assigned a new value. This value is then linearly propagated to the other components.

Because it does not require the inversion of the covariance matrix, this algorithm does not incur the dimensionality problem of the standard Gibbs sampler.

This can be used to simulate random processes related to Gaussian random fields even if they are nonstationary or subject to soft constraints.

References

– Desassis N. et Lantuéjoul C. (2011) - "Simulation d'un vecteur gaussien: une approche propagative de l'chantillonneur de Gibbs". 43^{emes} Journées de Statistique, Tunis (Tunisie).

– Lantuéjoul C. and Dessassis N. (2012) - "Simulation of a Gaussian random vector: A propagative version of the Gibbs sampler". 9^{th} Int. Geostatistics Congress, Oslo (Norway).

– Fouedjio F. (2014) - Développement de modèles géostatistiques globaux non-stationnaires. Thèse MinesParisTech (à soutenir).

Rate of convergence

Let $Y^{(n)}$ the random vector generated at the n^{th} scan.

Systematic scan

If $Y^{(0)} \sim \mathcal{N}(0, C^{(0)})$, then $Y^{(n)} \sim \mathcal{N}(0, C^{(n)})$. Besides, we have

$$C^{(n)} - C = B^n \big(C^{(0)} - C\big) {}^t B^n$$

where $B = (Id - L)^{-1} U$. The matrices Id, L and U are respectively the identity matrix, the lower and upper parts of C

$$C = Id + L + U$$

If follows that the rate of convergence is governed by the square of the spectral radius of B.