

# Miscellaneous

## A propagative version of the Gibbs sampler

Christian Lantuéjoul & Thomas Romary

[christian.lantuejoul@mines-paristech.fr](mailto:christian.lantuejoul@mines-paristech.fr)

[thomas.romary@mines-paristech.fr](mailto:thomas.romary@mines-paristech.fr)



# Outline

- 1 Propagative version of the Gibbs sampler
  - Presentation of the problem
  - Propagative version
  
- 2 Applications
  - Simulation of a nonstationary Gaussian random field
  - Conditional simulation of a Cox process

# Outline

- 1 Propagative version of the Gibbs sampler
  - Presentation of the problem
  - Propagative version
- 2 Applications
  - Simulation of a nonstationary Gaussian random field
  - Conditional simulation of a Cox process

## Reminder and notation

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

### Definition

The **simple kriging** of  $Y_s$  on  $Y_A$  can be written  $Y_s^A = \sum_{\alpha \in A} \lambda_{\alpha} Y_{\alpha}$ . The kriging weights  $\lambda_{\alpha}$  minimize the estimation variance and are solutions of the linear system

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

The estimation variance is

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

### Property

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

## Reminder and notation

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

### Definition

The **simple kriging** of  $Y_s$  on  $Y_A$  can be written  $Y_s^A = \sum_{\alpha \in A} \lambda_{\alpha} Y_{\alpha}$ . The kriging weights  $\lambda_{\alpha}$  minimize the estimation variance and are solutions of the linear system

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

The estimation variance is

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

### Property

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

# 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  $y_s^n \sim \mathcal{N}(y_s^{S \setminus s}, \sigma_s^{2S \setminus s})$*
- (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

## 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*  $y_s^n \sim \mathcal{N}(y_s^{S \setminus s}, \sigma_s^{2S \setminus s})$
- (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

## 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*  $y_s^n \sim \mathcal{N}(y_s^{S \setminus s}, \sigma_s^{2S \setminus s})$
- (iv) *put*  $y_S^c = y_S^n$  *and goto* (ii)

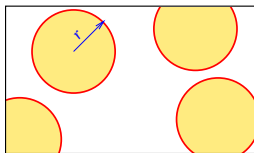
## Problem

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  $B_{s,r}$ .

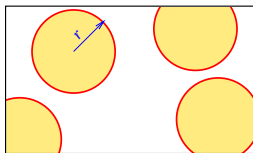


## Algorithm

- (i) *reset*  $y_S^c$
- (ii) *put*  $y_S^n = y_S^c$
- (iii) *select*  $s \sim \mathcal{U}(S)$  *and generate*  $y_S^n \sim \mathcal{N}(y_s^{B_{s,r} \setminus s}, \sigma_s^{2B_{s,r} \setminus s})$
- (iv) *put*  $y_S^c = y_S^n$  *and goto* (ii)

# 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

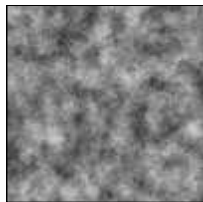
- (i) *reset*  $y_S^c$
- (ii) *put*  $y_S^n = y_S^c$
- (iii) *select*  $s \sim \mathcal{U}(S)$  *and generate*  $y_S^n \sim \mathcal{N}(y_s^{B_{s,r} \setminus s}, \sigma_s^{2B_{s,r} \setminus s})$
- (iv) *put*  $y_S^c = y_S^n$  *and goto* (ii)

Example with 10000 components

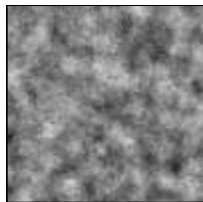
Spherical covariance (10), neighbourhood radius of 15



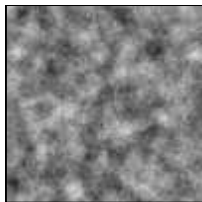
0



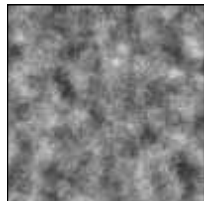
200



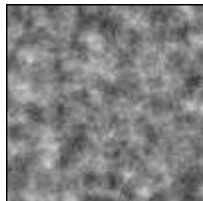
400



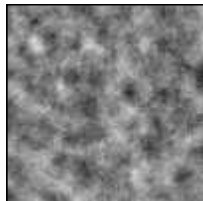
600



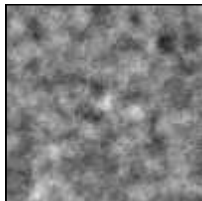
800



1000



1200



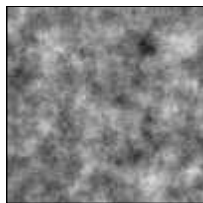
1400

Example with 10000 components

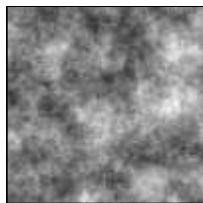
Spherical covariance (10), neighbourhood radius of 5



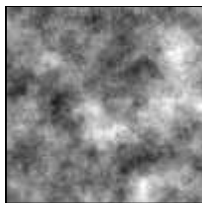
0



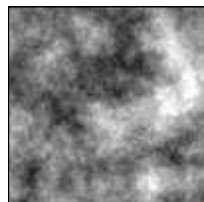
200



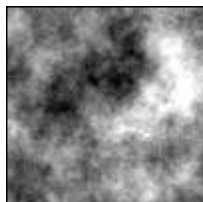
400



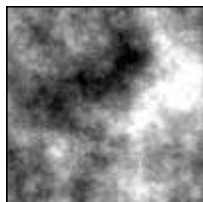
600



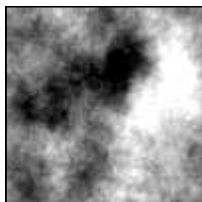
800



1000

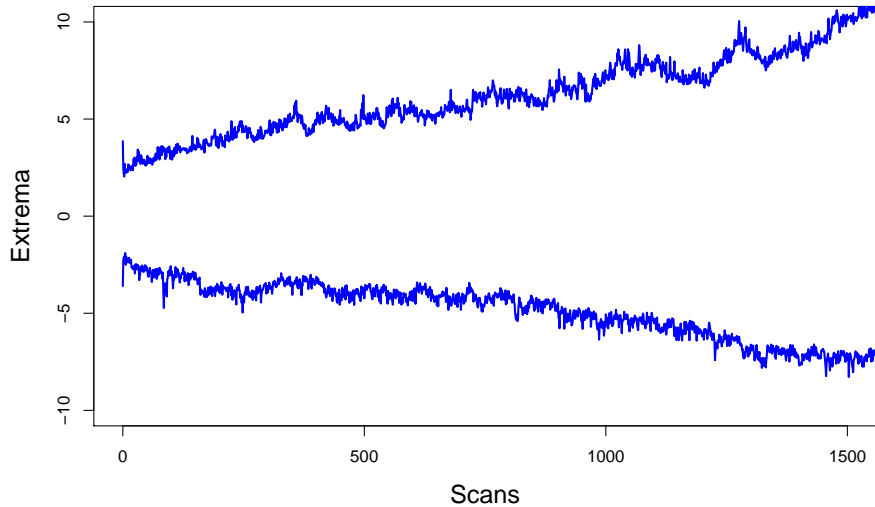


1200



1400

# Extrema



# Outline

- 1 Propagative version of the Gibbs sampler
  - Presentation of the problem
  - Propagative version
- 2 Applications
  - Simulation of a nonstationary Gaussian random field
  - Conditional simulation of a Cox process

## Two useful remarks

### Remark

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

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

P

### Remark (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  $Y = CX$ .

## Two useful remarks

### Remark

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

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

P

### Remark (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  $Y = CX$ .



## Two useful remarks

### Remark

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

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

P

### Remark (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  $Y = CX$ .

## Two useful remarks

### Remark

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

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

P

### Remark (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

$$X_s | X_{S \setminus s} = x_{S \setminus s} \sim \mathcal{N}(x_s - y_s, 1)$$

P

## 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**.

# Running the Gibbs sampler on $X$

## Remark

$$X_s | X_{S \setminus s} = x_{S \setminus s} \sim \mathcal{N}(x_s - y_s, 1)$$

P

## 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**.

# Running the Gibbs sampler on $X$

## Remark

$$X_s | X_{S \setminus s} = x_{S \setminus s} \sim \mathcal{N}(x_s - y_s, 1)$$

P

## 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 $Y$

## Result

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

P

## 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)

# Transporting the Gibbs sampler to $Y$

## Result

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

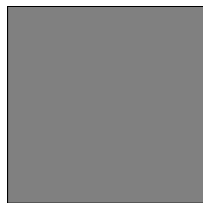
P

## Algorithm

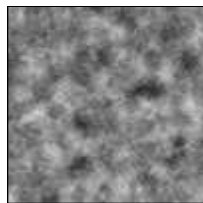
- (i) *reset  $y_s^c$*
- (ii) *select  $p \sim \mathcal{U}(S)$  and generate  $y_p^n \sim \mathcal{N}$*
- (iii) *put  $y_s^n = y_s^c + C_{sp}(y_p^n - y_p^c)$  for each  $s \in S$*
- (iv) *put  $y_s^c = y_s^n$  and goto (ii)*

# Example with 10000 components

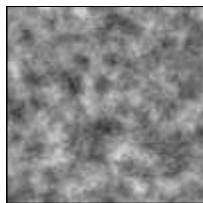
## Spherical covariance (10)



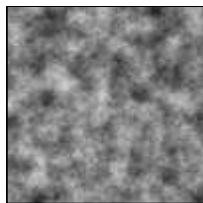
0



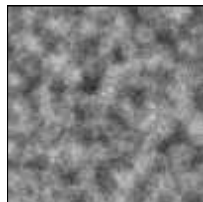
10



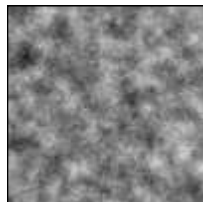
20



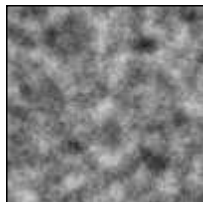
30



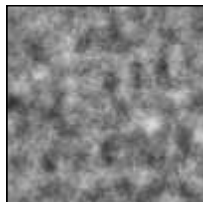
40



50



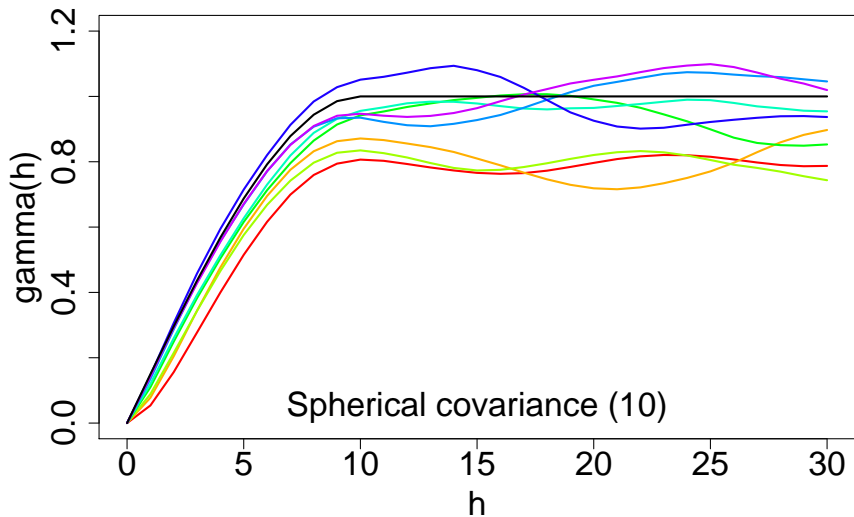
60



70



## Variograms of simulations



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

# First generalization: Generation of the pivot value

Sometimes, it may be advantageous to draw  $y^n$  close to  $y^c$ .

## Remark

$y_p^n$  does not have to be independent of  $y_p^c$ . It just has to be **conditionally** independent of  $y_{S \setminus p}^c$  given  $y_p^c$ .

## Example

One can take  $y_p^n \sim \mathcal{N}(ry_p^c, 1 - r^2)$  with  $r$  close to 1.

# First generalization: Generation of the pivot value

Sometimes, it may be advantageous to draw  $y^n$  close to  $y^c$ .

## Remark

$y_p^n$  does not have to be independent of  $y_p^c$ . It just has to be **conditionally** independent of  $y_{S \setminus p}^c$  given  $y_p^c$ .

## Example

One can take  $y_p^n \sim \mathcal{N}(ry_p^c, 1 - r^2)$  with  $r$  close to 1.

## Second generalization: Blocking strategy

### Remark

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

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**.

### Algorithm

- (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)*

## Second generalization: Blocking strategy

### Remark

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

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**.

### Algorithm

- (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)*

# Outline

- 1 Propagative version of the Gibbs sampler
  - Presentation of the problem
  - Propagative version
- 2 Applications
  - Simulation of a nonstationary Gaussian random field
  - Conditional simulation of a Cox process

# 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  $s \in S$*

# 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  $s \in S$*



# 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  $Y_S$ .

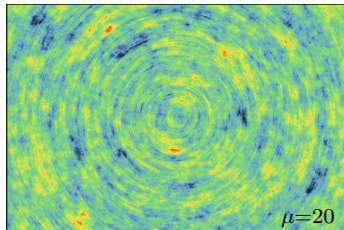
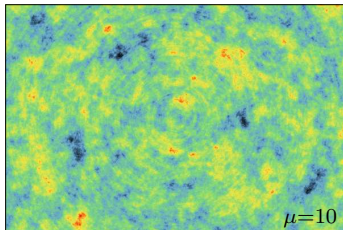
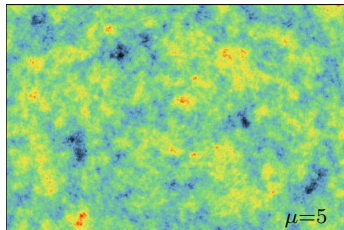
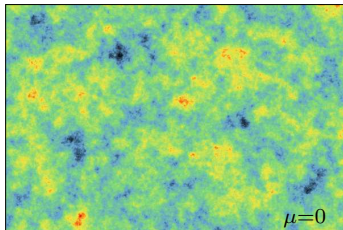
## Algorithm

- (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 of a point source model

C

$$C_{s,t} = \exp(-0.05|s - t|) \exp(-\mu |e^{-0.0025|s|} - e^{-0.0025|t|}|)$$



Simulation field  $600 \times 400$ .

# Outline

- 1 Propagative version of the Gibbs sampler
  - Presentation of the problem
  - Propagative version
- 2 Applications
  - Simulation of a nonstationary Gaussian random field
  - Conditional simulation of a Cox process

# Motivation

## Problem

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.

	5		
		2	
		0	

## Idea

Model the population of trees as a point process (e.g. a Cox process), and perform conditional simulations.

# Motivation

## Problem

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.

	5		
		2	
		0	

## Idea

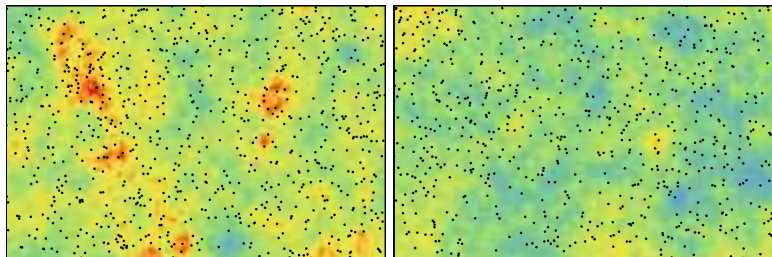
Model the population of trees as a point process (e.g. a **Cox process**), and perform conditional simulations.

# Cox process

P

## Definition

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



## Remark

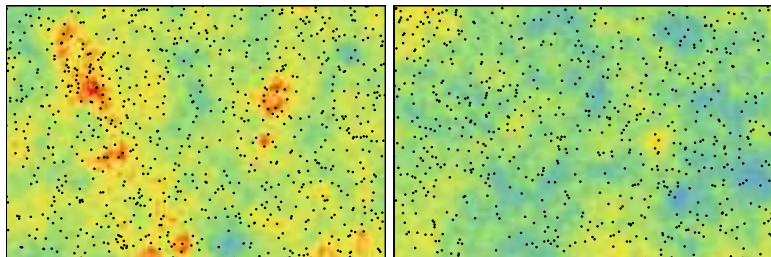
The random intensity function is also called **potential**.

# Cox process

P

## 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

## Notation

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

## Property

Given the potential, the numbers of trees on different plots are **conditionally independent**:

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

## Consequence

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

$$\text{Var}\{N_s\} = \text{Var}\{Z_s\} + E\{Z_s\}$$

$$\text{Cov}\{N_s, N_t\} = \text{Cov}\{Z_s, Z_t\} \quad s \neq t$$



# Properties of the Cox process

## Notation

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

## Property

Given the potential, the numbers of trees on different plots are **conditionally independent**:

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

## Consequence

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

$$\text{Var}\{N_s\} = \text{Var}\{Z_s\} + E\{Z_s\}$$

$$\text{Cov}\{N_s, N_t\} = \text{Cov}\{Z_s, Z_t\} \quad s \neq t$$

# Properties of the Cox process

## Notation

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

## Property

Given the potential, the numbers of trees on different plots are **conditionally independent**:

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

## Consequence

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

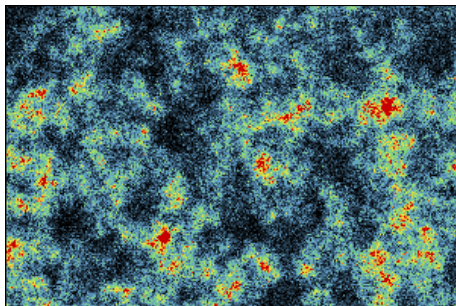
$$\text{Var}\{N_s\} = \text{Var}\{Z_s\} + E\{Z_s\}$$

$$\text{Cov}\{N_s, N_t\} = \text{Cov}\{Z_s, Z_t\} \quad s \neq t$$

# Cox-lognormal process

## Notation

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  $300 \times 200$  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)

# 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)*

# Conditional simulation of a Cox-lognormal process

## Problem

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

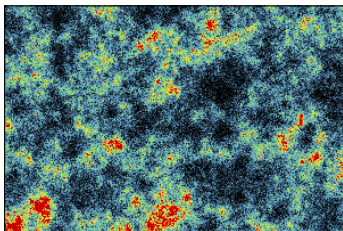
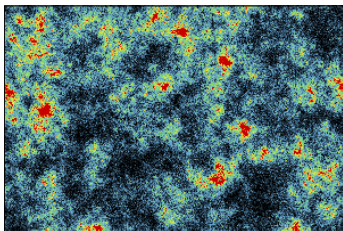
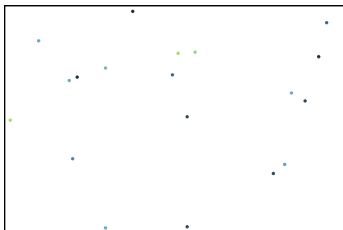
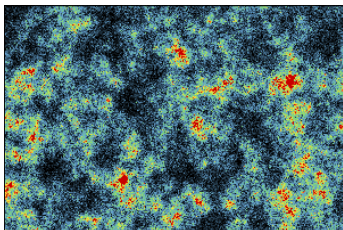
## Idea

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

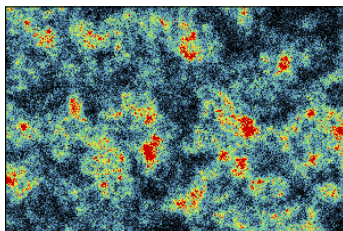
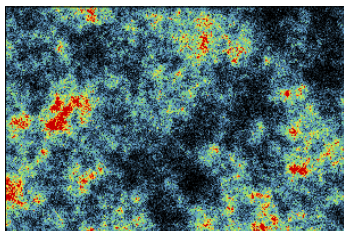
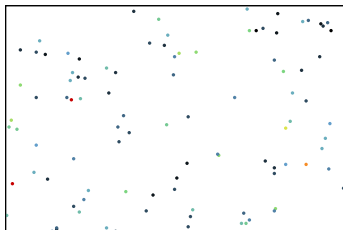
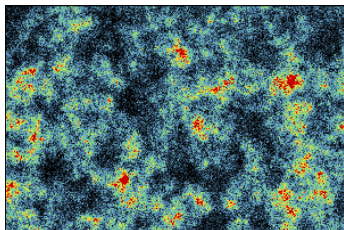
- (i) *put*  $y_S^c = 0$
- (ii) *generate*  $p \sim \mathcal{U}(S)$  and  $y_p^n \sim \mathcal{N}$
- (iii) *put*  $z_A^c = e^{\mu + \sigma y_A^c}$ ,  $z_A^n = z_A^c e^{\sigma C_{Ap}(y_p^n - y_p^c)}$ , and *generate*  $u \sim \mathcal{U}$
- (iv) *if*  $u > p(n_A | z_A^n) / p(n_A | z_A^c)$ , *then goto*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

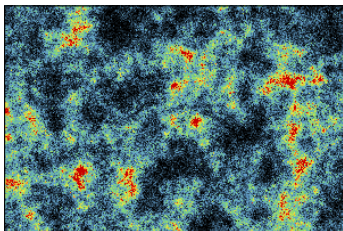
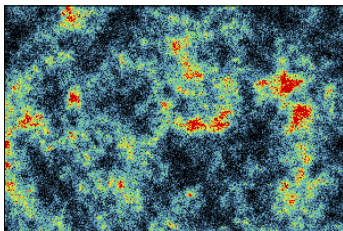
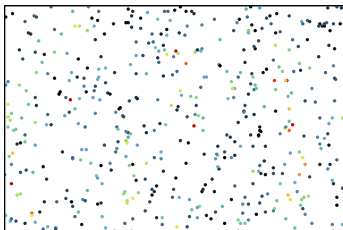
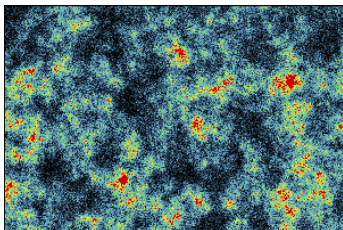
# 100 conditioning data points



"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)

# References

- Brown G., Ferreira J. and Lantuéjoul, C. (2008) – "Conditional simulation of a Cox process using multisupport samples". Proc. of the 8<sup>th</sup> Int. Geostatists Congress, Ortiz J.M. and Emery X. eds., Gecamin, Santiago (Chile).
- Fouedjio F. (2014) – Développement de modèles géostatistiques globaux non stationnaires. PhD Thesis, MinesParisTech.
- Galli A. and Gao H. (2001) – "Rate of convergence of the Gibbs sampler in the Gaussian case". *Mathematical Geology*, 33-6, 653–677.
- Geman S. and Geman D. (1984) – "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images". *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 6-6, 721-741.
- Lantuéjoul C. and Dessassis N. (2012) – "Simulation of a Gaussian random vector: A propagative version of the Gibbs sampler". 9<sup>th</sup> Int. Geostatistics Congress, Oslo (Norway).

## A useful remark

### Remark

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

### Proof

It suffices to check that the proposed weights satisfy the kriging equation  $\sum_{u \neq s} \lambda_u^s C_{tu} = C_{ts}$  for each  $t \neq s$ :

$$\sum_{u \neq s} \lambda_u^s C_{tu} = -\frac{1}{C_{ss}^{-1}} \sum_{u \neq s} C_{su}^{-1} C_{tu} = -\frac{1}{C_{ss}^{-1}} (1_{s=t} - C_{ss}^{-1} C_{ts}) = C_{ts}$$

Regarding the estimation variance, we have

$$\begin{aligned} \sigma_s^{2S \setminus s} &= 1 - \sum_{t, u \neq s} C_{ts} C_{us} C_{tu}^{-1} = 1 - \sum_{t \neq s} C_{ts} \lambda_t^s \\ &= 1 + \frac{1}{C_{ss}^{-1}} \sum_{t \neq s} C_{ts} C_{st}^{-1} = 1 + \frac{1}{C_{ss}^{-1}} (1 - C_{ss} C_{ss}^{-1}) = \frac{1}{C_{ss}^{-1}} \end{aligned}$$

# Running the Gibbs sampler on $X$

## Remark

$$X_s | X_{S \setminus s} = x_{S \setminus s} \sim \mathcal{N}(x_s - y_s, 1)$$

## Proof

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

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

R

# Transporting the Gibbs sampler to $Y$

## Result

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

## Proof

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

$$\begin{aligned} y_s^n &= \sum_t C_{st} x_t^n = \sum_{t \neq p} C_{st} x_t^c + C_{sp} x_p^n \\ &= y_s^c - C_{sp} x_p^c + C_{sp}(x_p^c - y_p^c + u) \\ &= y_s^c + C_{sp}(u - y_p^c) \end{aligned}$$

Taking  $s = p$ , we obtain  $y_p^n = u$ , which finally gives the desired result.

## Example of a point source model

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

### Property

The product of two covariance functions is a covariance function.

### Property

If  $A_{s,t}$  is a covariance function and  $F(s)$  a numerical function, then  $A_{F(s),F(t)}$  is a covariance function.

### Example

Take  $A_{s,t} = \exp(-\mu|s - t|)$  and  $F(s) = \exp(-\nu|s|)$ . Then  $A_{F(s),F(t)} = \exp(-\mu|e^{-\nu|s|} - e^{-\nu|t|}|)$  is a covariance function.

## Example of a point source model

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

### Property

The product of two covariance functions is a covariance function.

### Property

If  $A_{s,t}$  is a covariance function and  $F(s)$  a numerical function, then  $A_{F(s),F(t)}$  is a covariance function.

### Example

Take  $A_{s,t} = \exp(-\mu|s - t|)$  and  $F(s) = \exp(-\nu|s|)$ . Then  $A_{F(s),F(t)} = \exp(-\mu|e^{-\nu|s|} - e^{-\nu|t|}|)$  is a covariance function.

## Example of a point source model

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

### Property

The product of two covariance functions is a covariance function.

### Property

If  $A_{s,t}$  is a covariance function and  $F(s)$  a numerical function, then  $A_{F(s),F(t)}$  is a covariance function.

### Example

Take  $A_{s,t} = \exp(-\mu|s - t|)$  and  $F(s) = \exp(-\nu|s|)$ . Then  $A_{F(s),F(t)} = \exp(-\mu|e^{-\nu|s|} - e^{-\nu|t|}|)$  is a covariance function.



## Example of a point source model

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

### Property

The product of two covariance functions is a covariance function.

### Property

If  $A_{s,t}$  is a covariance function and  $F(s)$  a numerical function, then  $A_{F(s),F(t)}$  is a covariance function.

### Example

Take  $A_{s,t} = \exp(-\mu|s - t|)$  and  $F(s) = \exp(-\nu|s|)$ . Then  $A_{F(s),F(t)} = \exp(-\mu|e^{-\nu|s|} - e^{-\nu|t|}|)$  is a covariance function.

# Poisson point process

## Notation

This process is built using a non-negative function  $\theta$  that is called an **intensity function**. We put

$$\theta(A) = \int_A \theta(x) dx \quad A \in \mathcal{B}(\mathbb{R}^d)$$

## Definition

- (i) the number  $N(A)$  of points in  $A$  is **Poisson distributed** with mean  $\theta(A)$ ;
- (ii) if  $A_1, \dots, A_p$  are **pairwise disjoint**, then  $N(A_1), \dots, N(A_p)$  are **mutually independent**.

## Property

Given  $N(A) = n$ , the  $n$  points are **independently located** in  $A$  with the **same pdf**  $\theta(x)/\theta(A) 1_{x \in A}$ .

# Poisson point process

## Notation

This process is built using a non-negative function  $\theta$  that is called an **intensity function**. We put

$$\theta(A) = \int_A \theta(x) dx \quad A \in \mathcal{B}(\mathbb{R}^d)$$

## Definition

- (i) the number  $N(A)$  of points in  $A$  is **Poisson distributed** with mean  $\theta(A)$ ;
- (ii) if  $A_1, \dots, A_p$  are **pairwise disjoint**, then  $N(A_1), \dots, N(A_p)$  are **mutually independent**.

## Property

Given  $N(A) = n$ , the  $n$  points are **independently located** in  $A$  with the **same pdf**  $\theta(x)/\theta(A) 1_{x \in A}$ .

# Poisson point process

## Notation

This process is built using a non-negative function  $\theta$  that is called an **intensity function**. We put

$$\theta(A) = \int_A \theta(x) dx \quad A \in \mathcal{B}(\mathbb{R}^d)$$

## Definition

- (i) the number  $N(A)$  of points in  $A$  is **Poisson distributed** with mean  $\theta(A)$ ;
- (ii) if  $A_1, \dots, A_p$  are **pairwise disjoint**, then  $N(A_1), \dots, N(A_p)$  are **mutually independent**.

## Property

Given  $N(A) = n$ , the  $n$  points are **independently located** in  $A$  with the **same pdf**  $\theta(x)/\theta(A) \mathbf{1}_{x \in A}$ .