Models and inference for random fields indexed on undirected graphs

Mike PEREIRA[1,2], Nicolas DESASSIS[1]

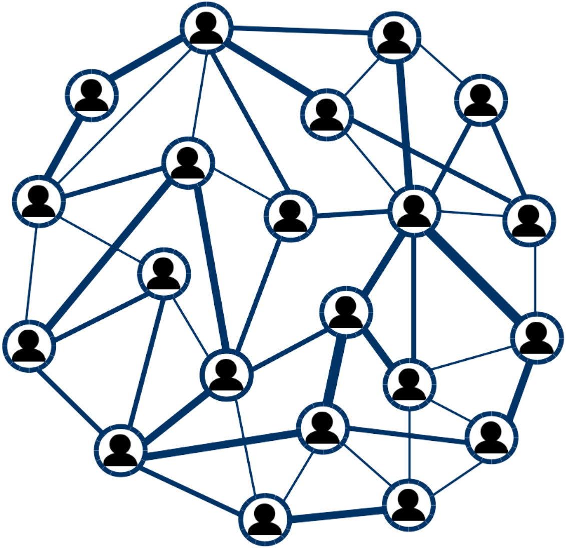
[1]Geostatistics team, MINES ParisTech, PSL Research University
[2]ESTIMAGES France

RESSTE Day
"Hierarchical Bayesian Models for spatio-temporal data"
May 16th, 2017

ESTIMAGES Decide with data

MINES ParisTech

Introduction



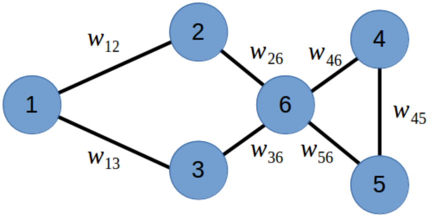
Outline

Outline

Graph : a mathematical definition

A graph $\mathcal{G}$ is a triplet $(\mathcal{V}, \mathcal{E}, \mathcal{W})$ where

- $\mathcal{V}$ = set of N vertices of the graph.
- $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ = set of edges. Adjacent vertices i and j are denoted $i \sim j$.
- $\mathcal{W} : \mathcal{E} \mapsto \mathbb{R}$ = symmetric weight function. Weight of edge (i, j) is denoted $w_{ij} = w_{ji}$.



Work Hypothesis

Only **undirected** and **loopless** graphs are studied.

Graph Signals

Graph signal

A **graph signal** is a vector of real values indexed by the vertices of a graph.
It is said **random** when its values at the vertices are random.

Example : marketing interest for a new product among the users of a social network.

Work Hypothesis

Only **Gaussian** random signals are considered.

Shift operator

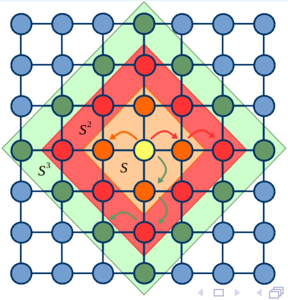
Definition : Shift Operator

A shift operator $\boldsymbol{S}$ on graph $\mathcal{G}$ is a $N \times N$ matrix such that :

$$S_{ij} \neq 0 \Rightarrow i = j \quad \text{ou} \quad i \sim j$$

Proposition

For $k \in \mathbb{N}$, $\boldsymbol{S}$ verifies : $[\boldsymbol{S}^k]_{ij} \neq 0 \Rightarrow i = j$ or $\exists$ a chain of vertices of length $\leq k$ between nodes i and j.



Outline

Graph filter

Work Hypothesis

$\boldsymbol{S}$ is **symmetric** : accordingly, it is diagonalizable on $\mathbb{R}$. Hence, denote $\lambda_1 \leq ... \leq \lambda_N$ its eigenvalues and $\boldsymbol{V}$ its eigenbasis ($\boldsymbol{V}\boldsymbol{V}^T = \boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}$) .

$$\boldsymbol{S} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^T \text{ with } \boldsymbol{\Lambda} = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_N \end{pmatrix}$$

Definition : Graph filter

A **graph filter** $\mathrm{h}(\boldsymbol{S})$ is a matrix defined from a function $\mathrm{h} : \mathbb{R} \mapsto \mathbb{R}$ by the relation :

$$\mathrm{h}(\boldsymbol{S}) := \boldsymbol{V}\mathrm{h}(\boldsymbol{\Lambda})\boldsymbol{V}^T = \boldsymbol{V} \begin{pmatrix} \mathrm{h}(\lambda_1) & & \\ & \ddots & \\ & & \mathrm{h}(\lambda_N) \end{pmatrix} \boldsymbol{V}^T$$

Note : Only need to know $\mathrm{h}(\lambda_1), ..., \mathrm{h}(\lambda_N)$ *to define* $\mathrm{h}(\boldsymbol{S})$

Stationarity on graphs

Definition : Stationarity on graphs

A random graph signal z is said $\boldsymbol{S}$**-stationary** if :

1. its mean is constant over $\mathcal{V}$ (denoted m_z)
2. its covariance matrix $\boldsymbol{\Sigma}_z$ is a graph filter for a function $\mathfrak{K}_z : \mathbb{R} \mapsto \mathbb{R}_+$, called the **spectrum function** of z:

$$\boldsymbol{\Sigma}_z := \mathbb{E}\{(z - m_z)(z - m_z)^T\} = \mathfrak{K}_z(\boldsymbol{S})$$

Note

$\boldsymbol{S}$-stationary signals with $\mathfrak{K}_z$ of the form $\mathfrak{K}_z(x) = (a_0 + a_1 x)^{-1}$ correspond to markov random fields with precision matrix :

$$\boldsymbol{Q} = a_0 \boldsymbol{I} + a_1 \boldsymbol{S}$$

Stationarity on graphs

Definition : Stationarity on graphs

A random graph signal z is said $\boldsymbol{S}$**-stationary** if :

1. its mean is constant over $\mathcal{V}$ (denoted m_z)
2. its covariance matrix $\boldsymbol{\Sigma}_z$ is a graph filter for a function $\mathfrak{K}_z : \mathbb{R} \mapsto \mathbb{R}_+$, called the **spectrum function** of z:

$$\boldsymbol{\Sigma}_z := \mathbb{E}\{(z - m_z)(z - m_z)^T\} = \mathfrak{K}_z(\boldsymbol{S})$$

Note

$\boldsymbol{S}$-stationary signals with $\mathfrak{K}_z$ of the form $\mathfrak{K}_z(x) = (a_0 + a_1 x)^{-1}$ correspond to markov random fields with precision matrix :

$$\boldsymbol{Q} = a_0 \boldsymbol{I} + a_1 \boldsymbol{S}$$

Example : White noise

The **graph white noise** ε is the random signal whose components are independent standard gaussian variables.

Proposition

To simulate a $\boldsymbol{S}$-stationary signal z and with spectrum function $\mathrm{f} : \mathbb{R} \to \mathbb{R}_+$:

- Generate a graph white noise ε
- Compute $z = \sqrt{\mathrm{f}}(\boldsymbol{S})\varepsilon$

Proof... Go

Problem

How to compute $\mathrm{h}(\boldsymbol{S})\varepsilon$?

$$\mathrm{h}(\boldsymbol{S})\varepsilon = \underline{\boldsymbol{V}\underline{\mathrm{h}(\Lambda)\underline{\boldsymbol{V}^T\varepsilon}}}$$

$\Rightarrow$ Diagonalization + Storage : Expensive!!

Example : White noise

The **graph white noise** ε is the random signal whose components are independent standard gaussian variables.

Proposition

To simulate a $\boldsymbol{S}$-stationary signal z and with spectrum function $\mathrm{f} : \mathbb{R} \to \mathbb{R}_+$:

- Generate a graph white noise ε
- Compute $z = \sqrt{\mathrm{f}}(\boldsymbol{S})\varepsilon$

Proof... Go

Problem

How to compute $\mathrm{h}(\boldsymbol{S})\varepsilon$?

$$\mathrm{h}(\boldsymbol{S})\varepsilon = \underline{\boldsymbol{V}\underline{\mathrm{h}(\Lambda)\underline{\boldsymbol{V}^T\varepsilon}}}$$

$\Rightarrow$ Diagonalization + Storage : Expensive!!

Simulation of stationary graph signals

Example : White noise

The **graph white noise** ε is the random signal whose components are independent standard gaussian variables.

Proposition

To simulate a $\boldsymbol{S}$-stationary signal z and with spectrum function $\mathrm{f} : \mathbb{R} \to \mathbb{R}_+$:

- Generate a graph white noise ε
- Compute $z = \sqrt{\mathrm{f}}(\boldsymbol{S})\varepsilon$

Proof... Go

Problem

How to compute $\mathrm{h}(\boldsymbol{S})\varepsilon$?

$$\mathrm{h}(\boldsymbol{S})\varepsilon = \underline{\boldsymbol{V}\underline{\mathrm{h}(\boldsymbol{\Lambda})\underline{\boldsymbol{V}^T\varepsilon}}}$$

$\Rightarrow$ Diagonalization + Storage : Expensive!!

Outline

Fast computation of graph filters

Idea

Computing $\mathrm{p}(\boldsymbol{S})$ for a polynomial function p is feasible without diagonalization! (*Proof...* Go)
For more general functions h : approximate h by a polynomial.

Workflow

- Find a polynomial approximation p of h s.t.

$$\forall k \in [\![1, N]\!], \mathrm{p}(\lambda_k) \approx \mathrm{h}(\lambda_k)$$

- Compute $\mathrm{p}(\boldsymbol{S})$ (matrix polynomial)
- Take $\mathrm{h}(\boldsymbol{S}) \approx \mathrm{p}(\boldsymbol{S})$ (same eigenbasis, similar eigenvalues)

$\Rightarrow$ Polynomial approximation of h on the interval $[\lambda_{\min}, \lambda_{\max}]$ using **Chebyshev polynomials** (fast by FFT)

Outline

General notation for graphs
Stationary signal processing on graphs
Computation of graph filters
Model Inference
Empirical method for model inference
Model inference by likelihood-based method
Efficient simulation scheme
Conclusion

Empirical method for model inference

Notation

$\boldsymbol{S}$ a symmetric shift operator : $\boldsymbol{S} = \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^T$ with :

- $\lambda_1 \leq ... \leq \lambda_N$ its eigenvalues
- $\boldsymbol{S} = \boldsymbol{V} \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_N \end{pmatrix} \boldsymbol{V}^T$

z random $\boldsymbol{S}$-stationary signal with :

- Mean 0
- Covariance matrix $\boldsymbol{\Sigma_z} = \mathfrak{K}_{\boldsymbol{z}}(\boldsymbol{S}) = \boldsymbol{V}\mathfrak{K}_{\boldsymbol{z}}(\boldsymbol{\Lambda})\boldsymbol{V}^T$

Problem

Given a realization of $\boldsymbol{S}$-stationary signal z, find its spectrum function $\mathfrak{K}_{\boldsymbol{z}}$ empirically.

Power spectral density (PSD)

Proposition

The signal $\tilde{z} = \boldsymbol{V}^T z$ of z has covariance matrix :

$$\boldsymbol{\Sigma}_{\tilde{z}} = \boldsymbol{V}^T \boldsymbol{\Sigma}_z \boldsymbol{V} = \mathfrak{K}_z(\boldsymbol{\Lambda}) = \begin{pmatrix} \mathfrak{K}_z(\lambda_1) & & \\ & \ddots & \\ & & \mathfrak{K}_z(\lambda_N) \end{pmatrix}$$

The components of $\tilde{z}$ are **independent** random variables.

Definition : Power spectral density

The power spectral density $\tilde{\boldsymbol{p}}_z$ of z is the vector defined as :

$$\tilde{\boldsymbol{p}}_z := \mathrm{diag}(\boldsymbol{V}^T \boldsymbol{\Sigma}_z \boldsymbol{V}) = (\mathfrak{K}_z(\lambda_1), \dots, \mathfrak{K}_z(\lambda_N))^T$$

Its elements are (equivalently) :

- the eigenvalues of the covariance matrix of z
- the variance of the components of $\tilde{z} = \boldsymbol{V}^T z$:

$$\mathfrak{K}_z(\lambda_k) = \mathrm{Var}(\tilde{z}_k) = [\boldsymbol{\Sigma}_{\tilde{z}}]_{kk} = \mathbb{E}(\tilde{z}_k^2)$$

Estimation of $\mathfrak{K}_z$

Problem : How to estimate $\mathfrak{K}_z$?

Idea : Kernel Density Estimation of $\mathfrak{K}_z$ over an interval $[a, b] \supset \{\lambda_1, ..., \lambda_N\}$ *(see Perraudin and Vandergheynst, 2016)*

The value of $\mathfrak{K}_z$ at point $x \in [a, b]$ can be estimated using a Gaussian kernel (centered at x), $g_\sigma^{(x)} : \lambda \mapsto \exp\left(-\frac{(\lambda - x)^2}{2\sigma^2}\right)$

$$\widehat{\mathfrak{K}}_z(x) = \frac{\mathbb{E}\left(\|g_\sigma^{(x)}(\boldsymbol{S})z\|^2\right)}{\mathbb{E}\left(\|g_\sigma^{(x)}(\boldsymbol{S})\varepsilon\|^2\right)}$$

Where $\|.\|$ is the Euclidean norm.

Proof... Go

In practice, $\mathbb{E}\left(\|g_\sigma^{(x)}(\boldsymbol{S})z\|^2\right)$ is computed from the single realization of z that is known.

Example of application

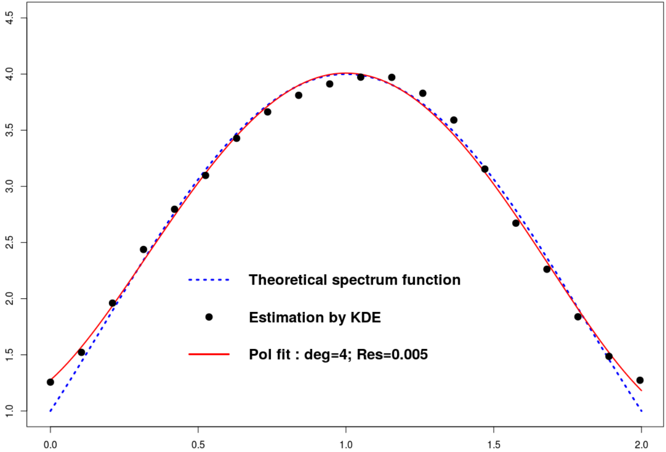


Figure: Estimation of the spectrum function of a stationary field simulated on a 200x200 grid

Likelihood-based method for model inference I

Problem

Given a realization of a $\boldsymbol{S}$-stationary signal z, find its spectrum function $\mathfrak{K}_z$ by a likelihood-based approach.

Suppose that $\mathfrak{K}_z = \mathfrak{K}_z^{\theta}$ depends on a vector of parameters $\boldsymbol{\theta}$. The log-likelihood associated to z and $\boldsymbol{\theta}$ is given by :

$$\mathfrak{L}(z, \boldsymbol{\theta}) = -\frac{1}{2}\left(N \log 2\pi + \log \det\left(\mathfrak{K}_z^{\theta}(\boldsymbol{S})\right) + z^T \mathfrak{K}_z^{\theta}(\boldsymbol{S})^{-1} z\right)$$

Idea

Use fast computation of graph filters technique to compute efficiently determinant and inverse.

Likelihood-based method for model inference II

We have :

$$\mathfrak{K}_z^{\theta}(\boldsymbol{S})^{-1} = \boldsymbol{V} \begin{pmatrix} 1/\mathfrak{K}_z^{\theta}(\lambda_1) & & \\ & \ddots & \\ & & 1/\mathfrak{K}_z^{\theta}(\lambda_N) \end{pmatrix} \boldsymbol{V}^T = \frac{1}{\mathfrak{K}_z^{\theta}}(\boldsymbol{S})$$

$\Rightarrow$ Use polynomial approximation of $\frac{1}{\mathfrak{K}_z^{\theta}}$

And

$$\log\det\left(\mathfrak{K}_z^{\theta}(\boldsymbol{S})\right) = \sum_{k=0}^{N-1} \log(\mathfrak{K}_z^{\theta}(\lambda_k))$$

Idea

Approximate this sum using the histogram of eigenvalues $\lambda_1, \ldots, \lambda_N$.

Determinant by histogram approx.

Definition

$[a,b] \supset \{\lambda_1, \dots, \lambda_N\}$. For $M \in \mathbb{N}$ (number of breaks) denote $\tau = \frac{b-a}{M}$ and $a_m = a + m\tau : m \in 0, \dots, M$
Denote $\text{hist}(a_m)$ the count :

$$\text{hist}(a_m) := \text{Card}\left\{ i \in [\![0, N-1]\!] : \lambda_i \in]a_m - \frac{\tau}{2}, a_m + \frac{\tau}{2}] \right\}$$

Proposition

$$\log\det\left(\mathfrak{K}_{\boldsymbol{z}}^{\boldsymbol{\theta}}(\boldsymbol{S})\right) \approx \sum_{m=0}^{M} \text{hist}(a_m) \log(\mathfrak{K}_{\boldsymbol{z}}^{\boldsymbol{\theta}}(a_m))$$

Where :

$$\boxed{\text{hist}(a_m) = \mathbb{E}\left(||\mathbf{1}_{]a_m - \frac{\tau}{2}, a_m + \frac{\tau}{2}]}(\mathbf{S})\varepsilon||^2 \right)}$$

The counts of the histogram can be obtained as follows :

$$\text{hist}(a_m) = \sum_{i=0}^{N-1} \mathbf{1}_{]a_m - \frac{\tau}{2}, a_m + \frac{\tau}{2}]}(\lambda_i) = \sum_{i=0}^{N-1} \mathbf{1}_{]a_m - \frac{\tau}{2}, a_m + \frac{\tau}{2}]}(\lambda_i)^2$$

Notice that if ε is a white noise, its PSD is the vector $\mathbf{1} = (1, ..., 1)^T$. And therefore,

$$\text{hist}(a_m) = \sum_{i=0}^{N-1} \mathbf{1}_{]a_m - \frac{\tau}{2}, a_m + \frac{\tau}{2}]}(\lambda_i)^2 \times \underbrace{1}_{=\mathbb{E}(\tilde{\varepsilon}_i^2)}$$

$$= \mathbb{E}\left(\left\| \begin{pmatrix} \mathbf{1}_{]a_m - \frac{\tau}{2}, a_m + \frac{\tau}{2}]}(\lambda_1) & & \\ & \ddots & \\ & & \mathbf{1}_{]a_m - \frac{\tau}{2}, a_m + \frac{\tau}{2}]}(\lambda_N) \end{pmatrix} \begin{pmatrix} \tilde{\varepsilon}_1 \\ \vdots \\ \tilde{\varepsilon}_N \end{pmatrix} \right\|^2 \right)$$

Determinant by histogram approx. : Proof

The counts of the histogram can be obtained as follows :

$$\text{hist}(a_m) = \sum_{i=0}^{N-1} \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\lambda_i) = \sum_{i=0}^{N-1} \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\lambda_i)^2$$

Notice that if ε is a white noise, its PSD is the vector $\mathbf{1} = (1, ..., 1)^T$. And therefore,

$$\text{hist}(a_m) = \sum_{i=0}^{N-1} \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\lambda_i)^2 \times \underbrace{1}_{=\mathbb{E}(\tilde{\varepsilon}_i^2)}$$

$$= \mathbb{E}\left(\left\| \begin{pmatrix} \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\lambda_1) & & \\ & \ddots & \\ & & \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\lambda_N) \end{pmatrix} \boldsymbol{V}^T \varepsilon \right\|^2 \right)$$

The counts of the histogram can be obtained as follows :

$$\text{hist}(a_m) = \sum_{i=0}^{N-1} \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\lambda_i) = \sum_{i=0}^{N-1} \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\lambda_i)^2$$

Notice that if ε is a white noise, its PSD is the vector $\mathbf{1} = (1, ..., 1)^T$. And therefore,

$$\text{hist}(a_m) = \sum_{i=0}^{N-1} \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\lambda_i)^2 \times \underbrace{1}_{=\mathbb{E}(\tilde{\varepsilon}_i^2)}$$

$$= \mathbb{E}\left(\left\| \boldsymbol{V} \begin{pmatrix} \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\lambda_1) & & \\ & \ddots & \\ & & \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\lambda_N) \end{pmatrix} \boldsymbol{V}^T \varepsilon \right\|^2 \right)$$

Hence :

$$\text{hist}(a_m) = \mathbb{E}\left(\left\| \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\mathbf{S})\varepsilon \right\|^2 \right)$$

Determinant by histogram approx. : Proof

The counts of the histogram can be obtained as follows :

$$\mathrm{hist}(a_m) = \sum_{i=0}^{N-1} \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\lambda_i) = \sum_{i=0}^{N-1} \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\lambda_i)^2$$

Notice that if ε is a white noise, its PSD is the vector $\mathbf{1} = (1, ..., 1)^T$. And therefore,

$$\mathrm{hist}(a_m) = \sum_{i=0}^{N-1} \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\lambda_i)^2 \times \underbrace{1}_{=\mathbb{E}(\tilde{\varepsilon}_i^2)}$$

$$= \mathbb{E}\left(\left\| \boldsymbol{V} \begin{pmatrix} \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\lambda_1) & & \\ & \ddots & \\ & & \mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\lambda_N) \end{pmatrix} \boldsymbol{V}^T \varepsilon \right\|^2 \right)$$

Hence :

$$\boxed{\mathrm{hist}(a_m) = \mathbb{E}\left(||\mathbf{1}_{]a_m-\frac{\tau}{2},a_m+\frac{\tau}{2}]}(\mathbf{S})\varepsilon||^2 \right)}$$

Example of application

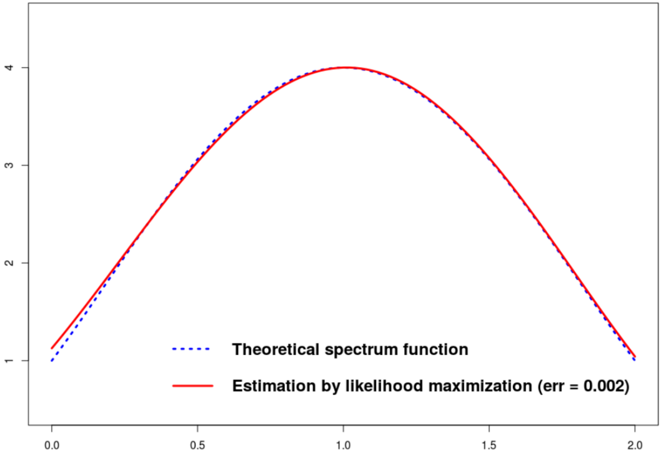


Figure: Estimation of the spectrum function of a stationary field simulated on a 200x200 grid by likelihood-based approach (error = integral of the squared difference)

Outline

Random field definition

Problem

Given a random field z defined on a spatial domain as the solution by finite elements of the following SPDE :

$$\left(1 - \mathrm{div}\left(\boldsymbol{H}(s)\nabla\right)\right)^{\alpha/2} z(s) = \mathfrak{W}(s)$$

Compute a (non-conditional) simulation of z.

Finite Element method ⇒ Discretization of differential operators.

The precision matrix of z can then be expressed using a (much) sparser matrix $\boldsymbol{M}$ *(see Lindgren et al. 2011)*:

$$\boldsymbol{Q} = \boldsymbol{D}\sum_{p=0}^{P} b_p \boldsymbol{M}^p \boldsymbol{D} = \boldsymbol{D}\mathrm{p}(\boldsymbol{M})\boldsymbol{D}$$

Random field definition

Problem

Given a random field z defined on a spatial domain as the solution by finite elements of the following SPDE :

$$\left(1 - \mathrm{div}\left(\boldsymbol{H}(s)\nabla\right)\right)^{\alpha/2} z(s) = \mathfrak{W}(s)$$

Compute a (non-conditional) simulation of z.

Finite Element method ⇒ Discretization of differential operators.
The precision matrix of z can then be expressed using a (much) sparser matrix $\boldsymbol{M}$ *(see Lindgren et al. 2011)*:

$$\boldsymbol{Q} = \boldsymbol{D}\sum_{p=0}^{P} b_p \boldsymbol{M}^p \boldsymbol{D} = \boldsymbol{D}\mathrm{p}(\boldsymbol{M})\boldsymbol{D}$$

$$\boldsymbol{Q} = \boldsymbol{D}\sum_{p=0}^{P} b_p \boldsymbol{M}^p \boldsymbol{D} = \boldsymbol{D}\mathrm{p}(\boldsymbol{M})\boldsymbol{D}$$

Current solution

A simulation of z is then computed using a Cholesky decomposition of $\boldsymbol{Q}$:

$$z = \boldsymbol{Q}^{-1/2}\varepsilon$$

$\Rightarrow$ Problem : Computing the Cholesky decomposition of $\boldsymbol{Q}$ is untractable for large problems.

Proposed solution

Use fast filtering technique to compute matrix $\boldsymbol{Q}^{-1/2} = \boldsymbol{D}^{-1} f(\boldsymbol{M})$ where $f : y \mapsto \frac{1}{\sqrt{\mathrm{p}(y)}} = \frac{1}{\sqrt{\sum_{p=0}^{P} b_p y^p}}$

$$\boldsymbol{Q} = \boldsymbol{D} \sum_{p=0}^{P} b_p \boldsymbol{M}^p \boldsymbol{D} = \boldsymbol{D}\mathrm{p}(\boldsymbol{M})\boldsymbol{D}$$

Current solution

A simulation of z is then computed using a Cholesky decomposition of $\boldsymbol{Q}$:

$$z = \boldsymbol{Q}^{-1/2}\varepsilon$$

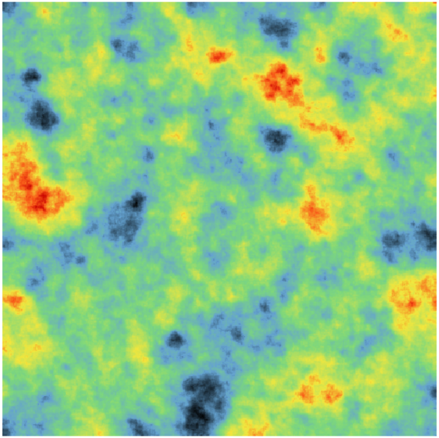
$\Rightarrow$ Problem : Computing the Cholesky decomposition of $\boldsymbol{Q}$ is untractable for large problems.

Proposed solution

Use fast filtering technique to compute matrix $\boldsymbol{Q}^{-1/2} = \boldsymbol{D}^{-1} f(\boldsymbol{M})$ where $f : y \mapsto \frac{1}{\sqrt{\mathrm{p}(y)}} = \frac{1}{\sqrt{\sum_{p=0}^{P} b_p y^p}}$

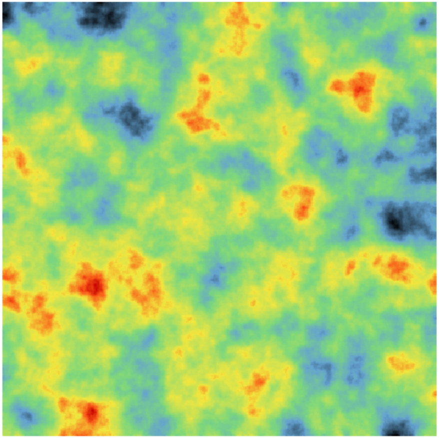
Efficient simulation scheme

Non Conditional simulation of a Matern model ($\alpha = 2$) on a 400x400 grid using Cholesky



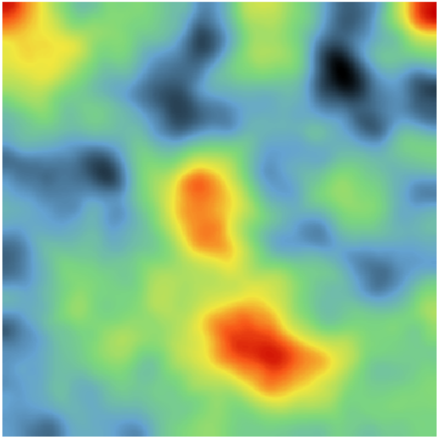
Efficient simulation scheme

Non Conditional simulation of a Matern model ($\alpha = 2$) on a 400x400 grid using Fast filtering



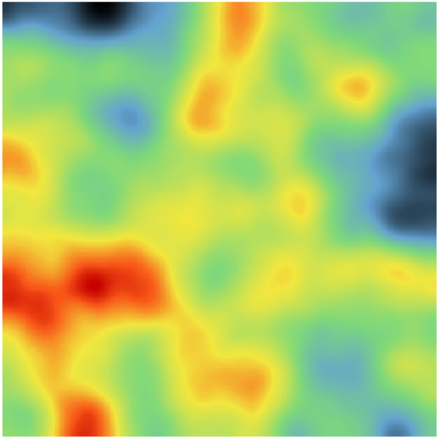
Efficient simulation scheme

Non Conditional simulation of a Matern model ($\alpha = 4$) on a 400x400 grid using Cholesky

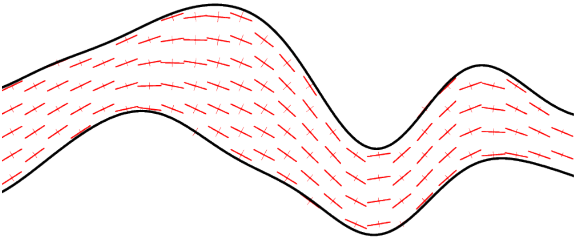


Efficient simulation scheme

Non Conditional simulation of a Matern model ($\alpha = 4$) on a 400x400 grid using Fast filtering

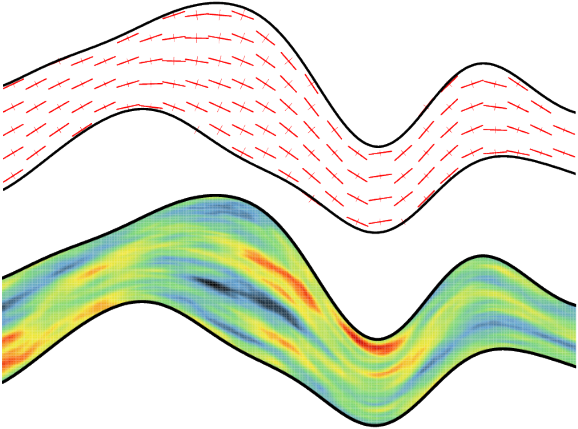


Non Conditional simulation of an (varying) anisotropic exponential model ($\alpha = 3/2$) (top = ellipses of anisotropy, bottom = field simulation).



Efficient simulation scheme

Non Conditional simulation of an (varying) anisotropic exponential model ($\alpha = 3/2$) (top = ellipses of anisotropy, bottom = field simulation).



Outline

To come...

- Model inference when data are missing (data augmentation/completion, EM algorithm) and signal interpolation
- Work on spatio-temporal models for prediction.

$$\text{Ex}: \frac{\partial z}{\partial t} + \mathrm{h}(\boldsymbol{S})z = \varepsilon$$

Thank you for your attention!
Questions?

ESTIMAGES
Decide with **data**

MINES ParisTech

Proof of the simulation process

Proof : Remark that $\boldsymbol{\Sigma}_{\varepsilon} = \boldsymbol{I}$. Then if $\boldsymbol{z} = \sqrt{\mathrm{f}}(\boldsymbol{S})\varepsilon$, we have :

$$\begin{aligned}
\boldsymbol{\Sigma}_{\boldsymbol{z}} &= \sqrt{\mathrm{f}}(\boldsymbol{S})\boldsymbol{I}\sqrt{\mathrm{f}}(\boldsymbol{S})^T = \sqrt{\mathrm{f}}(\boldsymbol{S})\sqrt{\mathrm{f}}(\boldsymbol{S})^T \\
&= \boldsymbol{V}\begin{pmatrix} \sqrt{\mathrm{f}(\lambda_1)} & & \\ & \ddots & \\ & & \sqrt{\mathrm{f}(\lambda_N)} \end{pmatrix} \underbrace{\boldsymbol{V}^T\boldsymbol{V}}_{=\boldsymbol{I}} \begin{pmatrix} \sqrt{\mathrm{f}(\lambda_1)} & & \\ & \ddots & \\ & & \sqrt{\mathrm{f}(\lambda_N)} \end{pmatrix} \boldsymbol{V}^T \\
&= \boldsymbol{V}\begin{pmatrix} \sqrt{\mathrm{f}(\lambda_1)} & & \\ & \ddots & \\ & & \sqrt{\mathrm{f}(\lambda_N)} \end{pmatrix} \begin{pmatrix} \sqrt{\mathrm{f}(\lambda_1)} & & \\ & \ddots & \\ & & \sqrt{\mathrm{f}(\lambda_N)} \end{pmatrix} \boldsymbol{V}^T \\
&= \boldsymbol{V}\begin{pmatrix} \sqrt{\mathrm{f}(\lambda_1)}^2 & & \\ & \ddots & \\ & & \sqrt{\mathrm{f}(\lambda_N)}^2 \end{pmatrix} \boldsymbol{V}^T = \boldsymbol{V}\begin{pmatrix} \mathrm{f}(\lambda_1) & & \\ & \ddots & \\ & & \mathrm{f}(\lambda_N) \end{pmatrix} \boldsymbol{V}^T \\
&= \mathrm{f}(\boldsymbol{S})
\end{aligned}$$

Back

Fast computation of graph filters : Proof

$$\boldsymbol{S}^k = \boldsymbol{S}\boldsymbol{S}\boldsymbol{S}\ldots\boldsymbol{S} = \boldsymbol{U}\Lambda\underbrace{\boldsymbol{U}^T\boldsymbol{U}}_{=\boldsymbol{I}}\Lambda\underbrace{\boldsymbol{U}^T\ldots}_{=\boldsymbol{I}}\underbrace{\ldots\boldsymbol{U}}_{=\boldsymbol{I}}\Lambda\boldsymbol{U}^T = \boldsymbol{U}\Lambda^k\boldsymbol{U}^T$$

For a polynomial $\mathrm{p} : x \mapsto a_0 + a_1 x + \ldots + a_m x^m$

$$\mathrm{p}(\boldsymbol{\Lambda}) = \begin{pmatrix} \mathrm{p}(\lambda_1) & & \\ & \ddots & \\ & & \mathrm{p}(\lambda_N) \end{pmatrix} = a_0\boldsymbol{I} + a_1\boldsymbol{\Lambda} + \ldots + a_m\boldsymbol{\Lambda}^m$$

Therefore

$$\begin{aligned}\mathrm{p}(\boldsymbol{S}) := \boldsymbol{U}\mathrm{p}(\boldsymbol{\Lambda})\boldsymbol{U}^T &= a_0\boldsymbol{U}\boldsymbol{I}\boldsymbol{U}^T + a_1\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T + \ldots + a_m\boldsymbol{U}\boldsymbol{\Lambda}^m\boldsymbol{U}^T \\ &= a_0\boldsymbol{I} + a_1\boldsymbol{S} + \ldots + a_P\boldsymbol{S}^m \rightarrow \text{ polynomial}\end{aligned}$$

Back

Estimation of $\mathfrak{K}_{\boldsymbol{z}}$: Proof I

$$\widehat{\gamma}_{\boldsymbol{z}}(x) = \frac{1}{C_x}\sum_{k=0}^{N-1} g_\sigma^{(x)}(\lambda_k)^2 \mathfrak{K}_{\boldsymbol{z}}(\lambda_k); \quad C_x = \sum_{k=0}^{N-1} g_\sigma^{(x)}(\lambda_k)^2$$

$$\sum_{k=0}^{N-1} g_\sigma^{(x)}(\lambda_k)^2 \underbrace{\mathfrak{K}_{\boldsymbol{z}}(\lambda_k)}_{=\mathbb{E}(\tilde{z}_k^2)} = \mathbb{E}\left(\sum_{k=0}^{N-1} g_\sigma^{(x)}(\lambda_k)^2 \tilde{z}_k^2\right)$$

$$= \mathbb{E}\left(\left\| \begin{pmatrix} g_\sigma^{(x)}(\lambda_1) & & \\ & \ddots & \\ & & g_\sigma^{(x)}(\lambda_N) \end{pmatrix} \begin{pmatrix} \tilde{z}_1 \\ \vdots \\ \tilde{z}_N \end{pmatrix} \right\|^2\right)$$

$$= \mathbb{E}\left(\left\| \begin{pmatrix} g_\sigma^{(x)}(\lambda_1) & & \\ & \ddots & \\ & & g_\sigma^{(x)}(\lambda_N) \end{pmatrix} \boldsymbol{V}^T \boldsymbol{z} \right\|^2\right)$$

$$= \mathbb{E}\left(\left\| \boldsymbol{V} \begin{pmatrix} g_\sigma^{(x)}(\lambda_1) & & \\ & \ddots & \\ & & g_\sigma^{(x)}(\lambda_N) \end{pmatrix} \boldsymbol{V}^T \boldsymbol{z} \right\|^2\right) = \mathbb{E}\left(\|g_\sigma^{(x)}(\boldsymbol{S})\boldsymbol{z}\|^2\right)$$

Estimation of $\mathfrak{K}_z$: Proof II

Notice that if ε is a white noise, its PSD is the vector $\mathbf{1} = (1, ..., 1)^T$. And therefore,

$$C_x = \sum_{k=0}^{N-1} g_\sigma^{(x)}(\lambda_k)^2 = \sum_{k=0}^{N-1} g_\sigma^{(x)}(\lambda_k)^2 \times 1 = \sum_{k=0}^{N-1} g_\sigma^{(x)}(\lambda_k)^2 \mathbb{E}(\tilde{\varepsilon}_k^2)$$

$$= \mathbb{E}\left(\left\| \begin{pmatrix} g_\sigma^{(x)}(\lambda_1) & & \\ & \ddots & \\ & & g_\sigma^{(x)}(\lambda_N) \end{pmatrix} \tilde{\varepsilon} \right\|^2 \right) = \mathbb{E}\left(\| g_\sigma^{(x)}(\boldsymbol{S})\varepsilon \|^2 \right)$$

Then we have :

$$\boxed{\widehat{\gamma}_{\boldsymbol{z}}(x) = \frac{\mathbb{E}\left(\| g_\sigma^{(x)}(\boldsymbol{S})\boldsymbol{z} \|^2 \right)}{\mathbb{E}\left(\| g_\sigma^{(x)}(\boldsymbol{S})\varepsilon \|^2 \right)}}$$

Back