Session 4.1: Models for geostatistical data and introduction to SPDE

Spatial and Spatio-Temporal Bayesian Models with `R-INLA`, University of São Paulo

29 September 2022

Learning Objectives

At the end of this session you should be able to:

- know the common models used for **geostatistical data**, i.e. Gaussian fields (GF);
- understand the basics of the Stochastic Partial Differential Equation (**SPDE**) approach;
- implement the SPDE approach using the `R-INLA` package.

The topics treated in this lecture can be found in **Section 6.4 -- 6.7** of the INLA book.

Outline

1. Introduction to spatial modeling for geostatistical data (based on GF)

2. Basics of the SPDE approach

3. The SPDE approach with `R-INLA`

Introduction to spatial modeling for geostatistical data (based on GF)

Geostatistical data

Definition | Examples

- The difference between models for **geostatistical** (or point referenced) data and the spatial models presented in Day 1 is that here we treat space as continuous, not discretised (areas).

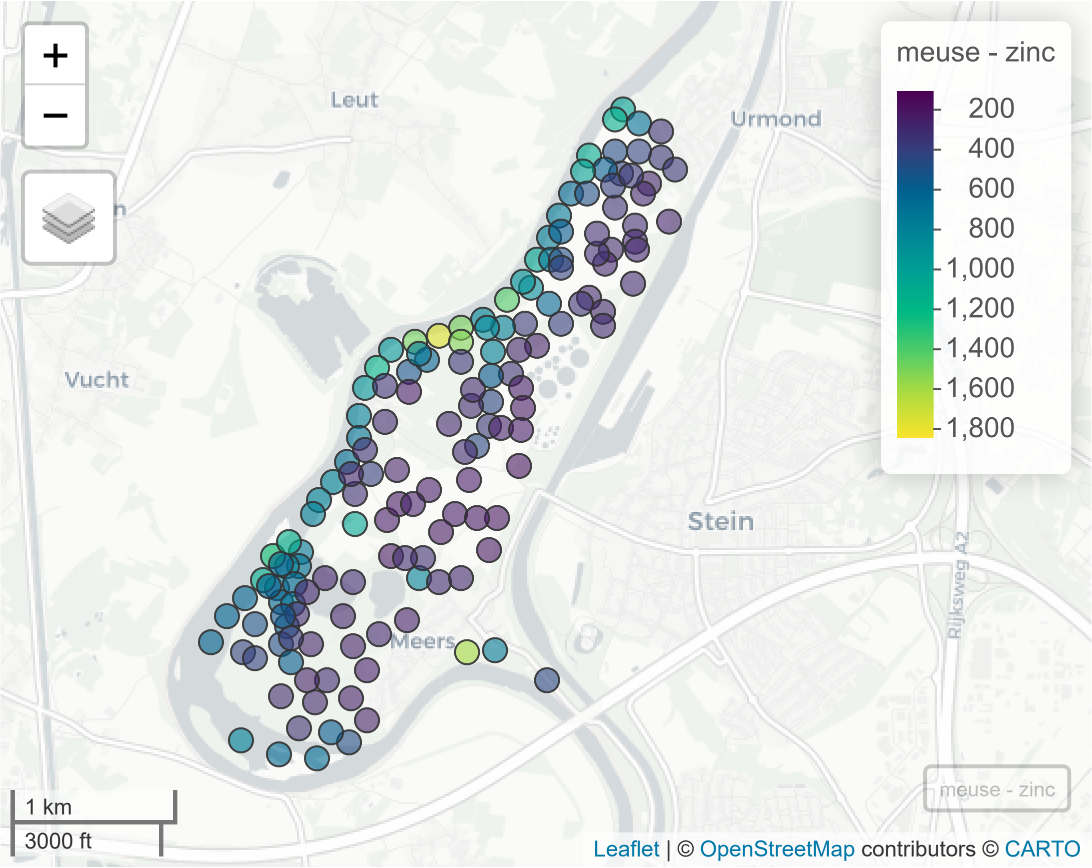
- We are concerned here with spatial data structures where the process of interest (response) is a spatial field $y(\boldsymbol{s})$, $\boldsymbol{s} \in \mathcal{D}$, i.e. real values stochastic process characterized by a spatial index $\boldsymbol{s}$ which varies **continuously** in the fixed domain $\mathcal{D}$.

- Data are measured (possibly with error) at n spatial locations $(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n)$ and are denoted by $\boldsymbol{y} = (y(\boldsymbol{s}_1), \ldots, y(\boldsymbol{s}_n)) = (y_1, \ldots, y_n)$.

Geostatistical data

Definition | Examples

- Examples:
  - in the field of environmental science: rainfall, air pollution concentrations, radioactive emission in soil, etc.,
  - in epidemiology when considering the risk of disease at different locations.



Aims

- To **reconstruct the spatial field** from a finite set of **noisy** observations taken at a finite number of spatial locations.
- To use the spatial dependence to **predict values** of the spatial field (together with associated uncertainty) at locations where there are no observations.
- The common methodological framework to geostatistical models is that of **Gaussian fields** (or processes) which are based on the multivariate Normal distribution.

Gaussian fields

- A spatial process $y(\boldsymbol{s})$ is a **Gaussian field** (GF) if for any $n \geq 1$ and for each set of locations $(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n)$, the vector $(y(\boldsymbol{s}_1), \ldots, y(\boldsymbol{s}_n))$ follows a multivariate Normal distribution with mean $\boldsymbol{\mu} = (\mu(\boldsymbol{s}_1), \ldots, \mu(\boldsymbol{s}_n))$ and spatially structured covariance matrix $\boldsymbol{\Sigma}$.

- The generic element of $\boldsymbol{\Sigma}$ is defined by a **spatial covariance function** $\mathcal{C}(\cdot, \cdot)$ such that $\Sigma_{ij} = \text{Cov}\left(y(\boldsymbol{s}_i), y(\boldsymbol{s}_j)\right) = \mathcal{C}(y(\boldsymbol{s}_i), y(\boldsymbol{s}_j))$.

Gaussian fields

- A spatial process $y(\boldsymbol{s})$ is a **Gaussian field** (GF) if for any $n \geq 1$ and for each set of locations $(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n)$, the vector $(y(\boldsymbol{s}_1), \ldots, y(\boldsymbol{s}_n))$ follows a multivariate Normal distribution with mean $\boldsymbol{\mu} = (\mu(\boldsymbol{s}_1), \ldots, \mu(\boldsymbol{s}_n))$ and spatially structured covariance matrix $\boldsymbol{\Sigma}$.

- The generic element of $\boldsymbol{\Sigma}$ is defined by a **spatial covariance function** $\mathcal{C}(\cdot, \cdot)$ such that $\Sigma_{ij} = \text{Cov}\left(y(\boldsymbol{s}_i), y(\boldsymbol{s}_j)\right) = \mathcal{C}(y(\boldsymbol{s}_i), y(\boldsymbol{s}_j))$.

- The spatial process is called **second-order stationary** if
  - $\boldsymbol{\mu}$ is constant (i.e., $\mu(\boldsymbol{s}_i) = \mu$ for each i)
  - the spatial covariance function depends only on the distance vector $(\boldsymbol{s}_i - \boldsymbol{s}_j) \in \mathbb{R}^2$, i.e. $\text{Cov}\left(y(\boldsymbol{s}_i), y(\boldsymbol{s}_j)\right) = \mathcal{C}(\boldsymbol{s}_i - \boldsymbol{s}_j)$.

Gaussian fields

- A spatial process $y(\boldsymbol{s})$ is a **Gaussian field** (GF) if for any $n \geq 1$ and for each set of locations $(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n)$, the vector $(y(\boldsymbol{s}_1), \ldots, y(\boldsymbol{s}_n))$ follows a multivariate Normal distribution with mean $\boldsymbol{\mu} = (\mu(\boldsymbol{s}_1), \ldots, \mu(\boldsymbol{s}_n))$ and spatially structured covariance matrix $\boldsymbol{\Sigma}$.

- The generic element of $\boldsymbol{\Sigma}$ is defined by a **spatial covariance function** $\mathcal{C}(\cdot, \cdot)$ such that $\Sigma_{ij} = \text{Cov}\left(y(\boldsymbol{s}_i), y(\boldsymbol{s}_j)\right) = \mathcal{C}(y(\boldsymbol{s}_i), y(\boldsymbol{s}_j))$.

- The spatial process is called **second-order stationary** if
  - $\boldsymbol{\mu}$ is constant (i.e., $\mu(\boldsymbol{s}_i) = \mu$ for each i)
  - the spatial covariance function depends only on the distance vector $(\boldsymbol{s}_i - \boldsymbol{s}_j) \in \mathbb{R}^2$, i.e. $\text{Cov}\left(y(\boldsymbol{s}_i), y(\boldsymbol{s}_j)\right) = \mathcal{C}(\boldsymbol{s}_i - \boldsymbol{s}_j)$.

- Moreover, a stationary process is **isotropic** if the covariance does not depend on the direction but just on the Euclidean distance $||\boldsymbol{s}_i - \boldsymbol{s}_j|| \in \mathbb{R}$.

Gaussian fields

- A spatial process $y(\boldsymbol{s})$ is a **Gaussian field** (GF) if for any $n \geq 1$ and for each set of locations $(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n)$, the vector $(y(\boldsymbol{s}_1), \ldots, y(\boldsymbol{s}_n))$ follows a multivariate Normal distribution with mean $\boldsymbol{\mu} = (\mu(\boldsymbol{s}_1), \ldots, \mu(\boldsymbol{s}_n))$ and spatially structured covariance matrix $\boldsymbol{\Sigma}$.

- The generic element of $\boldsymbol{\Sigma}$ is defined by a **spatial covariance function** $\mathcal{C}(\cdot, \cdot)$ such that $\Sigma_{ij} = \text{Cov}\left(y(\boldsymbol{s}_i), y(\boldsymbol{s}_j)\right) = \mathcal{C}(y(\boldsymbol{s}_i), y(\boldsymbol{s}_j))$.

- The spatial process is called **second-order stationary** if
  - $\boldsymbol{\mu}$ is constant (i.e., $\mu(\boldsymbol{s}_i) = \mu$ for each i)
  - the spatial covariance function depends only on the distance vector $(\boldsymbol{s}_i - \boldsymbol{s}_j) \in \mathbb{R}^2$, i.e. $\text{Cov}\left(y(\boldsymbol{s}_i), y(\boldsymbol{s}_j)\right) = \mathcal{C}(\boldsymbol{s}_i - \boldsymbol{s}_j)$.

- Moreover, a stationary process is **isotropic** if the covariance does not depend on the direction but just on the Euclidean distance $||\boldsymbol{s}_i - \boldsymbol{s}_j|| \in \mathbb{R}$.

- Several functions are available for the **spatial covariance function** (e.g. exponential, Matérn, spherical, etc.) parameterized by some parameters, e.g. spatial variance, range (Banerjee, Carlin, and Gelfand, 2015).

A common model for geostatistical (noisy) data

- Usually the following (mixed-effects) model is assumed

$$y(\boldsymbol{s}) = \mu(\boldsymbol{s}) + \xi(\boldsymbol{s}) + \epsilon(\boldsymbol{s})$$

where

- $\mu(\boldsymbol{s})$ is the so-called **large scale** component, including linear or non linear of covariates.
- $\xi(\boldsymbol{s})$ is a zero mean latent **Gaussian spatial process** commonly assumed to be stationary and isotropic with covariance function $Cov(\xi(\boldsymbol{s}_i), \xi(\boldsymbol{s}_j))$ which depends only on the distance between the locations.
- $\epsilon(\boldsymbol{s})$ represents the Gaussian **measurement error** (independent from $\xi(\boldsymbol{s})$) and its variance (σ^2_ϵ) is usually known as **nugget effect**.

A common model for geostatistical (noisy) data

- Usually the following (mixed-effects) model is assumed

$$y(\boldsymbol{s}) = \mu(\boldsymbol{s}) + \xi(\boldsymbol{s}) + \epsilon(\boldsymbol{s})$$

where

- $\mu(\boldsymbol{s})$ is the so-called **large scale** component, including linear or non linear of covariates.
- $\xi(\boldsymbol{s})$ is a zero mean latent **Gaussian spatial process** commonly assumed to be stationary and isotropic with covariance function $Cov(\xi(\boldsymbol{s}_i), \xi(\boldsymbol{s}_j))$ which depends only on the distance between the locations.
- $\epsilon(\boldsymbol{s})$ represents the Gaussian **measurement error** (independent from $\xi(\boldsymbol{s})$) and its variance (σ^2_ϵ) is usually known as **nugget effect**.

Given the data from n locations $(\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n)$ we have:

$$\begin{aligned} y(s_i) \mid \mu(\boldsymbol{s}_i), \xi(\boldsymbol{s}_i), \sigma^2_\epsilon &\sim \text{Normal}(\mu(\boldsymbol{s}_i) + \xi(\boldsymbol{s}_i), \sigma^2_\epsilon) \\ \boldsymbol{\xi} &\sim \text{GF}(\mathbf{0}, \boldsymbol{\Sigma}) \end{aligned}$$

where $\boldsymbol{\Sigma}$ is a **dense** matrix defined by a spatial covariance function $\mathcal{C}(||\boldsymbol{s}_i - \boldsymbol{s}_j||)$.

Matérn covariance function

Definition | The effect of range

The **Matérn covariance function** is defined by

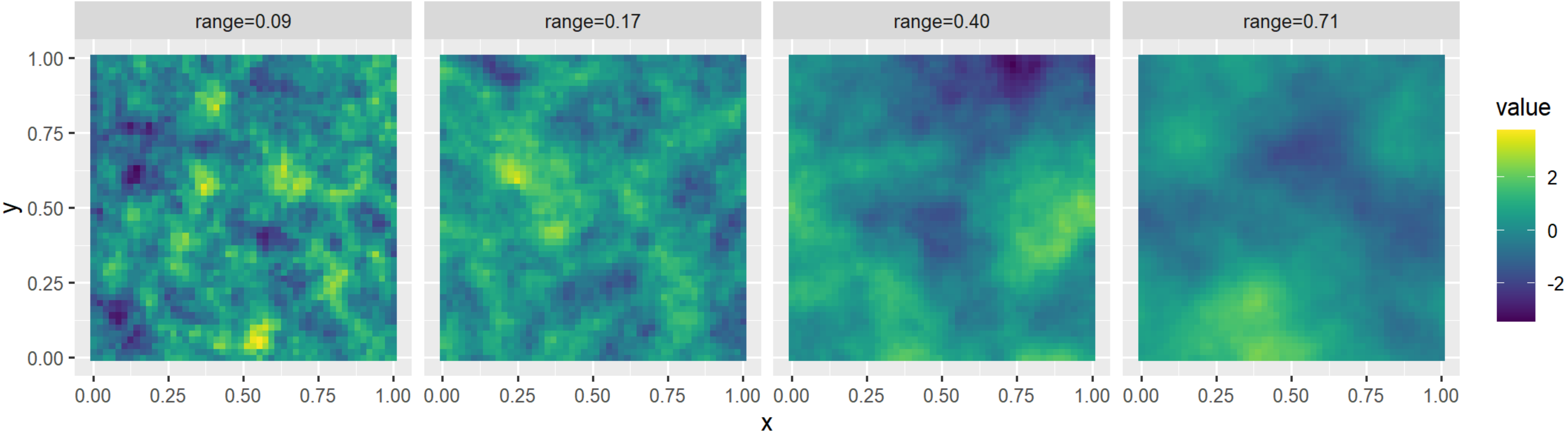
$$\mathrm{Cov}(\xi(\boldsymbol{s}_i), \xi(\boldsymbol{s}_j)) = \mathrm{Cov}(\xi_i, \xi_j) = \frac{\sigma^2}{\Gamma(\lambda)2^{\lambda-1}}(\kappa||\boldsymbol{s}_i - \boldsymbol{s}_j||)^\lambda K_\lambda(\kappa||\boldsymbol{s}_i - \boldsymbol{s}_j||)$$

where:

- $||\boldsymbol{s}_i - \boldsymbol{s}_j||$ is the Euclidean distance between two generic locations $\boldsymbol{s}_i, \boldsymbol{s}_j \in \mathbb{R}^2$,
- σ^2 is the marginal variance of the GF,
- K_λ denotes the modified Bessel function of second kind and order $\lambda > 0$, which measures the degree of **smoothness** of the process (it is usually kept fixed due to poor identifiability),
- $\kappa > 0$ is a **scale parameter** related to the **range** r, i.e. the distance at which the spatial correlation becomes almost null. Typically, the empirically derived definition for the range is $r = \frac{\sqrt{8\lambda}}{\kappa}$ (Lindgren, Rue, and Lindström, 2011), with r corresponding to the distance at which the spatial correlation is close to 0.1, for each $\lambda \geq 1/2$.
- The Matérn family is a very flexible class of covariance functions able to cover a wide range of spatial fields.

Definition | The effect of range

Simulation of different GFs with $\sigma = 1$ and different ranges:



Model based approach for estimation

- In the Bayesian framework, the classical approach for model estimation is Markov chain Monte Carlo methods (MCMC) considering that the likelihood function is a multivariate Gaussian distribution (see for example the `spBayes` and `spTimer` R packages).

- This requires to compute the Cholesky factorization of the dense covariance matrix $\boldsymbol{\Sigma}$ which is an operation of order n^3. This is known as **big n problem** (Banerjee, Carlin, and Gelfand, 2015).

- The **stochastic partial differential equation (SPDE) approach** (Lindgren, Rue, and Lindström, 2011) is an alternative to the use of MCMC: it represents the continuous spatial process $\xi(\boldsymbol{s})$ with Matérn covariance function using a discretely indexed spatial random process (i.e., a GMRF), which is characterized by a sparse precision matrix and enjoys computational benefits in terms of fast inference (the cost is typically of the order $n^{3/2}$ in $\mathbb{R}^2$).

Introduction to the SPDE approach

The SPDE approach: main references

Seminal paper | JSS paper | SPDE book | Other references

- Lindgren, F., Rue, H. and Lindstrom, J. (2011), An explicit link between Gaussian fields and Gaussian Markov random fields: the stochastic partial differential equation approach. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 73: 423-498. https://doi.org/10.1111/j.1467-9868.2011.00777.x

Journal of the Royal Statistical Society
SERIES B Statistical Methodology

J. R. Statist. Soc. B (2011)
73, *Part* 4, *pp.* 423–498

An explicit link between Gaussian fields and Gaussian Markov random fields: the stochastic partial differential equation approach

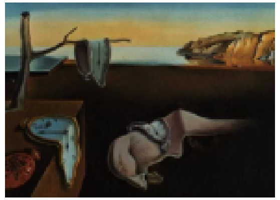
Finn Lindgren and Håvard Rue
Norwegian University of Science and Technology, Trondheim, Norway

and Johan Lindström
Lund University, Sweden

The SPDE approach: main references

Seminal paper | **JSS paper** | SPDE book | Other references

- Lindgren, F., & Rue, H. (2015). Bayesian Spatial Modelling with R-INLA. Journal of Statistical Software, 63(19), 1–25. https://doi.org/10.18637/jss.v063.i19



Journal of Statistical Software

January 2015, Volume 63, Issue 19. *http://www.jstatsoft.org/*

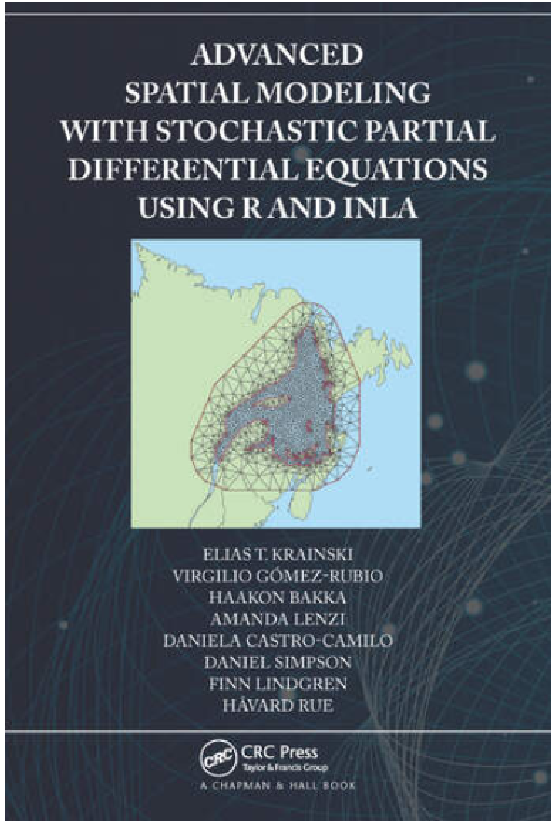
Bayesian Spatial Modelling with R-INLA

Finn Lindgren
University of Bath

Håvard Rue
Norwegian University of Science and Technology

Seminal paper | JSS paper | **SPDE book** | Other references

- E.T. Krainski, V. Gómez-Rubio, H. Bakka, A. Lenzi, D. Castro-Camilo, D. Simpson, F. Lindgren and H. Rue (2019) Advanced Spatial Modeling with Stochastic Partial Differential Equations Using R and INLA, CRC Press.
- https://becarioprecario.bitbucket.io/spde-gitbook/



The SPDE approach: main references

Seminal paper | JSS paper | SPDE book | **Other references**

- Lindgren F., Bolin D., Rue H., **The SPDE approach for Gaussian and non-Gaussian fields: 10 years and still running**, Spatial Statistics, Volume 50, 2022, https://doi.org/10.1016/j.spasta.2022.100599
- Bakka H, Rue H, Fuglstad GA, et al. **Spatial modeling with R-INLA: A review.** WIREs Comput Stat. 2018; 10:e1443. https://doi.org/10.1002/wics.1443
- Moraga P, **Geospatial Health Data. Modeling and Visualization with R-INLA and Shiny**, 2020, CRC press, www.paulamoraga.com/book-geospatial-info/

The SPDE approach for Matérn GF

- The starting point is the **linear fractional stochastic partial differential equation** (SPDE)

$$(\kappa^2 - \Delta)^{\alpha/2}(\tau\xi(\boldsymbol{s})) = \mathcal{W}(\boldsymbol{s})$$

where $\boldsymbol{s} \in \mathbb{R}^d$, Δ is the Laplacian operator, $\alpha > 0$ is the **smoothness** term, $\kappa > 0$ is the scale parameter, τ controls the variance and $\mathcal{W}(\boldsymbol{s})$ is a Gaussian spatial white noise process (with unit variance).

- Whittle in 1954 showed that the exact and stationary solution to this SPDE is the stationary Gaussian field $\xi(\boldsymbol{s})$ with Matérn covariance function.
- Lindgren, Rue, and Lindström (2011) represent the solution of the SPDE using the finite element method (this is possible only for some values of the smothness parameter).

- In $\mathbb{R}^2$ the link between the SPDE parameters τ, α, κ and the Matérn function parameters $\sigma^2, \lambda, \kappa$ is given by

$$\lambda = \alpha - 1 \qquad\qquad \sigma^2 = \frac{\Gamma(\lambda)}{\Gamma(\alpha)(4\pi)\kappa^{2\lambda}\tau^2}$$

- In `R-INLA` the default value for the smoothness parameter is $\alpha = 2 \rightarrow \lambda = 1$. Consequently

$$r = \sqrt{8\lambda}/\kappa = \sqrt{8}/\kappa \text{ (see slide 9)} \qquad\qquad \sigma^2 = 1/(4\pi\kappa^2\tau^2)$$

Piecewise linear approximation

General result | 1D | 2D

- The solution to the SPDE can be approximated through a **basis function representation** defined on a **triangulation** of the spatial domain

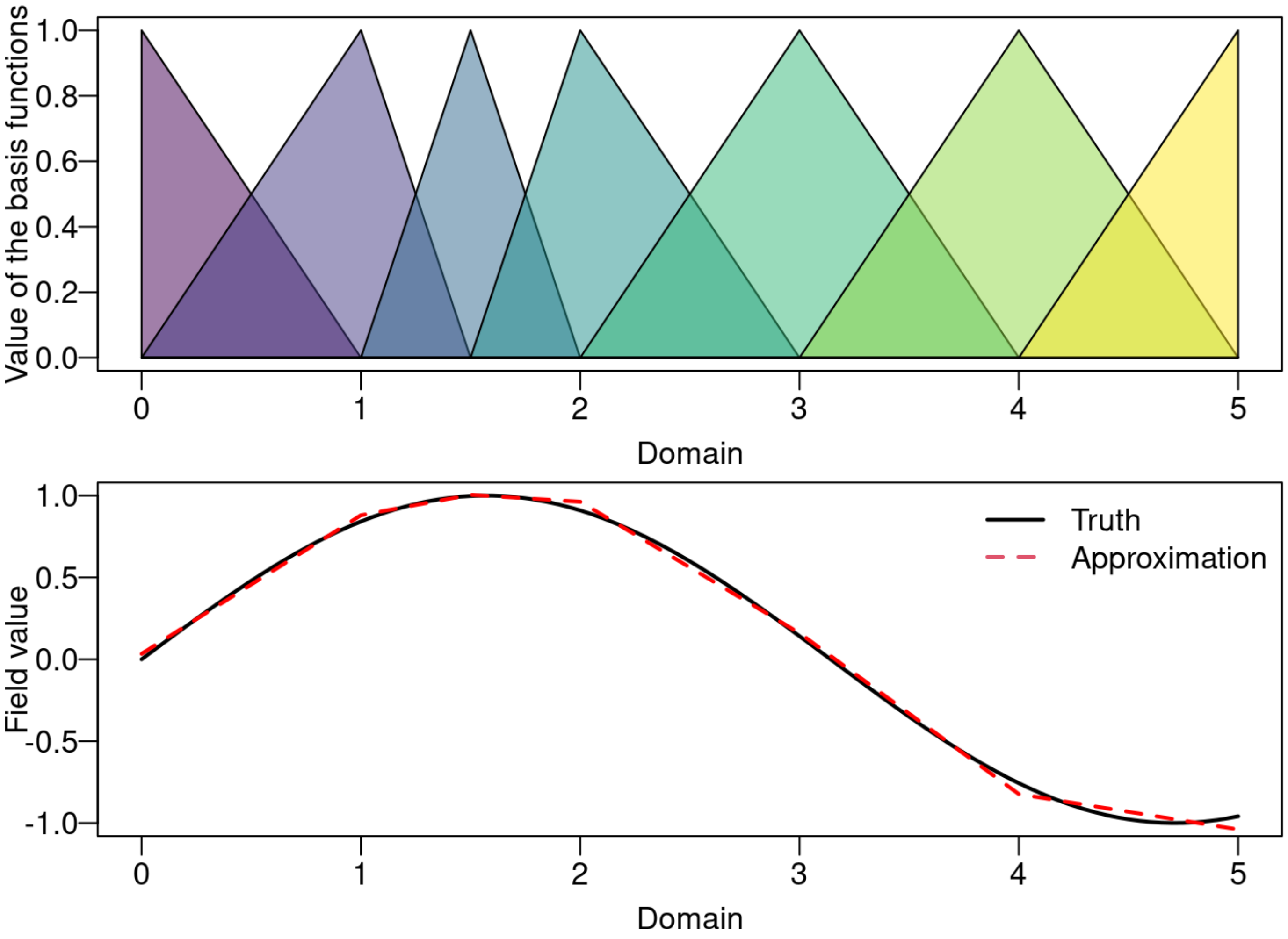
$$\xi(\boldsymbol{s}) = \sum_{g=1}^{G} \varphi_g(\boldsymbol{s}) \tilde{\xi}_g$$

where G is the total number of triangulation vertices, $\{\varphi_g\}$ is the set of (deterministic) basis functions and $\{\tilde{\xi}_g\}$ are zero mean Gaussian distributed weights.

- In order to obtain a Markov structure, the basis functions are chosen to be piecewise linear in each triangle, i.e. φ_g is 1 at vertex g and 0 at all other vertices.

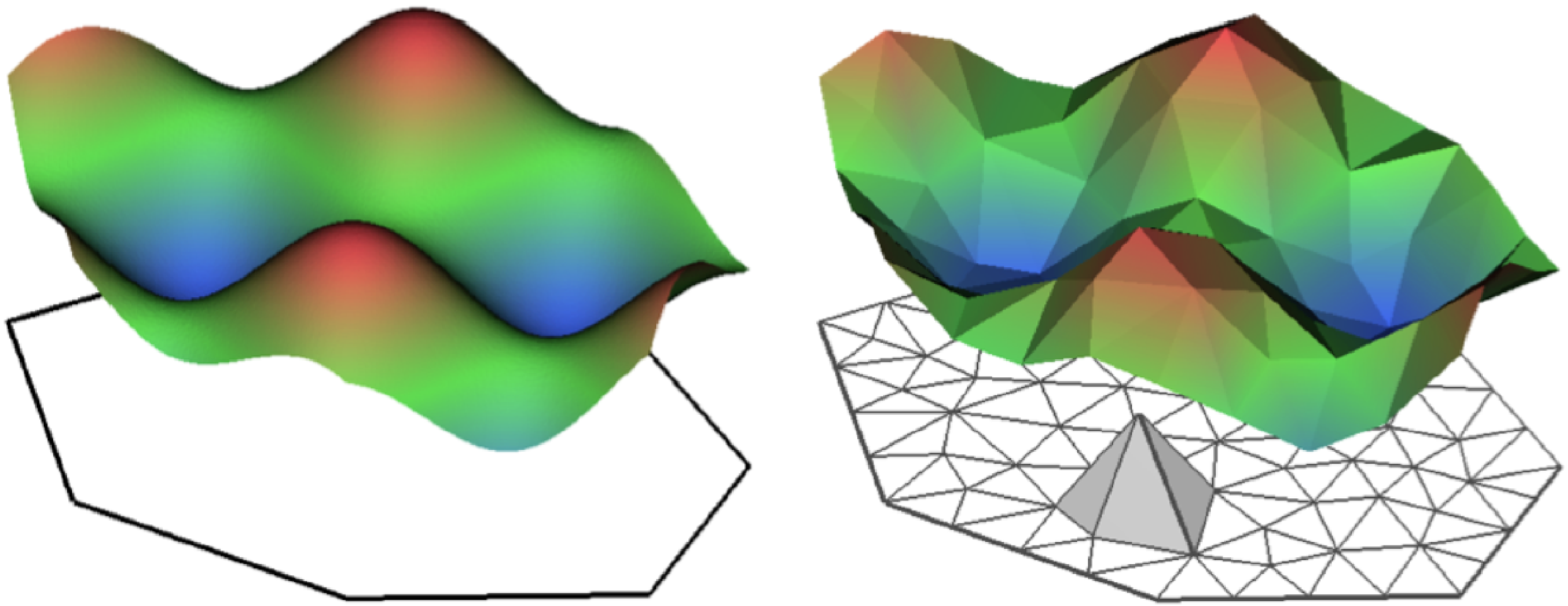
Piecewise linear approximation

General result | **1D** | 2D



Piecewise linear approximation

General result | 1D | **2D**



Piecewise linear approximation

- Using Neumann boundary conditions, it follows that the precision matrix $\boldsymbol{Q}$ for the Gaussian weight vector $\tilde{\boldsymbol{\xi}} = \{\tilde{\xi}_1, \ldots, \tilde{\xi}_G\}$ is given by

$$\boldsymbol{Q} = \tau^2 \left(\kappa^4 \boldsymbol{C} + 2\kappa^2 \boldsymbol{G} + \boldsymbol{G}\boldsymbol{C}^{-1}\boldsymbol{G} \right)$$

where

- the generic element of the diagonal matrix $\boldsymbol{C}$ is $C_{ii} = \int \varphi_i(\boldsymbol{s})\mathrm{d}\boldsymbol{s}$,
- the generic element of the sparse matrix $\boldsymbol{G}$ is $G_{ij} = \int \nabla\varphi_i(\boldsymbol{s})\nabla\varphi_j(\boldsymbol{s})\mathrm{d}\boldsymbol{s}$ (where ∇ denotes the gradient),

The precision matrix $\boldsymbol{Q}$, whose elements depend on τ and κ, is sparse and consequently $\boldsymbol{\xi}$ is a GMRF distributed as $\mathrm{Normal}(\boldsymbol{0}, \boldsymbol{Q}^{-1})$: it represents the approximated solution to the SPDE.

Take home message

- Do the modelling using GF and the computations using the GMRF representation (computational advantages thanks to algorithms for sparse matrices):

$$\boldsymbol{\xi} \sim \text{Normal}(\mathbf{0}, \boldsymbol{\Sigma}) \Longrightarrow \tilde{\boldsymbol{\xi}} \sim \text{Normal}\left(\mathbf{0}, \boldsymbol{Q}^{-1}\right)$$

The SPDE approach with `R-INLA`

SPDE toy example

Data Plot

We use the SPDEtoy dataset , consisting in 200 simulated values for the variable y which refer to as many randomly sampled locations in the unit square area delimited by the points $(0, 0)$ and $(1, 1)$ and with coordinates given by s1 and s2.

```
> library(INLA)
> data(SPDEtoy)
> dim(SPDEtoy)
```

```
[1] 200   3
```

```
> head(SPDEtoy, n=3)
```

```
          s1         s2         y
1 0.08265625 0.05640625 11.521206
2 0.61230625 0.91680625  5.277960
3 0.16200625 0.35700625  6.902959
```

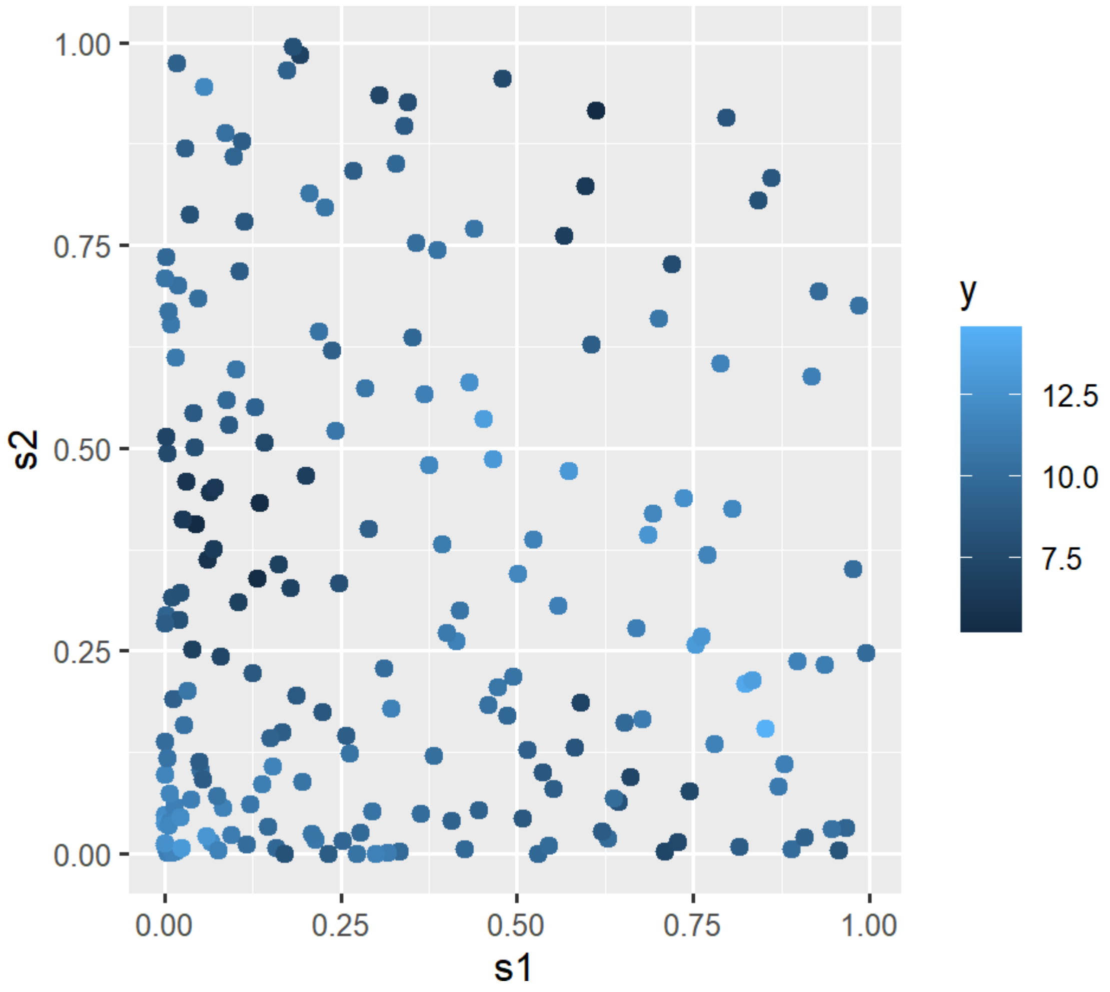
```
> summary(SPDEtoy$y)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  5.210   8.688  10.023   9.858  11.307  14.600
```

SPDE toy example

Data | **Plot**

```
> library(tidyverse)
> library(INLA)
>
> SPDEtoy %>%
+   ggplot() +
+   geom_point(aes(s1,s2, col=y),size=2)
```



SPDE toy example: model for simulation

- The model used for simulating the `SPDEtoy` data assumes that the distribution of the observation y_i is

$$y_i \mid \eta_i, \sigma^2_\epsilon \sim \text{Normal}(\eta_i, \sigma^2_e) \qquad i = 1, \ldots, 200$$

where σ^2_e is the variance of the zero mean measurement error e_i which is supposed to be Gaussian iid.

- The response mean, which coincides with the **linear predictor**, is defined as

$$\eta_i = b_0 + \xi_i$$

and includes the intercept b_0 and a random effect represented by ξ_i, which is the realization of the latent GF $\xi(\boldsymbol{s}) \sim \text{MVNormal}(\mathbf{0}, \boldsymbol{\Sigma})$. The covariance matrix $\boldsymbol{\Sigma}$ is defined by the Matérn spatial covariance function.

- The parameter values chosen for simulating the data are: $b_0 = 10$, $\sigma^2_e = 0.3$, $\sigma^2 = 5$, $\kappa = 7$, $r = \frac{\sqrt{8}}{\kappa} = 0.404$.

The SPDE representation and the projector matrix

- Using the SPDE basis function representation, the linear predictor η_i can be rewritten as

$$\eta_i = b_0 + \sum_{g=1}^{G} \varphi_g(\boldsymbol{s}_i)\tilde{\xi}_g$$

where $\varphi_g(\boldsymbol{s}_i)$ is the value of the g-th basis function evaluated in $\boldsymbol{s}_i$.

- More generally it is possible to express the linear predictor as

$$\eta_i = b_0 + \sum_{g=1}^{G} A_{ig}\tilde{\xi}_g$$

with $A_{ig} = \varphi_g(\boldsymbol{s}_i)$ being the generic element of the sparse matrix $\boldsymbol{A}$ (known as **projector matrix**) which maps the GMRF $\tilde{\boldsymbol{\xi}}$ from the G triangulation vertices to the n observation locations. This allows the SPDE model to be treated as standard indexed random effects.

Mesh

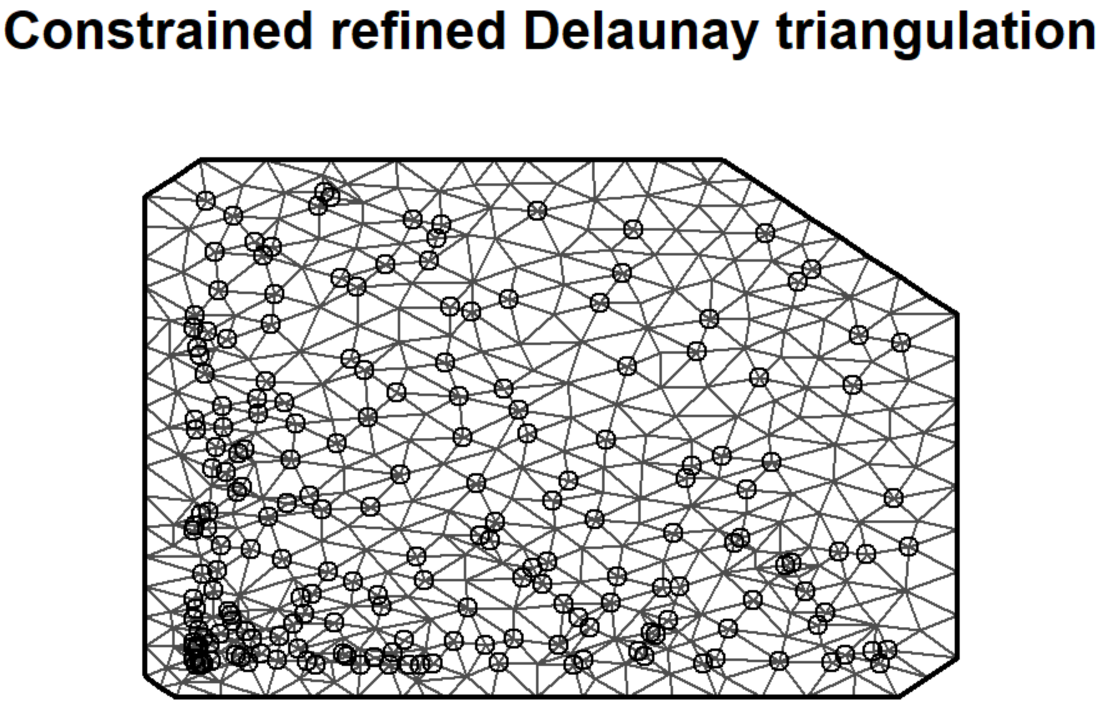
- The SPDE approach is based on a **triangulation** of the spatial domain given by the **mesh**.
- The definition of the mesh is a **trade-off** between the accuracy of the GMRF representation and computational costs, both depending on the number of vertices used in the triangulation: the bigger the number of mesh triangles, the finer the GF approximation but the higher the computational costs.
- To create the mesh in `R-INLA` we use the helper function `inla.mesh.2d` with the following options:
  - `loc` or `loc.domain`: specify information about the spatial domain
  - `max.edge`: specify the largest allowed triangle edge length. If a vector of two values is provided, the spatial domain is divided into an inner and an outer area whose triangle resolution is specified by `max.edge` (the higher the value for `max.edge` the lower the resolution and the accuracy).

Mesh: changing max.edge

mesh0 | mesh0 inlabru | mesh1 | mesh2

```
> coords <- as.matrix(SPDEtoy[,1:2])
> mesh0 <- inla.mesh.2d(loc = coords,
+                       max.edge = 0.1)
```

```
> plot(mesh0)
> points(coords)
```

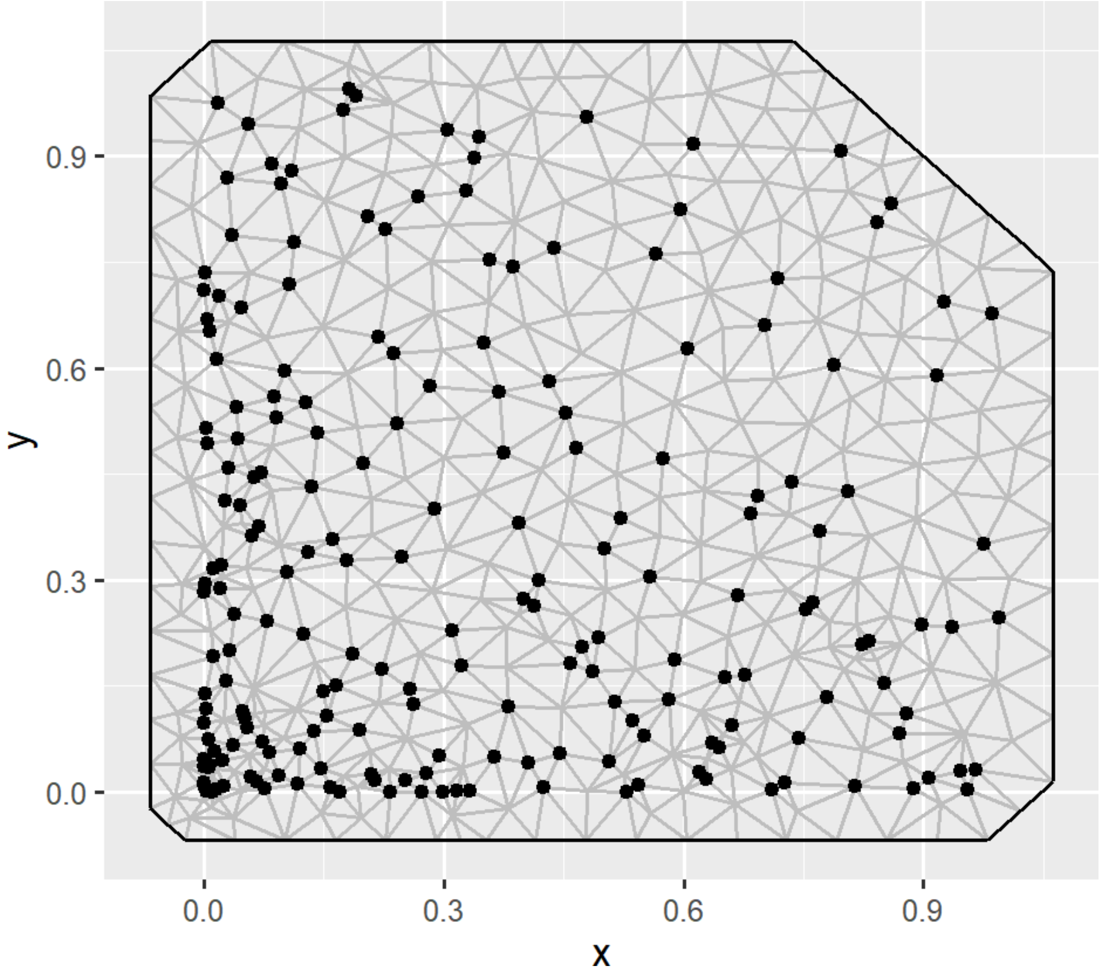


Mesh: changing max.edge

mesh0 | **mesh0 inlabru** | mesh1 | mesh2

```
> coords <- as.matrix(SPDEtoy[,1:2])
> mesh0 <- inla.mesh.2d(loc = coords,
+                       max.edge = 0.1)
```

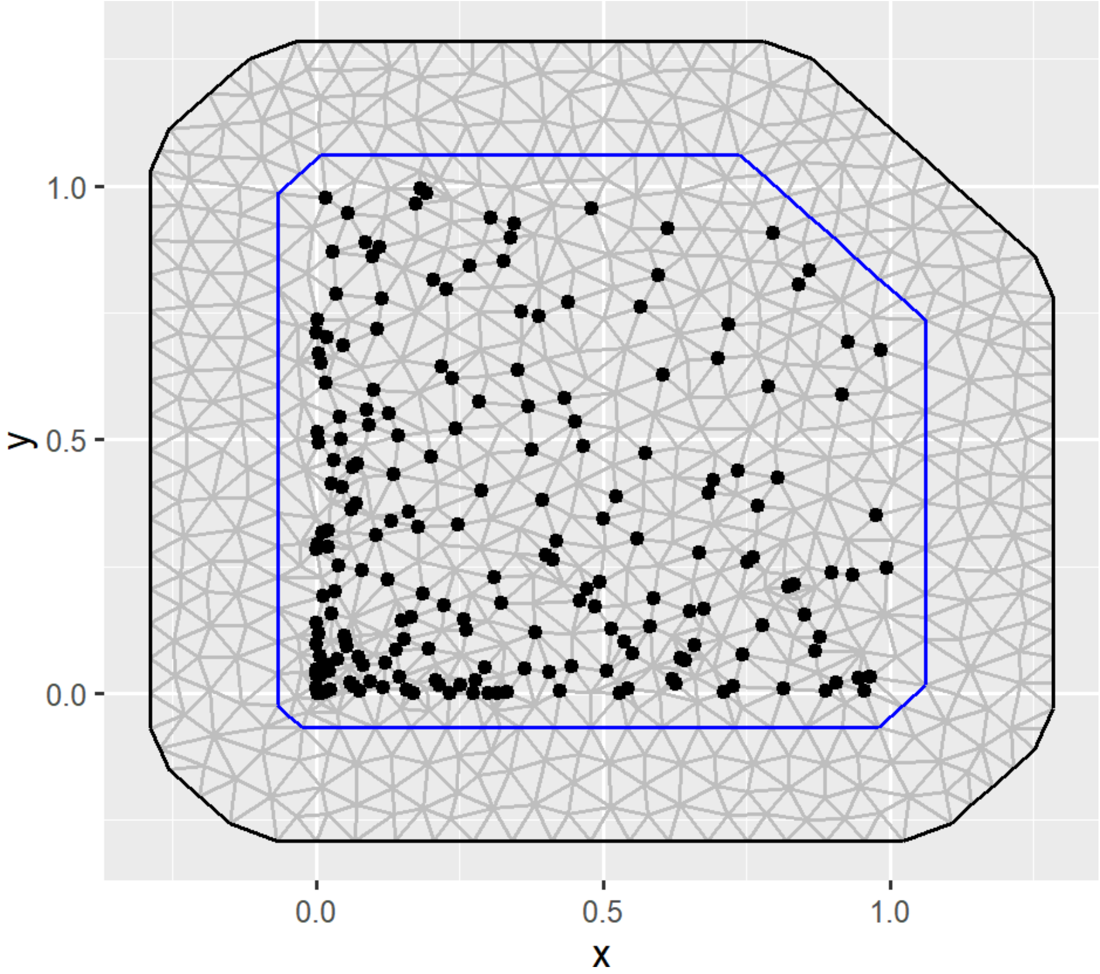
```
> library(tidyverse)
> library(inlabru)
> ggplot() +
+   gg(mesh0) +
+   geom_point(data = data.frame(coords),
+              aes(s1, s2))
```



mesh0 mesh0 inlabru **mesh1** mesh2

```
> mesh1 <- inla.mesh.2d(loc = coords,
+                        max.edge = c(0.1, 0.1))
```

```
> ggplot() +
+   gg(mesh1) +
+   geom_point(data = data.frame(coords),
+              aes(s1, s2))
```

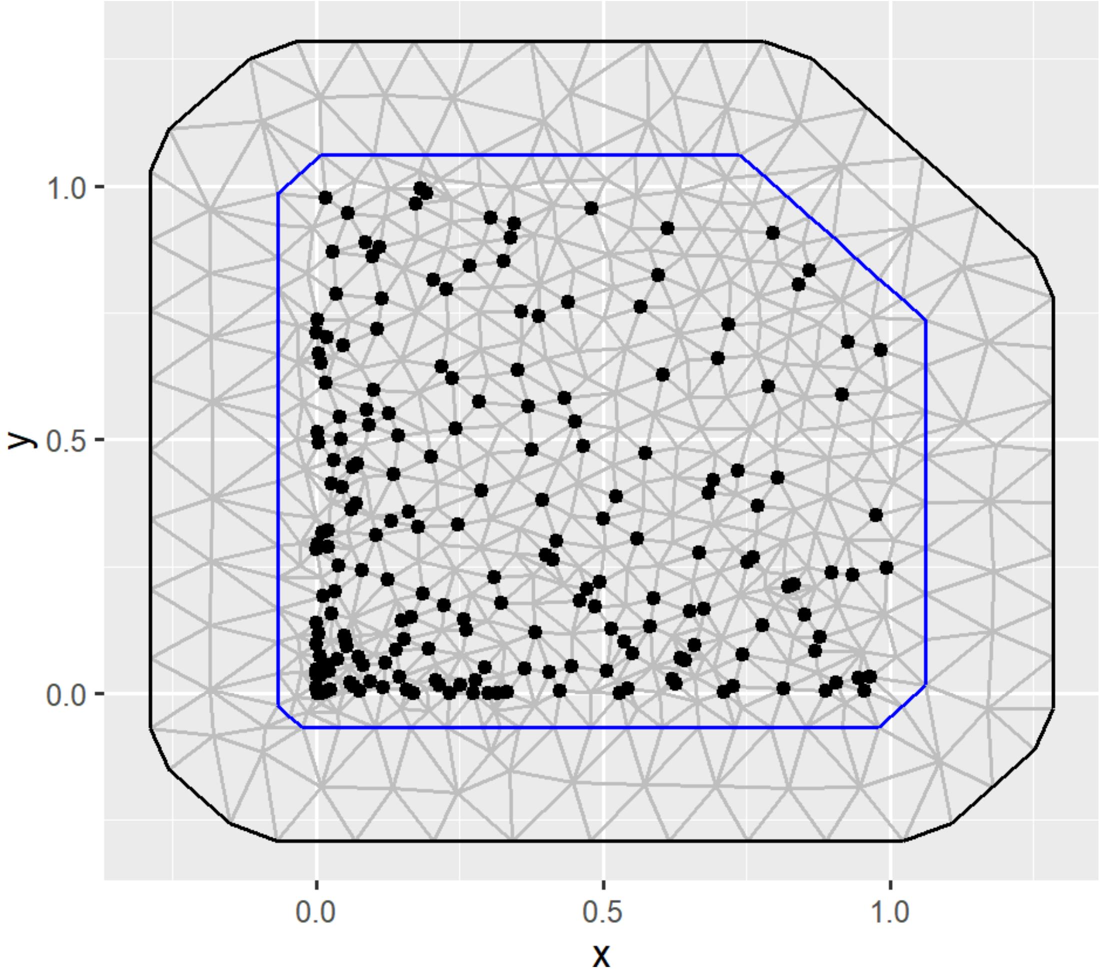


Mesh: changing max.edge

mesh0 | mesh0 inlabru | mesh1 | mesh2

```
> mesh2 <- inla.mesh.2d(loc = coords,
+                       max.edge = c(0.1, 0.2))
```

```
> ggplot() +
+   gg(mesh2) +
+   geom_point(data = data.frame(coords),
+              aes(s1, s2))
```



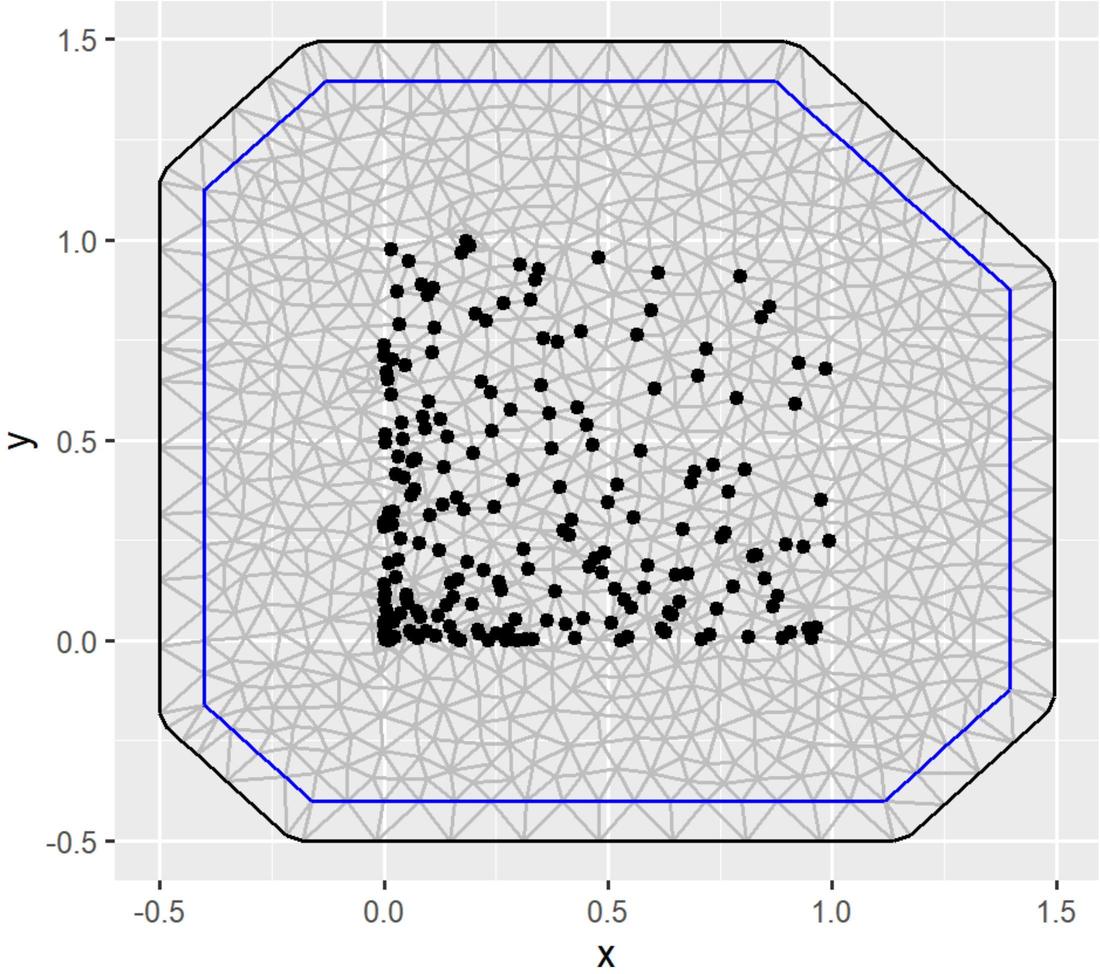
Mesh optional arguments: `offset`

mesh3 | mesh4

The option `offset` of the `inla.mesh.2d` function can be used to define how much the domain should be extended in the inner and outer part. The default values are `offset = c(-0.05, -0.15)`.

```
> mesh3 <- inla.mesh.2d(loc = coords,
+                       max.edge = c(0.1, 0.2),
+                       offset = c(0.4,0.1))
```

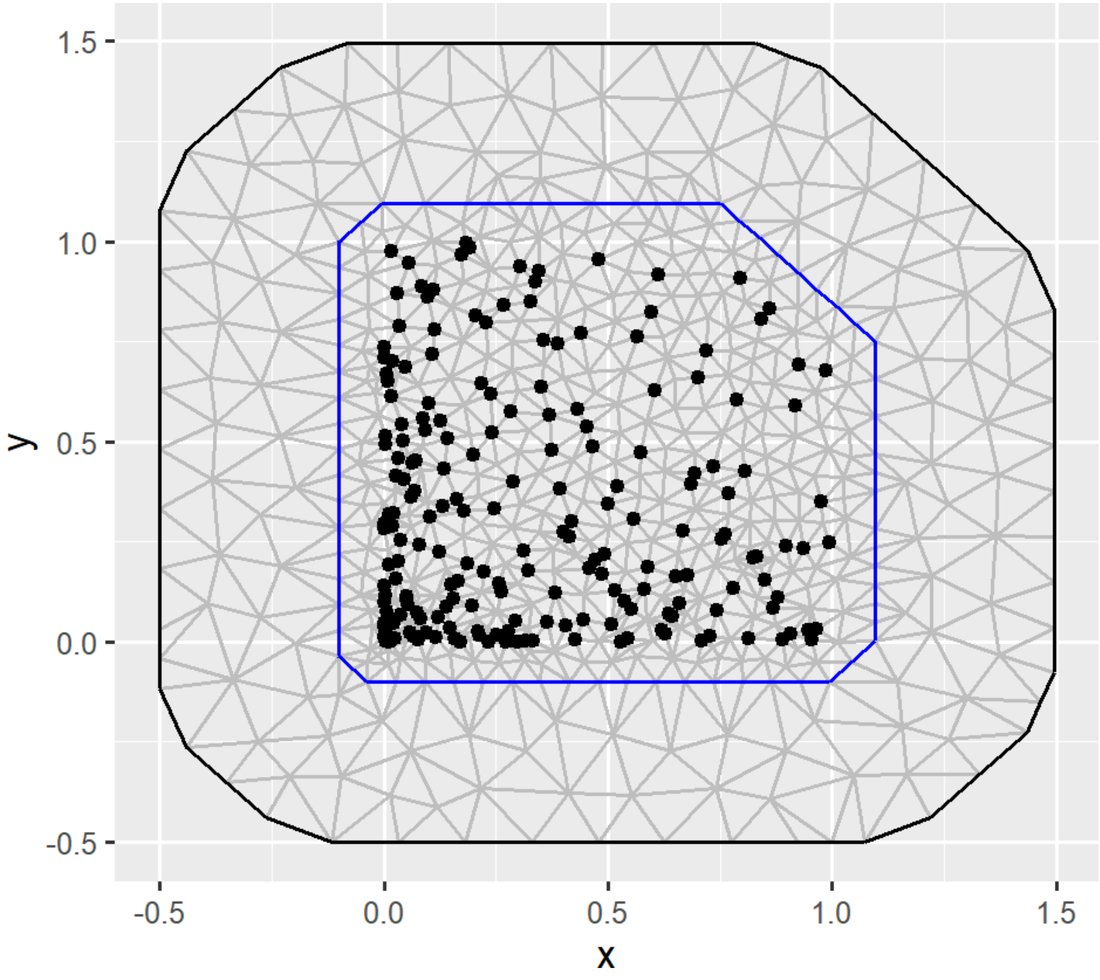
```
> ggplot() +
+   gg(mesh3)+
+   geom_point(data = data.frame(coords),
+              aes(s1, s2))
```



mesh3 | mesh4

```
> mesh4 <- inla.mesh.2d(loc = coords,
+                       max.edge = c(0.1, 0.2),
+                       offset = c(0.1,0.4))
```

```
> ggplot() +
+   gg(mesh4) +
+   geom_point(data = data.frame(coords),
+              aes(s1, s2))
```



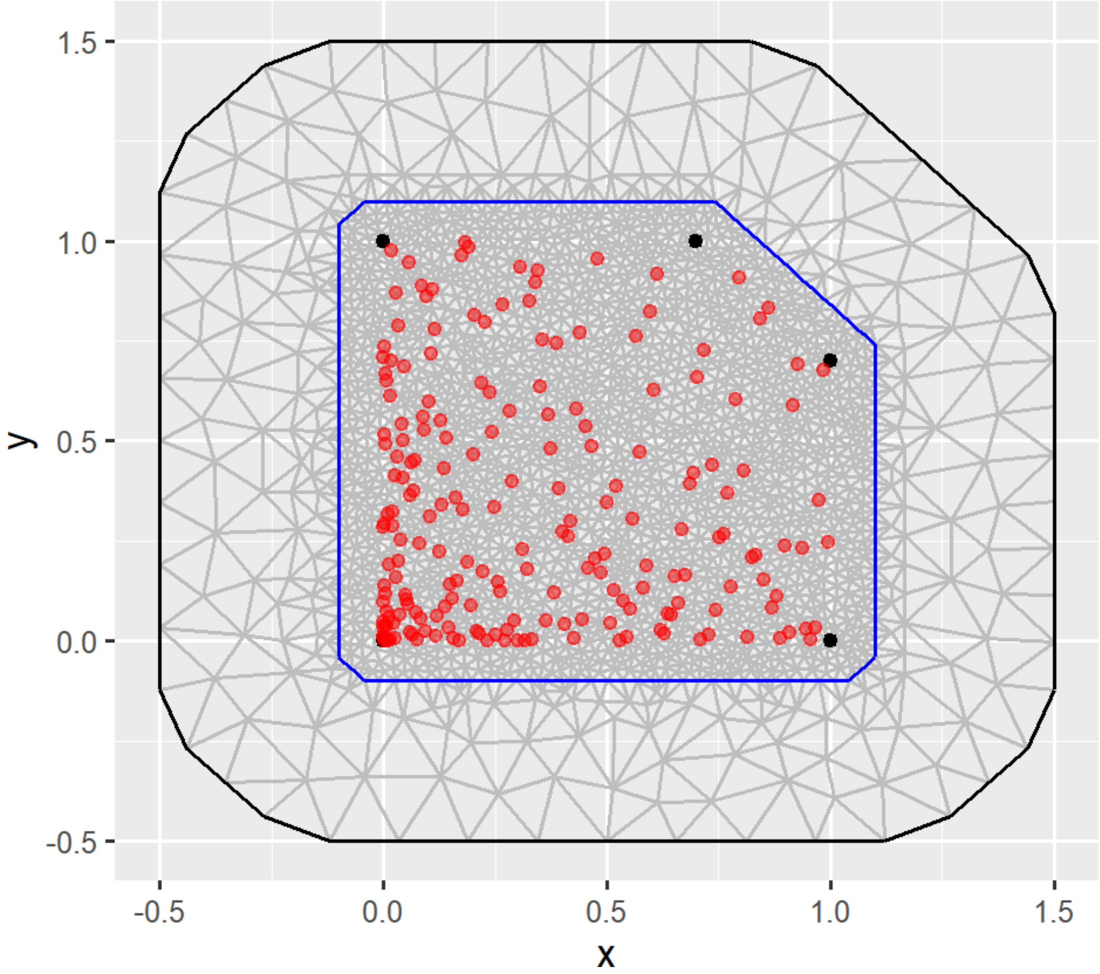
Mesh optional arguments cutoff

domain | mesh5 | mesh6

Instead of the point coordinates, it is possible to use other point locations used to determine the domain extent.

```
> domain <- matrix(cbind(c(0,1,1,0.7,0),
+                        c(0,0,0.7,1,1)), ncol=2)
> mesh5domain <- inla.mesh.2d(loc.domain = domain,
+                      max.edge = c(0.04, 0.2),
+                      offset = c(0.1, 0.4))
```

```
> ggplot() +
+   gg(mesh5domain) +
+   geom_point(data = data.frame(domain), aes(X1,
+     geom_point(data = data.frame(coords),
+                aes(s1, s2), col = "red", alpha =
```

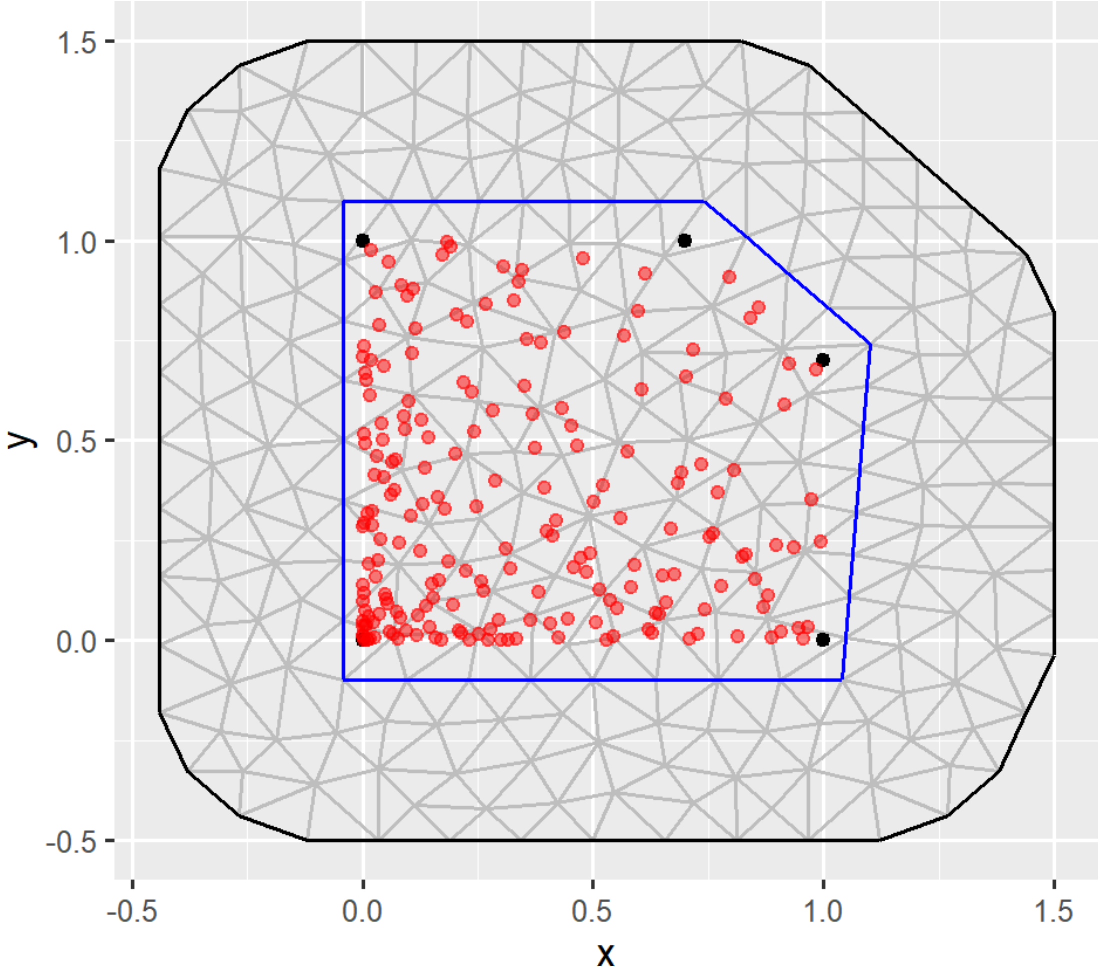


domain | mesh5 | mesh6

The option `cutoff` can be used to avoid building too many small triangles around clustered data locations (the default value is equal to 0).

```
> mesh5 <- inla.mesh.2d(loc.domain = domain,
+                       max.edge = c(0.04, 0.2),
+                       cutoff = 0.5,
+                       offset = c(0.1, 0.4))
```

```
> ggplot() +
+   gg(mesh5) +
+   geom_point(data = data.frame(domain),aes(X1,
+     geom_point(data = data.frame(coords),
+                aes(s1, s2), col = "red", alpha =
```

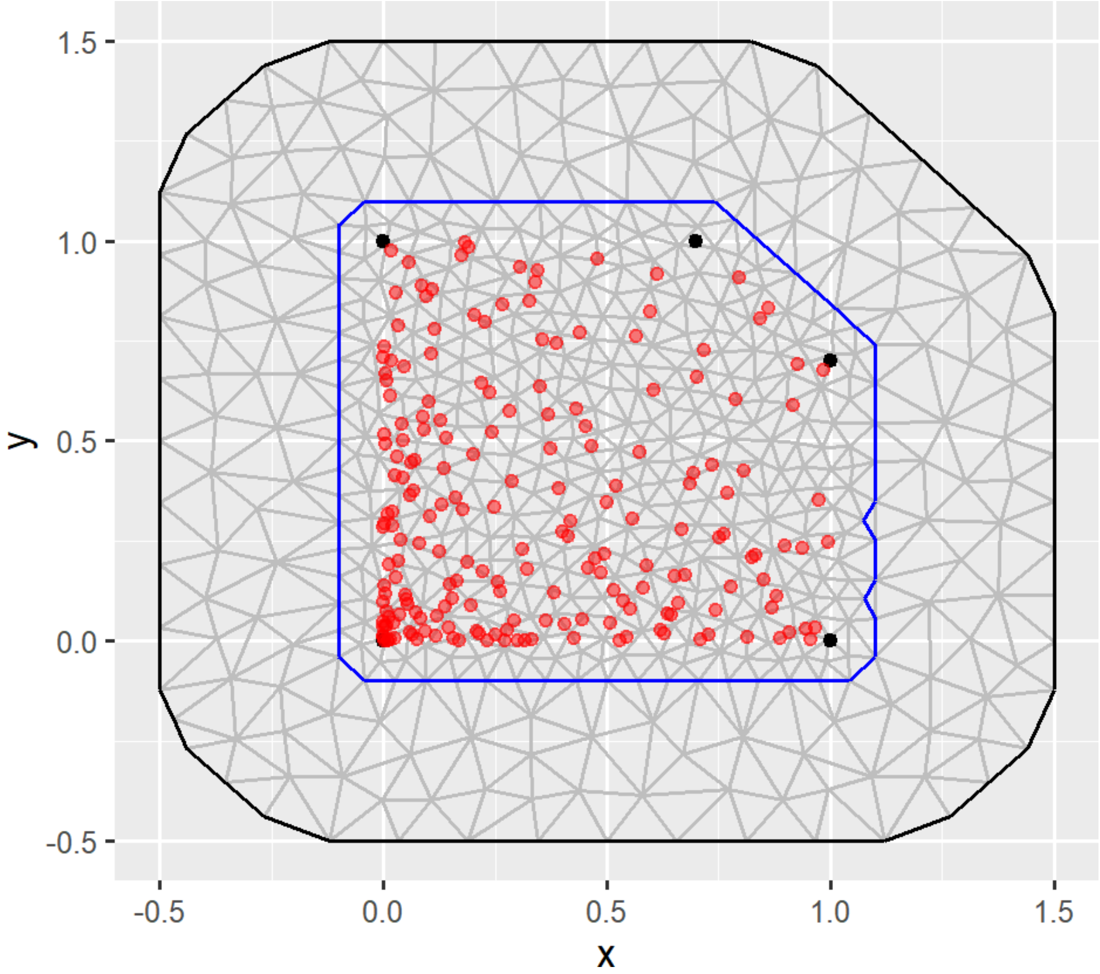


Mesh optional arguments cutoff

domain mesh5 **mesh6**

```
> mesh6 <- inla.mesh.2d(loc.domain = domain,
+                       max.edge = c(0.04, 0.2),
+                       cutoff = 0.05,
+                       offset = c(0.1, 0.4))
```

```
> ggplot() +
+   gg(mesh6) +
+   geom_point(data = data.frame(domain),
+              aes(X1, X2)) +
+     geom_point(data = data.frame(coords),
+              aes(s1, s2), col = "red", alpha =
```

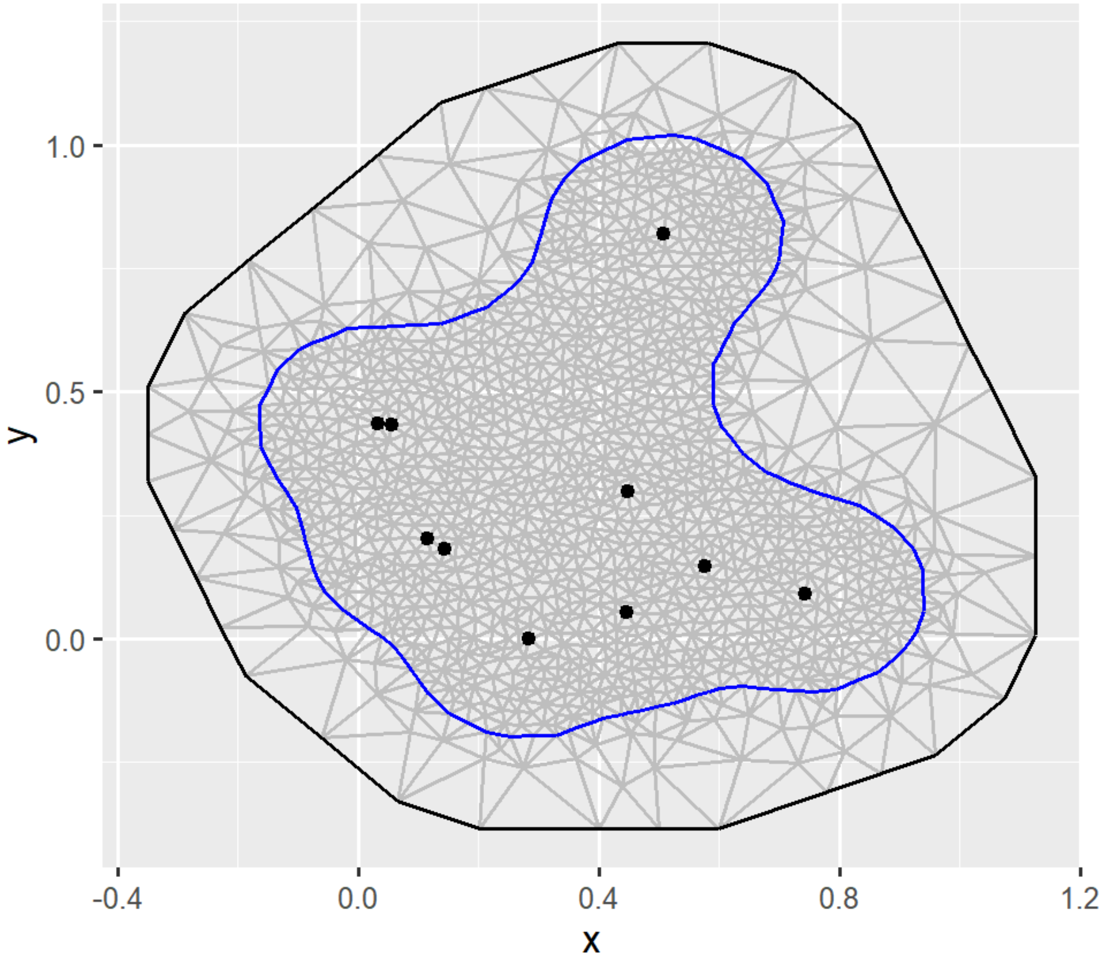


Mesh: non convex hull

A feature in `R-INLA` named `inla.nonconvex.hull` makes it possible to compute a non convex hull to be included as boundary in the mesh construction. This can be particularly useful when the shape of the domain is of some importance.

```
> set.seed(44)
> loc = matrix(runif(20), 10, 2)
>
> boundary = inla.nonconvex.hull(loc,
+                                 convex=0.2)
> meshNC <- inla.mesh.2d(loc = loc,
+                        boundary = boundary,
+                        max.edge = c(0.04, 0.2))
```

```
> ggplot() +
+   gg(meshNC)+
+   geom_point(data = data.frame(loc),
+              aes(X1, X2))
```

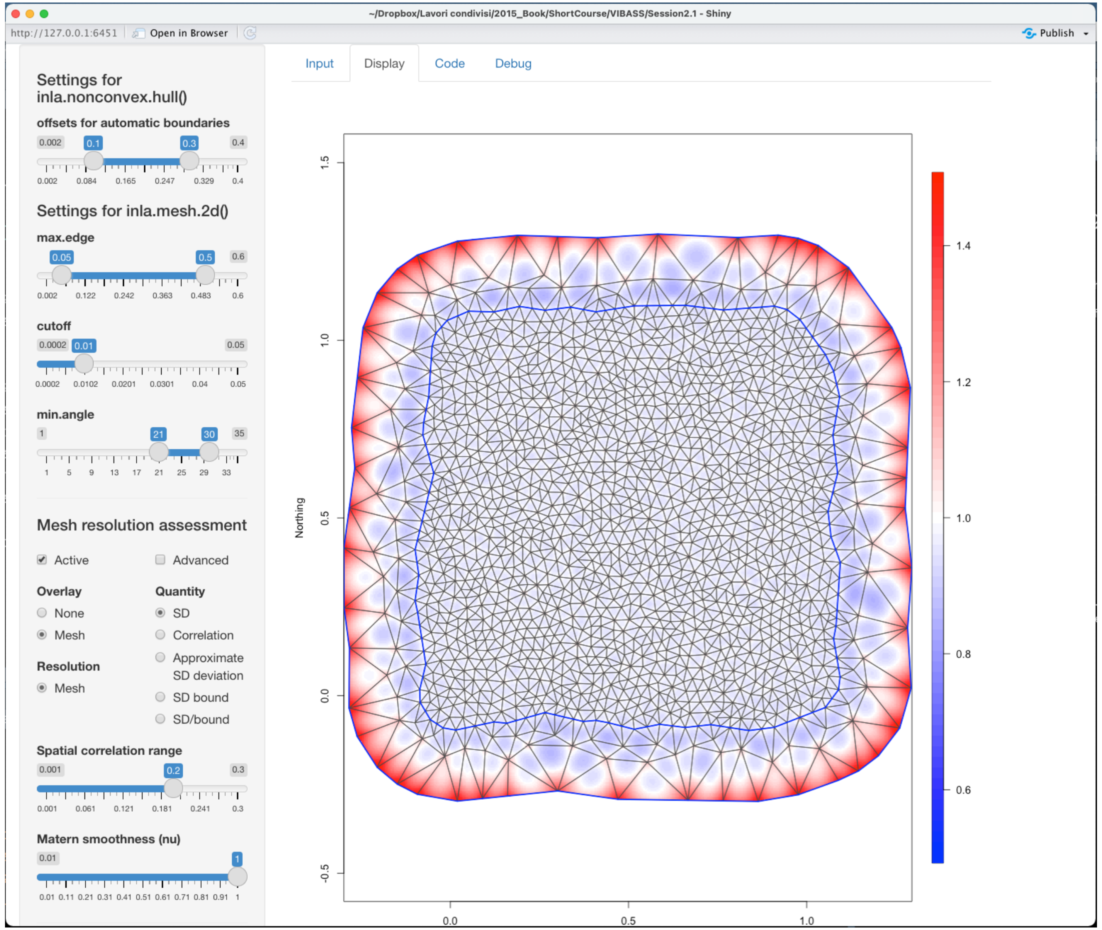


Mesh Shiny App

By running

```
> meshbuilder()
```

it is possible to access the interactive Shiny app for exploring triangle mesh constructions for use with SPDE models.



Projector matrix for mesh6

```
> A.est6 <- inla.spde.make.A(mesh = mesh6,
+                            loc = coords)
> dim(A.est6)
```

```
[1] 200 549
```

No more than 3 elements in each line are non-zero:

```
> table(rowSums(A.est6>0))
```

```
  3
200
```

The sum of each row is one:

```
> table(rowSums(A.est6))
```

```
  1
200
```

There are some columns whose sum is zero corresponding to triangles with no point location inside:

```
> table(colSums(A.est6) > 0)
```

```
FALSE  TRUE
  313   236
```

Model fitting

- The vector of parameters is defined as $\boldsymbol{\theta} = \{\tilde{\boldsymbol{\xi}}, b_0\}$ with hyper-parameter vector $\boldsymbol{\psi} = (\sigma_e^2, r, \tau)$, where $\tau = 1/\sigma$ and r (depending on κ) are the Matérn covariance function parameters.
- In `R-INLA` the default **internal representation** for the SPDE parameters is $\log(\tau) = \theta_1$ and $\log(\kappa) = \theta_2$, with θ_1 and θ_2 being given two independent Normal prior distributions.

1. Create the Matérn **SPDE model object** (by default $\alpha = 2$):

```
> spde = inla.spde2.matern(mesh = mesh6)
> spde$n.spde
```

```
[1] 549
```

2. In `R-INLA` the Matérn GF is part of the linear predictor and is specified in the `formula` environment using a proper specification for `f()`. Define the **linear predictor** through the `formula`:

```
> formula = y ~ -1 + intercept + f(spatial.field, model = spde)
```

where `spatial.field` is a proper index variable and `spde` is the model created previously with `inla.spde2.matern`. Note that the intercept is removed and is added manually in the linear predictor.

Model fitting

3. **Fit the model** as usual using the `inla` function:

```
> output6 <- inla(formula,
+                 data = list(y = SPDEtoy$y,
+                             intercept = rep(1,spde$n.spde),
+                             spatial.field = 1:spde$n.spde),
+                 control.predictor = list(A = A.est6, compute = TRUE))
```

Note that the projector matrix is passed to `inla` through `control.predictor`. Moreover, with the option `compute = TRUE` we ask for the computation of the marginals of the linear predictor.

Exploring the output: fixed effects and hyperparameters

```
> output6$summary.fixed[,c("mean","0.025quant","0.975quant")]
```

```
              mean 0.025quant 0.975quant
intercept 9.505305   8.057478   10.85346
```

```
> output6$summary.hyperpar[,c("mean","0.025quant","0.975quant")]
```

```
                                             mean 0.025quant 0.975quant
Precision for the Gaussian observations  2.877449   2.021960   3.945614
Theta1 for spatial.field                -4.039005  -4.353252  -3.714526
Theta2 for spatial.field                 2.105643   1.654185   2.540652
```

Exploring the output: spatial parameters

If we are interested in the posterior summaries of the spatial parameters on the scale of the variance $\sigma^2 = 1/\tau$ and range r (instead of the internal scale regarding $\theta_1 = \log(\tau)$ and $\theta_2 = \log(\kappa)$) we use

```
> output6.field <- inla.spde2.result(inla = output6,
+                                    name = "spatial.field",
+                                    spde = spde)
```

The resulting list contains the following elements:

```
> names(output6.field)
```

```
 [1] "summary.values"                 "marginals.values"
 [3] "summary.hyperpar"               "summary.theta"
 [5] "summary.log.tau"                "summary.log.kappa"
 [7] "summary.log.variance.nominal"   "summary.log.range.nominal"
 [9] "marginals.theta"                "marginals.log.tau"
[11] "marginals.log.kappa"            "marginals.log.variance.nominal"
[13] "marginals.log.range.nominal"    "marginals.tau"
[15] "marginals.kappa"                "marginals.variance.nominal"
[17] "marginals.range.nominal"
```

Exploring the output

The posterior mean of σ^2 and the range r can be obtained by typing

```
> inla.emarginal(function(x) x, output6.field$marginals.variance.nominal[[1]])
```

```
[1] 3.956856
```

```
> inla.emarginal(function(x) x, output6.field$marginals.range.nominal[[1]])
```

```
[1] 0.3531205
```

Also the other standard INLA functions can be applied to the marginal posteriors:

```
> inla.zmarginal(output6.field$marginals.range.nominal[[1]])
```

```
Mean               0.35312
Stdev              0.0803778
Quantile  0.025 0.223822
Quantile  0.25  0.295445
Quantile  0.5   0.343047
Quantile  0.75  0.399982
Quantile  0.975 0.538124
```

PC prior for the Matérn model

- Instead of `inla.spde2.matern` it is possible to use `inla.spde2.pcmatern` for creating an `inla.spde2` model object using a PC prior for the range r and the marginal standard deviation σ (Simpson, Rue, Riebler, Martins, and Sorbye, 2017).

- The prior for σ is such that

$$Pr(\sigma > \sigma_0) = p$$

  and this requires to specify σ_0 and p with the option `prior.sigma = c(sigma0,p)`.

- The prior for r is such that

$$Pr(r < r_0) = p$$

  and this requires to specify r_0 and p with the option `prior.range = c(r0,p)`.

For example:

```
> spde = inla.spde2.pcmatern(mesh,
+                            prior.range = c(0.01,0.1),
+                            prior.sigma = c(100,0.1))
```

References

Banerjee, S., B. Carlin, and A. Gelfand (2015). *Hierarchical Modeling and Analysis for Spatial Data*. Monographs on Statistics and Applied Probability. New York: Chapman and Hall.

Lindgren, F., H. Rue, and J. Lindström (2011). "An explicit link between Gaussian fields and Gaussian Markov random fields: the stochastic partial differential equation approach (with discussion)". In: *J. R. Statist. Soc. B* 73.4, pp. 423-498.

Simpson, D., H. Rue, A. Riebler, et al. (2017). "Penalising Model Component Complexity: A Principled, Practical Approach to Constructing Priors". In: *Statistical Science* 32.1, pp. 1 - 28. DOI: 10.1214/16-STS576. URL: https://doi.org/10.1214/16-STS576.