AStA Adv Stat Anal manuscript No.
(will be inserted by the editor)

Spatio-temporal modeling of particulate matter concentration through the SPDE approach

Michela Cameletti · Finn Lindgren · Daniel Simpson · Håvard Rue

the date of receipt and acceptance should be inserted later

Abstract In this work we consider a hierarchical spatio-temporal model for particulate matter (PM) concentration in the North-Italian region Piemonte. The model involves a Gaussian Field (GF), affected by a measurement error, and a state process characterized by a first order autoregressive dynamic model and spatially correlated innovations. This kind of model is well discussed and widely used in the air quality literature thanks to its flexibity in modeling the effect of relevant covariates (i.e. meteorological and geographical variables) as well as time and space dependence. However, Bayesian inference - through Markov chain Monte Carlo (MCMC) techniques - can be a challenge due to convergence problems and heavy computational loads. In particular, the computational issue refers to the infeasibility of linear algebra operations involving the big dense covariance matrices which occur when large spatio-temporal datasets are present. The main goal of this work is to present an effective estimating and spatial prediction strategy for the considered spatio-temporal model. This proposal consists in representing a GF with Matérn covariance function as a Gaussian Markov Random Field (GMRF) through the Stochastic Partial Differential Equations (SPDE) approach. The main advantage of moving from a GF to a GMRF stems from the good computational properties that the latter enjoys. In fact, GMRFs are defined by sparse matrices that allow for computationally effective numerical methods. Moreover, when dealing with Bayesian inference for GMRFs, it is possible to adopt the Integrated Nested Laplace Approximation (INLA) algorithm as an alternative to MCMC methods giving rise to additional computational advantages. The implementation of the SPDE approach through the `R`-library `INLA` (`www.r-inla.org`)

M. Cameletti
Dip. di Matematica, Statistica, Informatica e Applicazioni, Università di Bergamo, Bergamo, Italy, E-mail: michela.cameletti@unibg.it

F. Lindgren · D. Simpson · H. Rue
Dept. of Mathematical Sciences, Norwegian University of Science and Technology, Trondheim, Norway, E-mail: {Havard.Rue, Finn.Lindgren, Daniel.Simpson}@math.ntnu.no

is illustrated with reference to the Piemonte PM data. In particular, providing the step-by-step R-code, we show how it is easy to get prediction and probability of exceedence maps in a reasonable computing time.

Keywords hierarchical models · Integrated Nested Laplace Approximation · particulate matter PM_{10} · covariance functions · Gaussian fields · Gaussian Markov random fields

1 Introduction

Many environmental phenomena, even if defined continuously over a region and in time, can be monitored and measured only at a limited number of spatial locations and time points. This is the case, for example, of air pollutant concentration, meteorological fields (temperature, precipitation, wind velocity, etc.) as well as geohydrological and oceanographic variables (soil moisture, wave height, etc.). In the geostatistical approach (see for example Cressie 1993; Gelfand et al 2010; Cressie and Wikle 2011), data coming from monitoring networks are assumed to be realizations of a continuously indexed spatial process (*random field*) changing in time denoted by

$$Y(\boldsymbol{s},t) \equiv \{y(\boldsymbol{s},t) : (\boldsymbol{s},t) \in \mathcal{D} \subseteq \mathbb{R}^2 \times \mathbb{R}\}.$$

These realizations are used to make inference about the process and to predict it at desired locations. Usually, we deal with a Gaussian field (GF) that is completely specified by its mean and spatio-temporal covariance function $Cov\left(y(\boldsymbol{s},t), y(\boldsymbol{s}',t')\right) = \sigma^2\mathcal{C}((\boldsymbol{s},t),(\boldsymbol{s}',t'))$, defined for each $(\boldsymbol{s},t)$ and $(\boldsymbol{s}',t')$ in $\mathbb{R}^2 \times \mathbb{R}$. Moreover, the process is second-order stationary if its mean is constant and the spatio-temporal covariance function depends on the locations and time points only through the spatial distance vector $\mathbf{h} = (\boldsymbol{s} - \boldsymbol{s}') \in \mathbb{R}^2$ and the temporal lag $l = (t - t') \in \mathbb{R}$.

Even if a GF is easily defined directly through its first and second moments, its implementation suffers from the so-called "big n problem" (Banerjee et al 2004, page 387), that arises especially in case of large datasets in space and time. This problem is related to the computational costs of linear algebra operations required for model fitting and spatial interpolation and prediction. In fact, these computations involve dense covariance matrices, defined through the spatio-temporal covariance function $\sigma^2\mathcal{C}(\cdot,\cdot)$, whose dimension is given by the number of observations at all spatial locations and time points. Besides, this computational challenge gets worse in the Bayesian inference framework when linear algebra operations with dense matrices are computed for each iteration of the MCMC algorithm.

For facing the "big n problem" in the recent literature some solutions have been suggested such as, for example, covariance tapering, predictive process models and low rank kriging (see Furrer et al 2006; Banerjee et al 2008; Cressie and Johannesson 2008). These proposals generally try to reduce the dimension or simplify the structure of the dense covariance matrix of the GF.

In this work we consider a different approach that consists in representing a continously indexed GF with Matérn covariance function as a discretely indexed random process, i.e. a Gaussian Markov Random Field (GMRF, see Rue and Held 2005 for a complete discussion). This proposal is based on the work of Lindgren et al (2011), where an explicit link between GFs and GMRFs - formulated as a basis function representation - is provided through the Stochastic Partial Differential Equations (SPDE) approach. The key point is that the spatio-temporal covariance function and the dense covariance matrix of a GF are substituted, respectively, by a neighbourhood structure and by a sparse precision matrix, that together define a GMRF. The advantage of moving from a GF to a GMRF stems from the good computational properties that the latter enjoys. In fact, GMRFs are defined by a precision matrix with a sparse structure for which it is possible to use computationally effective numerical methods, especially for fast matrix factorization (see Rue and Held 2005). Moreover, when dealing with Bayesian inference for GMRFs, it is possible to make use of the Integrated Nested Laplace Approximation (INLA) algorithm proposed by Rue et al (2009) as an alternative to MCMC methods for latent Gaussian field models. The most outstanding advantage of INLA is computational because it produces almost immediately accurate approximations to posterior distributions, also in case of complex models. Thus, the joint use of the SPDE approach together with the INLA algorithm is a candidate for being a powerful solution in overcoming the computational issues related to GF modeling.

The main goal of this paper is to illustrate the implementation of the SPDE approach through the `R`-library `INLA` (`www.r-inla.org`) focusing on the motivating problem of particulate matter concentration in the North-Italian region Piemonte and on the following spatio-temporal model:

$$
\begin{aligned}
y(\boldsymbol{s},t) &= \boldsymbol{z}(\boldsymbol{s},t)\boldsymbol{\beta} + \xi(\boldsymbol{s},t) + \varepsilon(\boldsymbol{s},t) \\
\xi(\boldsymbol{s},t) &= a\xi(\boldsymbol{s},t-1) + \omega(\boldsymbol{s},t).
\end{aligned}
$$

In brief, the equations define a hierarchical model characterized by a GF $y(\boldsymbol{s},t)$ built from covariate information $z(\boldsymbol{s},t)$, measurement error $\varepsilon(\boldsymbol{s},t)$, and a first order autoregressive dynamic model for the latent process $\xi(\boldsymbol{s},t)$ with spatially correlated innovations $\omega(\boldsymbol{s},t)$. This kind of model is well discussed and widely used in the air quality literature thanks to its flexibity in modeling the effect of relevant covariates (i.e. meteorological and geographical variables) as well as time and space dependence (e.g. Cocchi et al 2007; Cameletti et al 2011; Sahu 2011).

The structure of the paper is as follows. In Section 2 we introduce our motivating problem regarding PM_{10} (particulate matter with an aerodynamic diameter of less than 10 μm) in Piemonte region, Italy. Here, after discussing the available data, we describe the geostatistical spatio-temporal model sketched out above. In Section 3 we provide the essential details for understanding the link between GFs and GMRFs: firstly we introduce the basics about GMRFs and secondly the SPDE approach with the basis function representation in a as

simple as possible way. We refer to Lindgren et al (2011) and Rue et al (2009) for all the details. In Section 4 we explain how the SPDE approach works for the considered spatio-temporal model, also with reference to parameter estimation and spatial prediction. Finally, in Section 5 we return to the analysis of the PM_{10} data described in Section 2. In particular, the section is devoted to the `R`-library `INLA` and provides a step-by-step description of the code required for the implementation of the SPDE approach for the considered case study, including spatial prediction and simple model validation. A discussion ends the paper, including also a comparison in terms of prediction capability and computational costs with a similar model presented in Cameletti et al (2011) and implemented with MCMC algorithms for the same PM_{10} data.

2 A hierarchical spatio-temporal model for air quality data

For some countries in southern Europe air pollution is an environmental emergency due to the adverse effects that high levels of pollutant concentrations could have on human health and the ecosystems. With regard to PM_{10}, the situation is particularly critical in the Po river basin located in northern Italy between the Alps and the Appenines. In this area the annual and daily limit values fixed by the European Union for human health protection (see EU Council Directive 1999/30/EC) are periodically exceeded. As a consequence, the population is exposed to pollution levels that can cause a multitude of harmful consequences, ranging from minor effects on the cardio-respiratory system to premature mortality (Samet et al 2000; Samoli et al 2008). The particular situation of Po valley is related to the complex orography of the area. In fact, the shelter effect of the Alps strongly influences meteorological phenomena that, in turn, have a major role in dispersion processes, removal mechanisms, and chemical formation of atmospheric particles. Moreover, the Po plain is characterized by urbanized areas where the most important emission sources of primary PM_{10} and secondary precursor pollutants are located, such as industrial sites and main roads with high levels of traffic.

In this context environmental agencies have to assess air quality in order to take proper and effective actions for improving the situation of the most polluted zones. Thus, continuous maps of PM_{10} concentration are required. To this aim, we propose a hierarchical spatio-temporal model able to catch the complex spatio-temporal dynamics of PM_{10} concentration, including also meteorological and geographical covariates. In particular, we consider Piemonte region which is situated in the western part of the Po valley.

2.1 PM_{10} data and covariates for Piemonte region

We analyze daily PM_{10} concentration measured by the Piemonte region monitoring network during the winter season October 2005 - March 2006 for a total of $T = 182$ days (the data are provided by the information system called

AriaWeb Regione Piemonte). In particular, we consider $d = 24$ monitoring stations for estimation purposes (see red triangles in Figure 1) and 10 sites for validation purposes (see blue dots in Figure 1). For an exploratory analysis of the PM_{10} data refer to Cameletti et al (2011). Moreover, the environmental

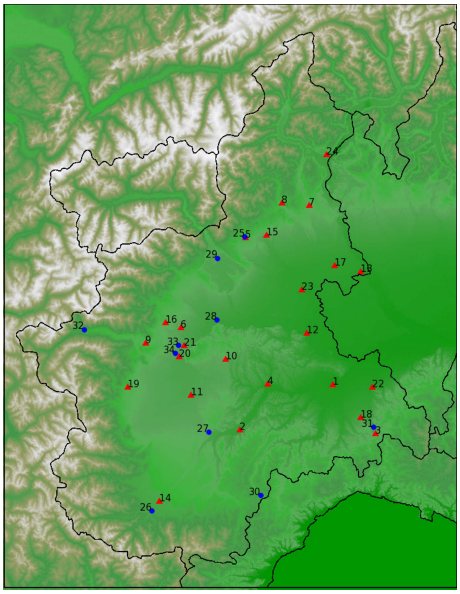

Fig. 1 Locations of the 24 PM_{10} monitoring sites (red triangles) and 10 validation stations (blue dots).

agency of Piemonte region (Arpa Piemonte) provides a set of covariates which are defined on a 4×4 km regular grid and with a hourly temporal resolution (Finardi et al 2008). As fully described in Cameletti et al (2011), by means of a preliminary analysis we have selected the following covariates, computing some daily synthesis and taking for each location the value of the grid pixel where the station lies: daily mean wind speed (WS, m/s), daily maximum mixing height ($HMIX$, m), daily precipitation (P, mm), daily mean temperature ($TEMP$, $^\circ K$) and daily emissions (EMI, g/s). Moreover, we consider altitude (A, m) and spatial geographic coordinates ($UTMX$ and $UTMY$, in km).

2.2 The spatio-temporal model

Let $y(\boldsymbol{s}_i, t)$ denote the realization of the spatio-temporal process $Y(\cdot, \cdot)$ that represents the PM_{10} concentration measured at station $i = 1, \ldots, d$ located at site $\boldsymbol{s}_i$ and day $t = 1, \ldots, T$. We assume the following measurement equation

$$y(\boldsymbol{s}_i, t) = \boldsymbol{z}(\boldsymbol{s}_i, t)\boldsymbol{\beta} + \xi(\boldsymbol{s}_i, t) + \varepsilon(\boldsymbol{s}_i, t) \tag{1}$$

where $\boldsymbol{z}(\boldsymbol{s}_i,t) = (z_1(\boldsymbol{s}_i,t),\ldots,z_p(\boldsymbol{s}_i,t))$ denotes the vector of p covariates for site $\boldsymbol{s}_i$ at time t, and $\boldsymbol{\beta} = (\beta_1,\ldots,\beta_p)'$ is the coefficient vector. Moreover, $\varepsilon(\boldsymbol{s}_i,t) \sim N\left(0,\sigma_\varepsilon^2\right)$ is the measurement error defined by a Gaussian white-noise process, both serially and spatially uncorrelated. Note that in the geostatistics literature, the term $\boldsymbol{z}(\boldsymbol{s}_i,t)\boldsymbol{\beta}$ is the so-called large scale component - depending in our case study on meteorological and geographical covariates - while the measurement error variance σ_ε^2 is referred to as nugget effect (see Cressie 1993). Finally, $\xi(\boldsymbol{s}_i,t)$ is the realization of the so-called state process, i.e. the true unobserved level of pollution. It is assumed to be a spatio-temporal Gaussian field that changes in time with first order autoregressive dynamics with coefficient a and coloured innovations, given by

$$\xi(\boldsymbol{s}_i,t) = a\xi(\boldsymbol{s}_i,t-1) + \omega(\boldsymbol{s}_i,t) \tag{2}$$

for $t = 2,\ldots,T$, where $|a| < 1$ and $\xi(\boldsymbol{s}_i,1)$ derives from the stationary distribution $N\left(0,\sigma_\omega^2/(1-a^2)\right)$. Moreover, $\omega(\boldsymbol{s}_i,t)$ has a zero-mean Gaussian distribution, is assumed to be temporally independent and is characterized by the spatio-temporal covariance function

$$\mathrm{Cov}\left(\omega\left(\boldsymbol{s}_i,t\right),\omega\left(\boldsymbol{s}_j,t'\right)\right) = \begin{cases} 0 & \text{if} \quad t \neq t' \\ \sigma_\omega^2 \mathcal{C}(h) & \text{if} \quad t = t' \end{cases} \tag{3}$$

for $i \neq j$. The purely spatial correlation function $\mathcal{C}(h)$ depends on the location $\boldsymbol{s}_i$ and $\boldsymbol{s}_j$ only through the Euclidean spatial distance $h = ||\boldsymbol{s}_i - \boldsymbol{s}_j|| \in \mathbb{R}$; thus, the process is assumed to be second-order stationary and isotropic (see Cressie 1993). It follows immediately that $\mathrm{Var}(\omega(s_i,t)) = \sigma_\omega^2$, for each $\boldsymbol{s}_i$ and t. The spatial correlation function $\mathcal{C}(h)$ is defined by the Matérn function and is given by

$$\mathcal{C}(h) = \frac{1}{\Gamma(\nu)2^{\nu-1}}\left(\kappa h\right)^\nu K_\nu\left(\kappa h\right) \tag{4}$$

with K_ν denoting the modified Bessel function of second kind and order $\nu > 0$. The parameter ν, which is usually kept fixed, measures the degree of smoothness of the process and its integer value determines the mean square differentiability of the process. Instead, $\kappa > 0$ is a scaling parameter related to the range ρ, i.e. a distance at which the spatial correlation becomes small. In particular, we use the empirically derived definition $\rho = \frac{\sqrt{8\nu}}{\kappa}$, with ρ corresponding to the distance where the spatial correlation is close to 0.1, for each ν (see Section 2 of Lindgren et al 2011 for more details).

Collecting all the observations measured at time t in a vector denoted by $\boldsymbol{y}_t = (y(\boldsymbol{s}_1,t),\ldots,y(\boldsymbol{s}_d,t))'$, it follows that (1) and (2) can be written as

$$\boldsymbol{y}_t = \boldsymbol{z}_t\boldsymbol{\beta} + \boldsymbol{\xi}_t + \boldsymbol{\varepsilon}_t, \qquad \boldsymbol{\varepsilon}_t \sim N(\boldsymbol{0},\sigma_\varepsilon^2 I_d) \tag{5}$$

$$\boldsymbol{\xi}_t = a\boldsymbol{\xi}_{t-1} + \boldsymbol{\omega}_t, \qquad \boldsymbol{\omega}_t \sim N(\boldsymbol{0},\Sigma = \sigma_\omega^2\tilde{\Sigma}) \tag{6}$$

where I_d is the identity matrix of dimension d, $\boldsymbol{z}_t = (\boldsymbol{z}(\boldsymbol{s}_1,t)',\ldots,\boldsymbol{z}(\boldsymbol{s}_d,t)')'$ and $\boldsymbol{\xi}_t = (\xi(\boldsymbol{s}_1,t),\ldots,\xi(\boldsymbol{s}_d,t))'$ with $\boldsymbol{\xi}_1$ coming from the stationary distribution of the AR(1) process $N\left(\boldsymbol{0},\Sigma/(1-a^2)\right)$. Moreover, $\tilde{\Sigma}$ is the dense correlation matrix of dimension d with elements $\mathcal{C}\left(\|\boldsymbol{s}_i - \boldsymbol{s}_j\|\right)$, where $\mathcal{C}\left(\cdot\right)$ is the Matérn function given by (4) and is parameterized by κ and ν.

Let $\boldsymbol{\theta} = \{\boldsymbol{\beta}, \sigma^2_\varepsilon, a, \sigma^2_\omega, \kappa\}$ denote the parameter vector to be estimated. The joint posterior distribution is given by

$$\pi\left(\boldsymbol{\theta}, \boldsymbol{\xi} \mid \boldsymbol{y}\right) \propto \pi\left(\boldsymbol{y} \mid \boldsymbol{\xi}, \boldsymbol{\theta}\right) \pi(\boldsymbol{\xi} \mid \boldsymbol{\theta}) \pi(\boldsymbol{\theta}) \tag{7}$$

where the notation $\pi(\cdot)$ is used for the probability density function, $\boldsymbol{y} = \{\boldsymbol{y}_t\}$ and $\boldsymbol{\xi} = \{\boldsymbol{\xi}_t\}$ with $t = 1, \ldots, T$. Usually independent prior distributions are chosen for the parameters, so that $\pi(\boldsymbol{\theta}) = \prod_{i=1}^{\dim(\boldsymbol{\theta})} \pi(\boldsymbol{\theta}_i)$. Considering that the observations $\boldsymbol{y}_t$ are serially independent conditionally on $\boldsymbol{\xi}$ and that the state process follows a Markovian time dynamic, Eq.(7) can be written as

$$\pi\left(\boldsymbol{\theta}, \boldsymbol{\xi} \mid \boldsymbol{y}\right) \propto \left(\prod_{t=1}^{T} \pi\left(\boldsymbol{y}_t \mid \boldsymbol{\xi}_t, \boldsymbol{\theta}\right)\right) \left(\pi(\boldsymbol{\xi}_1 \mid \boldsymbol{\theta}) \prod_{t=2}^{T} \pi(\boldsymbol{\xi}_t \mid \boldsymbol{\xi}_{t-1}, \boldsymbol{\theta})\right) \pi(\boldsymbol{\theta}). \tag{8}$$

From the Gaussian distributions defined in (5) and (6), it follows immediately that the joint posterior distribution (8) is given by

$$\begin{aligned}
\pi\left(\boldsymbol{\theta}, \boldsymbol{\xi} \mid \boldsymbol{y}\right) \propto & \left(\sigma^2_\varepsilon\right)^{-\frac{dT}{2}} \exp\left(-\frac{1}{2\sigma^2_\varepsilon} \sum_{t=1}^{T} \left(\boldsymbol{y}_t - \boldsymbol{z}_t\boldsymbol{\beta} - \boldsymbol{\xi}_t\right)' \left(\boldsymbol{y}_t - \boldsymbol{z}_t\boldsymbol{\beta} - \boldsymbol{\xi}_t\right)\right) \\
& \times \left(\frac{\sigma^2_\omega}{1-a^2}\right)^{-\frac{d}{2}} |\tilde{\Sigma}|^{-\frac{1}{2}} \exp\left(-\frac{1-a^2}{2\sigma^2_\omega} \boldsymbol{\xi}_1' \tilde{\Sigma}^{-1} \boldsymbol{\xi}_1\right) \\
& \times \left(\sigma^2_\omega\right)^{-\frac{d(T-1)}{2}} \left|\tilde{\Sigma}\right|^{-\frac{(T-1)}{2}} \exp\left(-\frac{1}{2\sigma^2_\omega} \sum_{t=2}^{T} \left(\boldsymbol{\xi}_t - a\boldsymbol{\xi}_{t-1}\right)' \tilde{\Sigma}^{-1} \left(\boldsymbol{\xi}_t - a\boldsymbol{\xi}_{t-1}\right)\right) \\
& \times \prod_{i=1}^{\dim(\boldsymbol{\theta})} \pi(\boldsymbol{\theta}_i).
\end{aligned}$$

where $\left|\tilde{\Sigma}\right|$ is the determinant of the dense d-dimensional covariance matrix $\tilde{\Sigma}$.

In a Bayesian framework, the common approach to make inference for this model (i.e. parameter estimation and spatial prediction) is MCMC sampling. See, for example, Cameletti et al (2011) and Sahu (2011) for a complete and detailed description of the adopted inferential procedures.

3 Essential details about GMRFs and the SPDE approach

In this section we provide the essential details useful for understanding how a Matérn field - a GF with Matérn covariance function - can be represented as a GMRF. We start introducing the basics of GMRFs and then we move to the SPDE approach. We try to make the discussion as simple as possible and we refer to Lindgren et al (2011) for the theoretical details and proofs of the results.

3.1 GMRFs

A GMRF is a spatial process that models the spatial dependence of data observed on areal units, such as regular grid, lattice structure or geographic regions. For a complete and detailed discussion about GMRFs see Rue and Held (2005). The notation $\boldsymbol{x} = (x_1, \ldots, x_n)'$ with $\boldsymbol{x} \sim N\left(\boldsymbol{\mu}, \mathbf{Q}^{-1}\right)$ refers to a n-dimensional GMRF with mean $\boldsymbol{\mu}$ and symmetric and positive definite precision matrix $\mathbf{Q}$, i.e. the inverse of the covariance matrix. Thus, the density of $\boldsymbol{x}$ is given by

$$\pi(\boldsymbol{x}) = (2\pi)^{-n/2}|\mathbf{Q}|^{1/2} \exp\left(-\frac{1}{2}\left(\boldsymbol{x}-\boldsymbol{\mu}\right)'\mathbf{Q}\left(\boldsymbol{x}-\boldsymbol{\mu}\right)\right).$$

For the purpose of this article, we need to know that a GMRF $\boldsymbol{x}$ can be specified through the conditional distributions for each component given all the others. Moreover, the Markovian property is related to the definition of a neighbourhood structure, in that the full conditional distribution of x_i $(i = 1, \ldots, n)$ depends only on a few of the components of $\boldsymbol{x}$. This set of compontents is denoted by δ_i, which constitutes the set of neighbours of unit i, and

$$\pi\left(x_i \mid \boldsymbol{x}_{-i}\right) = \pi\left(x_i \mid \boldsymbol{x}_{\delta_i}\right),$$

where the notation $\boldsymbol{x}_{-i}$ denotes all elements in $\boldsymbol{x}$ except for x_i. This is equivalent to saying that given the neighbourhood δ_i, the terms x_i and $\boldsymbol{x}_{-\{i,\delta_i\}}$ are independent. Following the notation of Rue and Held (2005), we have that this conditional independence relation can be written as

$$x_i \perp \boldsymbol{x}_{-\{i,\delta_i\}} \mid \boldsymbol{x}_{\delta_i}$$

for $i = 1, \ldots, n$. The key point is that this conditional independence property is strictly related to the precision matrix $\mathbf{Q}$. For a general couple i and j with $j \neq i$, it holds that

$$x_i \perp x_j \mid \boldsymbol{x}_{-\{i,j\}} \Longleftrightarrow \mathbf{Q}_{ij} = 0,$$

which means that the nonzero pattern of $\mathbf{Q}$ is given by the neighbourhood structure of the process. Thus, $\mathbf{Q}_{ij} \neq 0$ if $j \in \{i, \delta_i\}$.

The computational advantage of making inference with a GMRF stems directly from the sparsity of the precision matrix $\mathbf{Q}$. In fact, linear algebra operations can be performed using numerical methods for sparse matrices, resulting in a considerable computational gain (see Rue and Held 2005 for detailed algorithms). For example, matrix factorization, that usually requires $\mathcal{O}(n^3)$ flops for a dense matrix, reduces to $\mathcal{O}(n)$, $\mathcal{O}(n^{3/2})$ and $\mathcal{O}(n^2)$ for the sparse matrix of temporal, spatial and spatio-temporal GMRFs, respectively. Moreover, the computational properties of GMRFs are enhanced by using Integrated Nested Laplace Approximations (INLA, Rue et al 2009) for Bayesian inference. INLA is a computationally effective algorithm that produces fast and accurate approximations to posterior distributions.

An example of a GMRF, that will be used later in Section 4, is the autoregressive process of order 1 given by

$$x_t = ax_{t-1} + \varepsilon_t, \qquad \varepsilon_t \overset{\text{iid}}{\sim} N(0, \sigma^2)$$

with $t = 1, \ldots, n$ where $|a| < 1$ and x_1 derives from the stationary distribution $N(0, \sigma^2/(1-a^2))$. This model belongs to the class of conditional autoregressive model (CAR) introduced by Besag (1974). In this case, x_s and x_t with $1 \leq s < t \leq n$ are conditionally indepedent given $\{x_{s+1}, \ldots, x_{t-1}\}$ if $t - s > 1$. In terms of full conditional distributions, it means that

$$\pi(x_t \mid \boldsymbol{x}_{-t}) = \pi(x_t \mid x_{t-1,} x_{t+1}).$$

From this conditional independence property, it follows that the precision matrix $\mathbf{Q}$ of the autoregressive process has the following tridiagonal structure

$$\mathbf{Q} = \begin{pmatrix} \sigma^2 & -a/\sigma^2 & & & \\ -a/\sigma^2 & (1+a^2)/\sigma^2 & & & \\ & & \ddots & & \\ & & & (1+a^2)/\sigma^2 & -a/\sigma^2 \\ & & & -a/\sigma^2 & \sigma^2 \end{pmatrix}$$

with zero entries outside the diagonal and first off-diagonals. The nonzero values derives from the specification of the full conditional distributions $\pi(x_t \mid \boldsymbol{x}_{-t})$ (see Rue and Held 2005, Chap.1).

3.2 The SPDE approach

Let $X(\boldsymbol{s}) \equiv \{x(\boldsymbol{s}), \boldsymbol{s} \in \mathcal{D} \subseteq \mathbb{R}^2\}$ denote a Matérn field, i.e. a second-order stationary and isotropic GF with a Matérn covariance function, given in (4) and depending on the scale and smoothness parameters κ and ν. Moreover, let suppose to observe a realization of the process $X(\boldsymbol{s}_i)$ at d spatial locations $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_d$.

The objective of the SPDE approach is to find a GMRF, with local neighbourhood and sparse precision matrix $\mathbf{Q}$, that best represents the Matérn field. Given this representation, it is possible to make inference using the GMRF enjoying its good computational properties. This makes it possible to avoid the "big n problem" that arises when working with the dense covariance matrix of a GF.

Basically the SPDE approach uses a finite element representation to define the Matérn field as a linear combination of basis functions defined on a triangulation of the domain $\mathcal{D}$. This consists in subdividing $\mathcal{D}$ into a set of non-intersecting triangles meeting in at most a common edge or corner. Firstly the triangle initial vertices are placed at the locations $\boldsymbol{s}_1, \ldots, \boldsymbol{s}_d$ and then additional vertices are added in order to get a proper triangulation useful

for spatial prediction purposes. To illustrate the concept of triangulation we provide an example referring to the Piemonte case study of Section 2.1. The left panel of Figure 2 displays the locations of the 24 PM_{10} monitoring stations while the right panel shows a triangulation of the region using 123 vertices.

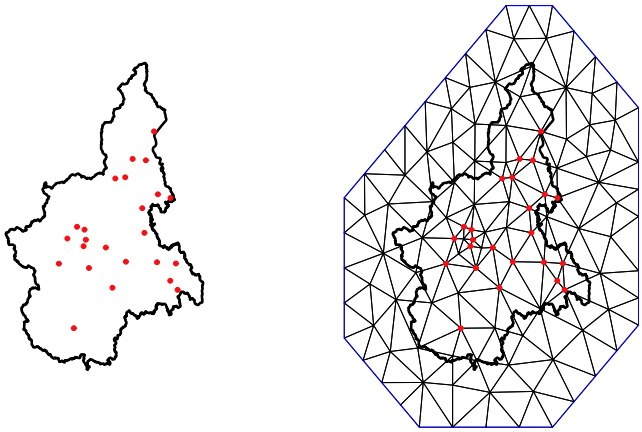

Fig. 2 Left panel: locations of the 24 monitoring stations in Piemonte region. Right panel: triangulation of Piemonte region using 123 vertices.

Given the triangulation, the basis function representation of the Matérn field $X(\boldsymbol{s})$ is given by

$$X(\boldsymbol{s}) = \sum_{l=1}^{n} \psi_l(\boldsymbol{s})\omega_l \tag{9}$$

where n is the total number of vertices, $\{\psi_l(\boldsymbol{s})\}$ are the basis functions and $\{\omega_l\}$ are Gaussian distributed weights. The functions $\{\psi_l(\boldsymbol{s})\}$ are chosen to be piecewise linear on each triangle, i.e. $\psi_l(\boldsymbol{s})$ is 1 at vertex l and 0 at all other vertices. An example is given in Figure 3 that displays a continously indexed spatial random field (left panel) and the corresponding finite element representation with piecewise linear basis functions defined on a given triangulation of the domain (right panel). The height of each triangle (the value of the spatial field at each triangle vertix) is given by the weight w_l and the values in the interior of the triangle are determined by linear interpolation.

The key point of the SPDE approach is the finite element representation (9) that establishes the link between the GF $X(\boldsymbol{s})$ and the GMRF defined by the Gaussian weights $\{\omega_l\}$ to which a Markovian structure can be given, as proved in Lindgren et al (2011). In particular, the precision matrix $\mathbf{Q}$ of the GMRF $\boldsymbol{\omega} = (\omega_1, \ldots, \omega_n)'$ is defined by Eq.(10) of Lindgren et al (2011) as

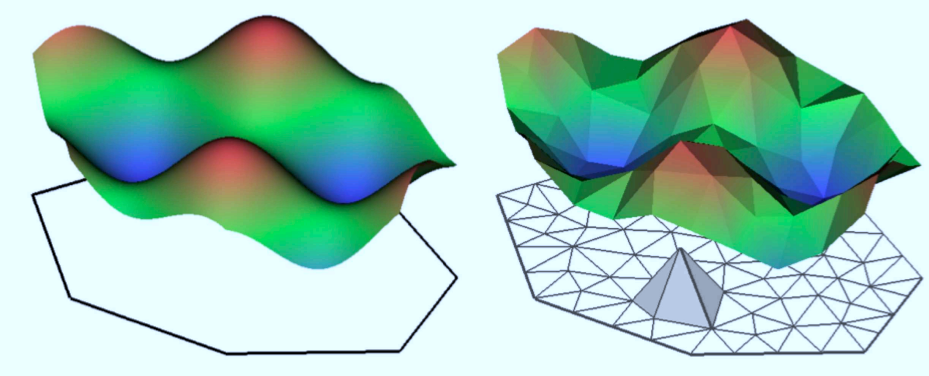

Fig. 3 Left panel: example of a spatial random field (left) given by $X(\boldsymbol{s}) = \cos(s_1)+\sin(s_2)$, where $\boldsymbol{s} = \{s_1, s_2\}$. Right panel: corresponding finite element representation of the spatial random field $X(\boldsymbol{s})$ according to Eq.(9).

a function of κ^2, for $\alpha = 1, 2, \ldots$ and $\nu = 0, 1, 2, \ldots$, where $\alpha = \nu + 1$. This defines an explicit mapping from the parameters of the GF covariance function (κ and ν) to the elements of the precision matrix $\mathbf{Q}$ of the GMRF $\boldsymbol{\omega}$, with a computational cost of $\mathcal{O}(n)$ for any triangulation.

4 How the SPDE approach works

In this section we describe how to implement the spatio-temporal model described in Section 2.2 using the SPDE approach. First we describe how to redefine the model making use of the link between GF and GMRF. Then, we focus on the estimation and spatial prediction procedures.

4.1 Rewriting the model

For each time point $t = 1, \ldots, T$, the Matérn field $\boldsymbol{\omega}_t$ introduced in Eq.(6) is represented through the GMRF $\tilde{\boldsymbol{\omega}}_t \sim N(\mathbf{0}, \mathbf{Q}_S^{-1})$, where the precision matrix $\mathbf{Q}_S$ comes from the SPDE representation discussed in Section 3.2 and is computed using Eq.(10) of Lindgren et al (2011). The matrix $\mathbf{Q}_S$ does not change in time - due to the serial independence hypothesis specified by (3) - and its dimension n is given by the number of vertices of the domain triangulation. Thus, Eq.(6) can be written as

$$\boldsymbol{\xi}_t = a\boldsymbol{\xi}_{t-1} + \tilde{\boldsymbol{\omega}}_t, \qquad \tilde{\boldsymbol{\omega}}_t \sim N(\mathbf{0}, \mathbf{Q}_S^{-1}) \tag{10}$$

for $t = 1, \ldots, T$ and with $\boldsymbol{\xi}_1 \sim N\left(\mathbf{0}, Q_S^{-1}/(1-a^2)\right)$. It follows that the joint distribution of the Tn-dimensional GMRF $\boldsymbol{\xi} = (\boldsymbol{\xi}_1', \ldots, \boldsymbol{\xi}_T')'$ is

$$\boldsymbol{\xi} \sim N\left(\mathbf{0}, \mathbf{Q}^{-1}\right) \tag{11}$$

with $\mathbf{Q} = \mathbf{Q}_T \otimes \mathbf{Q}_S$ where

$$\mathbf{Q}_T = \begin{pmatrix} \sigma_\omega^2 & -a/\sigma_\omega^2 & & & \\ -a/\sigma_\omega^2 & (1+a^2)/\sigma_\omega^2 & & & \\ & & \cdots & & \\ & & & (1+a^2)/\sigma_\omega^2 & -a/\sigma_\omega^2 \\ & & & -a/\sigma_\omega^2 & \sigma_\omega^2 \end{pmatrix}$$

is the T-dimensional precision matrix of the temporal autoregressive process of order 1 specified by (10). Moreover, Eq.(5) can be rewritten as

$$\boldsymbol{y}_t = \boldsymbol{z}_t\boldsymbol{\beta} + \boldsymbol{B}\boldsymbol{\xi}_t + \boldsymbol{\varepsilon}_t, \qquad \boldsymbol{\varepsilon}_t \sim N(\mathbf{0}, \sigma_\varepsilon^2 I_d) \tag{12}$$

where the $(d \times n)$-dimensional matrix $\boldsymbol{B}$ selects the value of the GMRF $\boldsymbol{\xi}_t$ for each observation vector $\boldsymbol{y}_t$. In particular, $\boldsymbol{B}$ is a sparse matrix with only one unit element for each row and such that

$$y(\boldsymbol{s}_i, t) = \boldsymbol{z}(\boldsymbol{s}_i, t)\boldsymbol{\beta} + \sum_{j=1}^{n} \boldsymbol{B}_{ij}\boldsymbol{\xi}_t + \varepsilon(\boldsymbol{s}_i, t)$$

where $\boldsymbol{B}_{ij} = 1$ if the triangle vertix j is placed at location $\boldsymbol{s}_i$ and 0 elsewhere.

4.2 Parameter estimation and spatial prediction

The hierarchical model defined by (12) and (10) belongs to the class of latent Gaussian models and can be estimated using the INLA algorithm proposed in Rue et al (2009). INLA is a computational approach for Bayesian inference and is an alternative to MCMC for getting the approximated posterior marginals for the latent variables as well as for the hyperparameters.

Following Rue et al (2009), let $\boldsymbol{x} = \{\boldsymbol{\xi}, \boldsymbol{\beta}\}$ denote the underlying latent field with a priori independent components. We assign vague Gaussian prior with known precision to $\boldsymbol{\beta}$ and the GMRF distribution (11) to $\boldsymbol{\xi}$. Thus, the density $\pi(\boldsymbol{x} \mid \boldsymbol{\theta})$ is Gaussian with zero mean and precision matrix $\mathbf{Q}(\boldsymbol{\theta}_1)$ with hyperparameter vector $\boldsymbol{\theta}_1 = \left(\sigma_\omega^2, a, \kappa\right)$. Moreover, we have that the observations $\boldsymbol{y} = \{\boldsymbol{y}_t\}$ are normally distributed and conditionally independent given $\boldsymbol{x}$ and $\boldsymbol{\theta}_2 = \sigma_\varepsilon^2$. Thus, denoting by $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$ the hyperparameter vector, the joint posterior distribution is given by

$$\pi\left(\boldsymbol{x}, \boldsymbol{\theta} \mid \boldsymbol{y}\right) = \pi(\boldsymbol{\theta})\pi(\boldsymbol{x} \mid \boldsymbol{\theta}) \prod_{t=1}^{T} \pi(\boldsymbol{y}_t \mid \boldsymbol{x}, \boldsymbol{\theta})$$

where $\pi(\boldsymbol{y}_t \mid \boldsymbol{x}, \boldsymbol{\theta}) \sim N(\boldsymbol{z}_t\boldsymbol{\beta} + \boldsymbol{B}\boldsymbol{\xi}_t, \sigma_\varepsilon^2 I_d)$ is the conditional distribution of the PM_{10} observations at time t defined by (12).

We are interested in the posterior marginal distributions of the latent field and of the hyperparameters, given by:

$$\pi(\boldsymbol{x}_i \mid \boldsymbol{y}) = \int \pi(\boldsymbol{x}_i \mid \boldsymbol{\theta}, \boldsymbol{y})\pi(\boldsymbol{\theta} \mid \boldsymbol{y})d\boldsymbol{\theta} \tag{13}$$

$$\pi(\boldsymbol{\theta}_j \mid \boldsymbol{y}) = \int \pi(\boldsymbol{\theta} \mid \boldsymbol{y})d\boldsymbol{\theta}_{-j} \tag{14}$$

for $i = 1, \ldots, T + p$ and $j = 1, \ldots, 4$. The INLA algorithm - which is designed for non-Gaussian responses - substitutes MCMC simulations with accurate deterministic approximations to these distributions, denoted by $\tilde{\pi}(\boldsymbol{x}_i \mid \boldsymbol{y})$ and $\tilde{\pi}(\boldsymbol{\theta}_j \mid \boldsymbol{y})$ (for the details refer to Rue et al 2009). It is worth to note that for the particular model we are dealing with, characterized by Gaussian observations, we have that $\tilde{\pi}(\boldsymbol{x}_i \mid \boldsymbol{y})$ is exact and Gaussian and the only approximation is the numerical integration required for computing $\tilde{\pi}(\boldsymbol{\theta}_j \mid \boldsymbol{y})$.

With regard to spatial prediction, it is worth to note that the INLA algorithm provides the posterior conditional distribution of $\boldsymbol{\xi}$ for all the n triangulation vertices. Once $\boldsymbol{\xi}$ is given, it is then immediate to get a prediction for $\boldsymbol{y}_t$ for the triangulated domain to be mapped. This is a considerable advantage in terms of computing time with respect to MCMC methods that require first to get the full conditional distribution of the parameters, and then to simulate from the posterior predictive distribution of $y(\boldsymbol{s}_0, t)$ for each $\boldsymbol{s}_0 \in \mathcal{D}$ (see for example Cameletti et al 2011; Sahu 2011).

5 Implementing the SPDE approach through the R-library INLA for the Piemonte case study

In this section we describe how to use the INLA-library of R-software (R Development Core Team, 2011) in order to estimate the parameters of the model described in the previous section and to map PM_{10} concentration for a given day all over Piemonte region. As already discussed in Section 2.1, we consider the October 2005-March 2006 winter season with $T = 182$ days, $d = 24$ monitoring stations and 10 validation sites. The Piemonte dataset and the full R-code are available on the INLA website www.r-inla.org.

5.1 Data import and domain triangulation

First of all we need to load the required libraries together with the Piemonte data (PM_{10} and covariate data, site coordinates - also for the validation stations - and region borders):

```
library(INLA)
library(fields) #for color palette

##--- for the 24 stations and 182 days
Piemonte_data = read.table("Piemonte_data_byday.csv",header=TRUE,sep=",")
coordinates = read.table("coordinates.csv",header=TRUE,sep=",")
```

```
##--- for the 10 validation stations and 182 days
Piemonte_data_validation = read.table("Piemonte_data_byday_validation.csv",
                                      header=TRUE,sep=",")
coordinates_validation = read.table("coordinates_validation.csv",
                                      header=TRUE,sep=",")

borders = read.table("Piemonte_borders.csv",header=TRUE,sep=",")

n_stations = length(coordinates$Station) ## 24
n_stations_val = length(coordinates_validation$Station.ID) ## 10
n_data = length(Piemonte_data$Station.ID) ## 4368
n_days = as.integer(n_data/n_stations) ## 182
```

Note that the `Piemonte_data` and `Piemonte_data_validation` dataframes are composed of 4368 and 1820 rows, respectively, and one row for each day, with 12 columns (Station.ID, Date, *A*, *UTMX*, *UTMY*, *WS*, *TEMP*, *HMIX*, *PREC*, *EMI*, *PM10*; see section 2.1 for covariate acronyms). The required format for the data is shown in Table 5.1. The `coordinates`, `coordinates_validation` and `borders` dataframes are composed of 2 columns containing respectively the *UTMX* and *UTMY* coordinates of the monitoring stations and of the points belonging to the region border.

Station.ID	Date	A	UTMX	UTMY	WS	TEMP	HMIX	PREC	EMI	PM10
1	01/10/05	95.20	469.45	4972.85	0.90	288.81	1294.60	0.00	26.05	28
2	01/10/05	164.10	423.48	4950.69	0.82	288.67	1139.80	0.00	18.74	22
...	...	...	...	...	...	...	...	...	...	...
23	01/10/05	131.00	454.26	5019.81	0.80	288.18	860.93	0.00	26.15	32
24	01/10/05	201.40	466.40	5086.60	1.27	289.28	617.04	1.80	16.78	11
1	02/10/05	95.20	469.45	4972.85	1.53	286.48	566.56	8.92	22.21	14
2	02/10/05	164.10	423.48	4950.69	1.41	285.79	513.36	16.72	15.37	10
...	...	...	...	...	...	...	...	...	...	...
23	02/10/05	131.00	454.26	5019.81	2.65	285.52	454.80	23.35	21.82	26
24	02/10/05	201.40	466.40	5086.60	2.97	285.53	692.51	41.14	13.96	5
...	...	...	...	...	...	...	...	...	...	...

Table 1 Required format for the data: days are stacked row by row and variables are on the columns.

As the ranges of the $p = 8$ covariates are quite different, we apply a standardization procedure subtracting the mean and dividing by the standard error computed consider the 24 monitoring stations.

```
mean_covariates = apply(Piemonte_data[,3:10],2,mean)
sd_covariates = apply(Piemonte_data[,3:10],2,sd)
Piemonte_data[,3:10] = scale(Piemonte_data[,3:10],
                      mean_covariates, sd_covariates)
```

We apply the same standardization also for the covariates of the validation stations:

```
Piemonte_data_validation[,3:10] = scale(Piemonte_data_validation[,3:10],
                        mean_covariates, sd_covariates)
```

Moreover, in order to stabilize the variances, which increase with the mean values, and to make the distribution of PM_{10} data approximately normal, we use a logarithmic transformation and add a new variable named `logPM10` to each of the `Piemonte_data` and the `Piemonte_data_validation` dataframes:

```
Piemonte_data$logPM10 = log(Piemonte_data$PM10)
Piemonte_data_validation$logPM10 = log(Piemonte_data_validation$PM10)
```

In order to simplify keeping track of the temporal aspect of the data, we also add `time` as the index of each observation day

```
Piemonte_data$time = rep(1:n_days, each = n_stations)
Piemonte_data_validation$time = rep(1:n_days, each = n_stations_val)
```

Next, we consider how to triangulate the Piemonte region. We seek a triangulation based on initial vertices at the $d = 24$ station locations, with further vertices added in order to satisfy triangulation quality constraints. To ensure that the triangles cover our target spatial domain for spatial prediction, the Piemonte region, we could manually construct the precise boundary using `inla.mesh.segment`, but that would result in a much too detailed boundary. A better alternative is to use the builtin option to construct convex sets covering the true boundary:

```
mesh = inla.mesh.create.helper(points=cbind(coordinates$UTMX,
                                            coordinates$UTMY),
                               points.domain=borders,
                               offset=c(10, 140),
                               max.edge=c(50, 1000),
                               min.angle=c(26, 21))
```

The helper function starts by creating a triangulation of the domain defined by the region border, using a convex set with 8 edges at a distance of 10 km. Next, to reduce the visible boundary effects of the SPDE, a further extension with 16 edges is added at a distance of a further 140 km, giving a total extension of 150 km. The maximal edge length is specified to 50 km in the interior of the region, and to 1000 km in the outer extension. Since the outer extension is of no practical interest, the resolution needs only to be as fine as required by the numerics. In this case, the boundary effect can be handled well by only specifying the minimal allowed interior angle for the triangles, with 21 degrees being the largest number guaranteeing termination of the triangulation algorithm. Values as large as 34 are possible for this particular setting, giving 862 vertices, but this is excessive compared to the modest 122 vertices obtained by using minimum angle 21 degrees. Using the compromise of 26 degrees for the inner region and 21 degrees for the outer extension, the number of vertices becomes 142 (stored in `mesh$n`). We found no practical differences in the results using finer scale triangulations, only higher computational cost.

Figure 4, obtained using the following code, shows the obtained triangulation covering the Piemonte region and stretching out towards the extended boundary.

```
plot(mesh)
lines(borders,lwd=3)
```

```
points(coordinates$UTMX,coordinates$UTMY,
       pch=20,cex=2, col=2)
points(coordinates_validation$UTMX,coordinates_validation$UTMY,
       pch=20,cex=2,col=4)
```

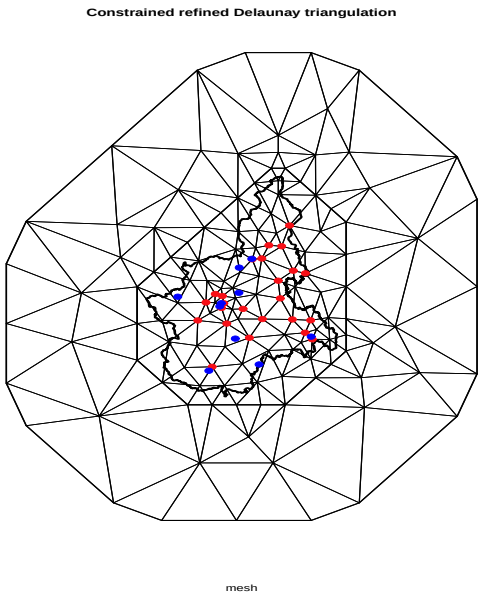

Fig. 4 The Piemonte region triangulation with 142 vertices. The red dots mark the 24 monitoring stations, and the blue dots mark the 10 validation stations.

5.2 Definition of the SPDE model object and call of the `inla(.)` function

We now create a SPDE model object for a Matérn-like spatial covariance function using the function `inla.spde2.matern(.)` specifying the obtained triangulation (given by the `mesh` object) and the parameter $\alpha = 2$ and, as noted at the end of Section 3.2, it follows that the smoothness parameter ν of the Matérn covariance function is equal to 1. The following code defines the `spde` model object:

```
spde = inla.spde2.matern(mesh=mesh, alpha=2)
```

In order to avoid having to keep track of vertex indexing, we make use of a `R-INLA` feature that allows the observation equation to be written on matrix form, $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{\eta} + \boldsymbol{\epsilon}$, where $\boldsymbol{y}$ are the observations, $\boldsymbol{\eta}$ is a linear predictor, $\boldsymbol{\epsilon}$ is the observation noise, and $\boldsymbol{A}$ is an *observation matrix*. The function `inla.stack(.)` is used to build the necessary data structures, combining simple model building blocks into large complicated models.

Using a helper function, we construct an observation matrix that extracts the values of the spatio-temporal field at the measurement locations and time points used for the parameter estimation:

```
A.est = inla.spde.make.A(mesh,
                         loc=as.matrix(coordinates[Piemonte_data$Station.ID,
                                                   c("UTMX","UTMY")]),
                         group=Piemonte_data$time,
                         n.group=n_days)
```

with corresponding code for `A.val` for the validation data, and

```
A.pred = inla.spde.make.A(mesh, group=i_day, n.group=n_days)
```

for the observation matrix for prediction of day number `i_day`. The full model uses a combination of a latent spatio-temporal model and covariate effects. The observation models allows us to simplify the specification of the latent model through the helper function `inla.spde.make.index`, which generates vectors of indices for the spatial and temporal components of the model. The call

```
field.indices = inla.spde.make.index("field", n.mesh=mesh$n, n.group=n_days)
```

generates a list with fields `field` and `field.group`, where the former contains spatial vertex indices and the latter contains temporal indices. The full data structure needed for the model is then constructed by the following code:

```
stack.est =
    inla.stack(data=list(logPM10=Piemonte_data$logPM10),
               A=list(A.est, 1),
               effects=
                   list(c(field.indices,
                          list(Intercept=1)),
                        list(Piemonte_data[,3:10])),
                   tag="est")
stack.val = [Omitted for brevity]
scaled.mesh.loc =
    list(UTMX=(rep(scale(mesh$loc[,1],
                         mean_covariates["UTMX"],
                         sd_covariates["UTMX"]), n_days)),
         UTMY=(rep(scale(mesh$loc[,2],
                         mean_covariates["UTMY"],
                         sd_covariates["UTMY"]), n_days)))
stack.pred =
        inla.stack(data=list(logPM10=NA),
                   A=list(A.pred),
                   effects=list(c(field.indices,
                                  scaled.mesh.loc,
                                  list(Intercept=1))),
                   tag="pred")
stack = inla.stack(stack.est, stack.val, stack.pred)
```

In each `inla.stack` call, `effects` is a list of linear predictor component groups, such that each group has its own observation matrix, specified as the list of matrices `A`, with `1` interpreted as an identity matrix. The result is a linear predictor model with the sum of the observed component groups as its final value, as well as predictors for the validation stations and a joint

predictor for all the spatial components on the given prediction day. However, this is only in the mind of the user, and we still need to specify the linear predictor model as an `R-INLA` formula object. This includes the $p = 8$ covariates (fixed effects) together with random effect components, specified by the `f(.)` function, which is designed to define non-fixed effects such as spatial random effects, time trends, seasonal effects, etc.. With the following specification of `f(.)`, through the `group` and `control.group` options, we specify that, at each time point, the spatial locations are linked by the `spde` model object, while across time, the process evolves according to an AR(1) process.

```
formula <- (logPM10 ~ -1 + Intercept +
                      A + UTMX + UTMY + WS + TEMP + HMIX + PREC + EMI +
                      f(field, model = spde, group = field.group,
                        control.group = list(model="ar1")))
```

Due to the way `inla.stack` is implemented, an automatic intercept effect cannot be used, and we instead specify an explicit `Intercept` covariate. Finally, the specified model with Gaussian response can be run calling the `inla(.)` function as follows:

```
result = inla(formula,
              data = inla.stack.data(stack, spde=spde),
              family = "gaussian",
              control.predictor = list(A=inla.stack.A(stack), compute=TRUE))
```

The posterior summary statistics (mean, quantiles and standard deviations) of the fixed effects, i.e. the $\boldsymbol{\beta}$ covariate coefficients, are obtained from `result$summary.fixed`, and are shown in Table 2. In particular, the posterior mean of the intercept is 3.69 on the log scale, which corresponds to an average pollution level of about 40 $\mu g/m^3$, after adjustment for covariates. As expected, a significant and positive relationship is observed between emissions (EMI) and PM_{10} concentration. Moreover, the significance of the coefficients of WS, $TEMP$ and $PREC$ confirms the importance of meteorological variables on air quality. Finally, altitude (A) has a significant effect in reducing PM_{10} concentration.

The summary statistics of the posterior distribution of the AR(1) coefficient a are obtained from the `result$summary.hyperpar` matrix with the following command

```
result$summary.hyperpar["GroupRho for field",]
```

and are reported in Table 3. We note that the `inla(.)` function provides us with the mean, quantiles and standard deviation of the Gaussian observation precision parameter $1/\sigma^2_\varepsilon$. As we are interested in the variance σ^2_ε, we need to transform the marginal density of the precision using the `inla.tmarginal(.)` function, as follows

```
sigma2eps_marg = inla.tmarginal(function(x) 1/x,
        result$marginals.hyperpar$"Precision for the Gaussian observations")
```

Then using the `inla.emarginal(.)` and `inla.eqmarginal(.)` functions we can easily compute the mean, the standard deviation and the quantiles of σ^2_ε, given in Table 3:

			Quantiles		
Covariate	Mean	St.Dev.	0.025	0.5	0.975
Intercept	3.69	0.45	2.79	3.69	4.57
A	-0.20	0.05	-0.29	-0.20	-0.10
UTMX	-0.16	0.16	-0.48	-0.16	0.16
UTMY	-0.18	0.15	-0.48	-0.18	0.11
WS	-0.06	0.01	-0.08	-0.06	-0.04
TEMP	-0.12	0.04	-0.19	-0.12	-0.05
HMIX	-0.02	0.01	-0.05	-0.02	0.002
PREC	-0.05	0.01	-0.07	-0.05	-0.04
EMI	0.04	0.01	0.01	0.04	0.07

Table 2 Posterior estimates (mean, standard deviation and quantiles) of the covariate coefficient vector $\boldsymbol{\beta}$.

```
sigma2eps_m1 = inla.emarginal(function(x) x, sigma2eps_marg)
sigma2eps_m2 = inla.emarginal(function(x) x^2, sigma2eps_marg)
sigma2eps_stdev = sqrt(sigma2eps_m2 - sigma2eps_m1^2)
sigma2eps_quantiles = inla.qmarginal(c(0.025, 0.5, 0.975), sigma2eps_marg)
```

The properties of the parameter estimates for the spatial SPDE model can be obtained by running the code

```
result.field = inla.spde.result(result, "field", spde, do.transform=TRUE)
```

which extracts all the relevant bits of information from `result`, and also transforms the results from internal parameter scales, giving posterior distributions for nominal variance and nominal range in addition to the internal $\theta_1 = \log(\tau)$ and $\theta_2 = \log(\kappa)$. From

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

we get a value of 275 km for the empirically derived correlation range $\rho = \frac{\sqrt{8\nu}}{\kappa}$. As this is the distance at which the correlation is close to 0.1, we can conclude that the data are characterized by a strong spatial correlation which decreases slowly with distance. As reported in Section 2 of Lindgren et al (2011), the variance σ_ω^2 is given by

$$\sigma_\omega^2 = \frac{1}{4\pi\kappa^2\tau^2}$$

where τ is the scaling parameter with estimate stored in the element `Theta1 for field` of `result$summary.hyperpar`. The posterior mean of σ_ω^2 as obtained from

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

was 1.2762. All the posterior estimates (mean, quantiles and standard deviation) for the hyperparameters σ_ε^2, σ_ω^2, ρ and a are collected in Table 3. We observe that more variation is explained by the spatial term rather than by the measurement error. Moreover, the high value of the AR(1) temporal correlation coefficient confirms the short-term persistence of particulate matter.

			Quantiles		
Parameter	Mean	St.Dev	0.025	0.5	0.975
σ^2_ε	0.0326	0.0014	0.0300	0.0325	0.0353
σ^2_ω	1.2762	0.240	0.9300	1.2302	1.8603
ρ	275	16.8	244	275	310
a	0.9601	0.0081	0.9453	0.9598	0.9759

Table 3 Posterior estimates (mean, standard deviation and quantiles) of σ^2_ε, σ^2_ω, ρ and a.

5.3 Spatial prediction

For spatial prediction purposes we consider a 4×4 km grid of 56×72 grid points ranging from 309 km to 529 km in the Eastern direction and from 4875 km to 5159 km in the Northern direction. Our objective is to get a map, for a given day, of the PM_{10} concentration (on the logarithmic scale) together with a probability of exceedance map using the threshold of 50 $\mu g/m^3$ (this is the value fixed by the European directive 2008/50/EC for the daily mean concentration and cannot be exceeded more than 35 days in a year) . If we want to take into account the prediction of the smooth PM_{10} concentration field without the nugget term, we simply add the large scale component $\boldsymbol{z}(\boldsymbol{s}_0, t)\boldsymbol{\beta}$ to the value of the latent field $\xi(\boldsymbol{s}_0, t)$, with $\boldsymbol{s}_0 \in \mathcal{D}$ and $1 \leq t \leq 182$. Since the observation stations cover only a limited altitude range, we present the results only for elevations below 1 km, to avoid inappropriate linear extrapolation of the effect of elevation.

To perform spatial prediction we first load the covariate array

```
load("covariate_array_std.Rdata")
dim(covariate_array_std)
```

whose dimension in our data setting is $56 \times 72 \times 8$, where 8 is the number p of covariates (stored in the order *A, UTMX, UTMY, WS, TEMP, HMIX, PREC, EMI*). Then, using the `inla.mesh.projector(.)` function, we define a lattice projection starting from the `inla.mesh` object created before (named `mesh`) and the definition of the Piemonte grid:

```
proj_grid =
    inla.mesh.projector(mesh,
                        xlim=range(Piemonte_grid[,1]),
                        ylim=range(Piemonte_grid[,2]),
                        dims=c(56,72))
```

Successively, we extract the posterior mean and the standard deviation of the latent field

```
field_pred_mean =
    result$summary.linear.predictor[inla.stack.index(stack,"pred")$data, "mean"]
field_pred_sd =
    result$summary.linear.predictor[inla.stack.index(stack,"pred")$data, "sd"]
```

obtaining the results for the specific day specified for prediction in the original `inla` call, here $i = 122$, corresponding to January 30, 2006, and again with the

`inla.mesh.project(.)` function we project the latent field from the mesh to the grid `proj_grid`:

```
grid_latent_mean = inla.mesh.project(proj_grid, field_pred_mean)
grid_latent_sd = inla.mesh.project(proj_grid, field_pred_sd)
```

Finally, we compute the smooth PM_{10} concentration predictions for all the $56 \times 72 = 4032$ grid points adding to the $\boldsymbol{z}(\boldsymbol{s}_0, t)\boldsymbol{\beta}$ term the gridded latent field given by `grid_latent_mean`. The predictions are stored in the `grid_mean` matrix:

```
beta = result$summary.fixed[,"mean"]
grid_mean = grid_latent_mean
grid_var = grid_latent_sd^2
for (b in c(2,5:9)) {
    grid_mean = grid_mean + covariate_array_std[,,b-1]*beta[b]
    grid_var = grid_var + covariate_array_std[,,b-1]^2*beta_sd[b]^2
}
grid_sd = grid_var^0.5
```

Note that two of the covariates are skipped in these approximate calculations, since they were properly included in the Bayesian integration as performed by `inla`. The following code produces the prediction and exceedance probalility maps reported in Figure 5.3 using the `levelplot(.)` function of the `lattice` library together with the color palette `tim.colors(.)` contained in the `fields` library:

```
levelplot(row.values=proj_grid$x, column.values=proj_grid$y,
        x=grid_mean,
        col.regions=tim.colors(64),
        ylim=c(4875,5159),xlim=c(309,529),
        aspect="iso",
        contour=TRUE, cuts=11, labels=FALSE, pretty=TRUE,
        xlab="Easting",ylab="Northing")
trellis.focus("panel", 1, 1, highlight=FALSE)
lpoints(borders,col=1,cex=.25)
lpoints(coordinates$UTMX, coordinates$UTMY,col=1,lwd=2,pch=21)
trellis.unfocus()

u_level = log(50) #threshold
grid_prob_plugin = pnorm((grid_mean-u_level)/grid_sd)
levelplot(row.values=proj_grid$x, column.values=proj_grid$y,
        x=grid_prob_plugin,
        col.regions=tim.colors(64),
        ylim=c(4875,5159),xlim=c(309,529),
        aspect="iso",
        at=(0:10)/10,
        contour=TRUE, cuts=11, labels=FALSE, pretty=TRUE,
        xlab="Easting",ylab="Northing")
trellis.focus("panel", 1, 1, highlight=FALSE)
lpoints(borders,col=1,cex=.25)
lpoints(coordinates$UTMX, coordinates$UTMY,col=1,lwd=2,pch=21)
trellis.unfocus()
```

For higher accuracy for the exceedance probabilities one could use the full posterior distributions as provided by `inla(.)`, but in our experience the Gaussian plugin estimator used here is sufficient and much faster.

As expected, higher levels of PM_{10} pollution and exceedance probabilities are detected in the metropolitan areas located near the main cities of the region (Torino, Vercelli and Novara) and moving eastwards toward Milan.

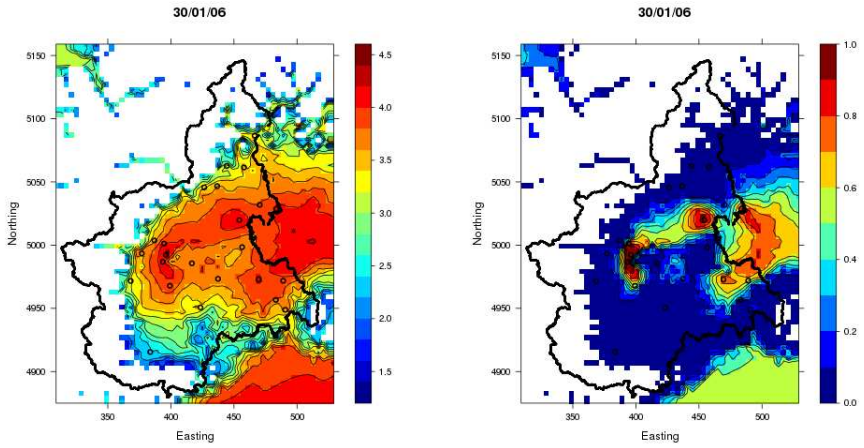

Fig. 5 Map of the PM_{10} posterior mean on the logarithmic scale (left) and exceedance probability for 50 $\mu g/m^3$ (right) for January 30th, 2006. Only locations with an altitude below 1 km are shown.

5.4 Validation

In order to assess the validity of the estimated model, we perform a simple residual analysis. Using `inla.stack.index` to identify the data indices corresponding to the 10 validation sites, we first calculate the residuals (`res`) and standardised residuals (`res.std`). The standardisation takes into account the variance of the Gaussian data likelihood:

```
validation = list()
index = inla.stack.index(stack,"val")$data
tmp.mean = result$summary.linear.predictor[index,"mean"]
tmp.sd = result$summary.linear.predictor[index,"sd"]
validation$res = Piemonte_data_validation$logPM10 - tmp.mean
validation$res.std = validation$res /
    sqrt(tmp.sd^2 + 1/result$summary.hyperpar[1,"mean"])
```

From this, we calculate the actual coverage probability of a prediction interval with nominal coverage probability 95%:

```
validation$p = pnorm(validation$res.std)
validation$cover = mean((validation$p>0.025) & (validation$p<0.975), na.rm=TRUE)
```

This yelds an actual coverage probability of 89.7%, indicating that we've underestimated the uncertainty of predictions. This is likely due to the model

overfitting the data, as indicated by the corresponding value 98.2% for the data from the 24 estimation sites.

Further residual metrics can also be calculated, such as root mean squared error (RMSE) and correlation coefficient between observations and predictions:

```
validation$rmse = sqrt(mean(validation$res^2, na.rm=TRUE))
validation$cor = cor(Piemonte_data_validation$logPM10, tmp.mean,
                     use="pairwise.complete.obs", method="pearson")
```

The resulting RMSE is 0.5328, and the correlation coefficient 0.7015. For comparison, *Model C* in Cameletti et al (2011) - which is applied to the same data with MCMC methods - achieved a RMSE of 0.3476 and a correlation of 0.8637. The only difference between the models is that the model from Cameletti et al (2011) uses smoothness parameter $\nu = 1/2$ in the covariance, corresponding to an exponential spatial covariance function. Moreover, if we compare the parameter estimates for both the models, similar results are obtained for the covariate coefficients, whereas the estimated range ρ and the AR(1) coefficient a are quite different: 275 km and 0.96 for the SPDE approach and 1046 km and 0.654 for Cameletti et al (2011), respectively. Considering the characteristics of PM_{10} pollution in Piemonte region and in the Po Valley, it seems that the estimates of ρ and a obtained with the SPDE approach are more reasonable. Some differences can be detected also for the variances: while in Cameletti et al (2011) the estimates for σ^2_ω and σ^2_ε are equal to 0.950 and 0.013, with the SPDE approach we get 1.276 and 0.033, respectively. In any case, to properly compare the SPDE/GMRF/INLA approach to the classical MCMC based computations, one could use the spectral approximation method from the authors' response to the discussion of Lindgren et al (2011) to construct a GMRF approximation to an exponential covariance, corresponding to $\alpha = 3/2$ in the SPDE model.

6 Discussion

In this work we describe how to employ the SPDE approach for a spatio-temporal hierarchical model that involves a GF and a state process characterized by first order autoregressive dynamics and spatially correlated innovations. In particular, we show, through a motivating problem regarding PM_{10} data in Piemonte, how to use the `R`-library `INLA` to get the parameter posterior estimates together with prediction and uncertainty maps. The results we obtain are comparable to the ones reported for *Model C* in Cameletti et al (2011), where the same data are modeled through an almost identical spatio-temporal model, with the only difference that $\nu = 1/2$ which corresponds to an exponential spatial covariance function. In Cameletti et al (2011) all the inferential procedures are carried out using MCMC methods, requiring on average 0.4 seconds for each iteration using an Intel Xeon 8 CPU cluster (2.66 Ghz, 8 GB RAM) and `Matlab R2009b` with the Parallel Computing Toolbox. In the hyphotetical case of 50000 MCMC iterations, this means that almost 6

hours are needed for completing the estimation step and making spatial predictions over the set of 10 validation stations considered there. Instead, using the SPDE approach and the `R`-library `INLA` on an Intel Xeon 12 CPU machine (3.33GHz, 96 GB RAM), the `inla` program used only 240 seconds to calculate the posterior distributions of the hyperparameters and of the latent field over the triangulated domain. Unlike the different machine settings and employed software, the computational strength of the SPDE approach implemented by the INLA algorithm stands out clearly. Besides, it is worth to note that when working with the INLA algorithm problems of convergence and mixing - typical of the sampling-based MCMC methods - are not an issue at all. Furthermore, even if in this work we focus on a particular hierarchical model, the SPDE approach can be immediately extended to a wide class of spatio-temporal models. For example, it is possible to consider models with more complex hierarchical structures or with non-separable covariance function as well as non-stationary cases characterized by parameters that change in time. For all these reasons, we believe that the SPDE approach, combined with the INLA algorithm, is an outstanding computational framework for performing Bayesian inference on complex spatio-temporal GFs, also when dealing with massive datasets.

Another issue we focus on in this work is the `R`-library `INLA`. In particular, we describe step-by-step the `R`-code required for modeling PM_{10} data in Piemonte making use of the SPDE approach. As it is quite easy to get prediction and uncertainty maps in a reasonable computing time, we think that the user-friendly `INLA`-library is particularly suitable for environmental agencies that seek effective tools for modeling and mapping high-dimensional air quality data.

Acknowledgements Cameletti's research was funded in part by Lombardy Region under "Frame Agreement 2009" (Project EN17, "Methods for the integration of different renewable energy sources and impact monitoring with satellite data".

References

Banerjee S, Carlin B, Gelfand A (2004) Hierarchical Modeling and Analysis for Spatial Data. Monographs on Statistics and Applied Probability, Chapman and Hall, New York

Banerjee S, Gelfand A, Finley A, Sang H (2008) Gaussian predictive process models for large spatial datasets. J R Statist Soc B 70(4):825–848

Besag J (1974) Spatial interaction and the statistical analysis of lattice systems (with discussion). J R Statist Soc B 36(2):192–225

Cameletti M, Ignaccolo R, Bande S (2011) Comparing spatio-temporal models for particulate matter in piemonte. Environmetrics DOI: 10.1002/env.1139

Cocchi D, Greco F, Trivisano C (2007) Hierarchical space-time modelling of PM_{10} pollution. Atmospheric environment 41:532–542

Cressie N (1993) Statistics for Spatial Data. Wiley, New York

Cressie N, Johannesson G (2008) Fixed rank kriging for large spatial datasets. J R Statist Soc B 70:209–226

Cressie N, Wikle C (2011) Statistics For Spatio- Temporal Data. Wiley

Finardi S, De Maria R, D'Allura A, Cascone C, Calori G, Lollobrigida F (2008) A deterministic air quality forecasting system for Torino urban area, Italy. Environmental Modelling and Software 23(3):344–355

Furrer R, Genton M, Nychka D (2006) Covariance tapering for interpolation of large spatial datasets. Journal of Computational and Graphical Statistics 15(3):502–523

Gelfand A, Diggle P, Fuentes M, Guttorp P (eds) (2010) Handbook of Spatial Statistics. Chapman & Hall

Lindgren F, Rue H, Lindström J (2011) An explicit link between Gaussian fields and Gaussian Markov random fields: the stochastic partial differential equation approach (with discussion). J R Statist Soc B 73(4):423–498

R Development Core Team (2011) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, URL `http://www.R-project.org`, ISBN 3-900051-07-0

Rue H, Held L (2005) Gaussian Markov Random Fields. Theory and Applications. Chapman & Hall

Rue H, Martino S, Chopin N (2009) Approximate Bayesian inference for latent Gaussian model by using integrated nested Laplace approximations (with discussion). J R Statist Soc B 71:319–392

Sahu S (2011) Hierarchical Bayesian models for space-time air pollution data. In: Rao C (ed) Handbook of Statistics - Time Series Analysis, Methods and Applications, Handbook of Statistics, vol 30, Elsevier Publishers, Holland, pp 0000 – 0000

Samet J, Dominici F, Curriero F, Coursac I, Zeger S (2000) Fine particulate air pollution and mortality in 20 US cities: 1987-1994. New England Journal of Medicine 343:1742–1749

Samoli E, Peng R, Ramsay T, Pipikou M, Touloumi G, Dominici F, Burnett R, Cohen A, Krewski D, Samet J, Katsouyanni K (2008) Acute effects of ambient particulate matter on mortality in Europe and North America: results from the APHENA study. Environmental Health Perspectives 116:1480–1486