Soumission

Journal de la Société Française de Statistique

Analyzing spatio-temporal data with R: Everything you always wanted to know – but were afraid to ask

Titre: Donnees spatio-temporelles avec R : tout ce que vous avez toujours voulu savoir sans jamais avoir osé le demander

RESSTE Network[1]

Abstract: We present an overview of (geo-)statistical models, methods and techniques for the analysis and prediction of continuous spatio-temporal processes residing in continuous space. Various approaches exist for building statistical models for such processes, estimating their parameters and performing predictions. We cover the Gaussian process approach, very common in spatial statistics and geostatistics, and we focus on R-based implementations of numerical procedures. To illustrate and compare the use of some of the most relevant packages, we treat a real-world application with high-dimensional data. The target variable is the daily mean PM_{10} concentration predicted thanks to a chemistry-transport model and observation series collected at monitoring stations across France in 2014. We give R code covering the full work-flow from importing data sets to the prediction of PM_{10} concentrations with a fitted parametric model, including the visualization of data, estimation of the parameters of the spatio-temporal covariance function and model selection. We conclude with some elements of comparison between the packages that are available today and some discussion for future developments.

Résumé : Nous présentons un aperçu des modèles, méthodes et techniques (géo-)statistiques pour l'analyse et la prévision de processus spatio-temporels continus. De nombreuses approches sont possibles pour la construction de modèles statistiques pour ces processus, l'estimation de leurs paramètres et leur prédiction. Nous avons choisi de présenter l'approche par processus gaussien, la plus communément utilisée en statistiques spatiales et en géostatistiques, ainsi que son implémentation avec le logiciel R. Le variable cible est la moyenne de la concentration quotidienne PM_{10} à l'échelle de la France, prédite à l'aide d'un modèle de transport en chimie de l'atmosphère et de séries d'observations obtenues à des stations de surveillance de la qualité de l'air. En suivant le fil d'une application réelle de grande dimension, nous comparons certains des paquets R les plus utilisés. Le code R permettant la visualisation des données, l'estimation des paramètres de la fonction de covariance spatio-temporelle ainsi que la sélection d'un modèle et la prédiction de la concentration de PM_{10} est également présenté afin d'illustrer l'enchaînement des étapes. Nous concluons avec une comparaison entre les paquets qui sont disponibles aujourd'hui et ainsi que les pistes de développement qui nous paraissent intéressantes.

Keywords: Space-time, Covariance function, Geostatistics, Kriging, Air pollution
Mots-clés : Fonction de covariance, Géostatistique, Krigeage, Pollution atmosphérique
AMS 2000 subject classifications: 62-01, 62-07, 62F99, 62M40

[1] The RESSTE Network gathers academics from several organizations. See the Acknowledgments Section for details and visit http://informatique-mia.inra.fr/resste/

Soumis au Journal de la Société Française de Statistique
File: main.tex, compiled with jsfds, version : 2009/12/09
date: September 16, 2016

1. Introduction

Several points of view can be adopted for analyzing and modeling spatio-temporal data. They depend on the organization of the spatio-temporal design, the type of data and the statistical community the statistician in charge of the analysis belongs to.

When data are sparse in space but dense in time, which is typically the case when the data arise from a small set of sensors with regular measurements along time, the analysis can be done within the framework of multivariate time series. This point of view is sometimes adopted by scientists used to analyze time series. When data are dense in space and sparse in time, thus providing snapshots of the spatio-temporal field, one can work in a multivariate geostatistical setting, where repetitions in time is treated as several variables. When data are dense both in time and space which is the case of interest in this paper, these approaches fail due to the impossibility to build very large dimensional multivariate temporal or spatial processes. In addition, for a variety of reasons, data are not necessarily collected according to a factorial design in $\{$ Space $\times$ Time $\}$, implied by the multivariate time series or geostatistical framework mentioned above. In particular, data might be collected along moving devices such as sensors on vehicles or mobile phones.

A much more flexible and theoretically richer approach is to consider the field $Z(s,t)$, with $(s,t) \in D \times T$ where $D \subset \mathbb{R}^d$ and $T \subset \mathbb{R}$ as a single random field. Mathematically, one may consider $Z(s,t)$ as a random function in $\mathbb{R}^{d+1}$ and do geostatistics as usual. This approach is mathematically correct but it misses a key point, which is the special role played by the time dimension. As we all know and experience in our everyday life, time flows only in one direction; more formally, we can refer to the physical laws of thermodynamics that substantiate the asymmetries that can arise along the time arrow. Useful statistical models are expected to to take into account this asymmetry, at least partially.

The statistical environment `R` (R Core Team, 2013) is a priceless tool in both the direct and figurative sense that brought effortless freedom to statisticians to quickly test ideas, to progressively perform elaborate statistical analyses and to nicely visualize their scientific reporting. It is still possible to deal with spatio-temporal data writing our own programs on the fly with basic `R` commands such as `data.frame`, `image` and so on. But when facing the dimensional burden of handling spatio-temporal datasets, there are good reasons to willingly sacrifice some of our undisciplined freedom and curb to strictly typed standardized spatio-temporal data formats: (i) to share our work with collaborators, (ii) to benefit from the work of others. Were it necessary to put forward an authoritative argument, the most famous textbook for spatio-temporal statistics Cressie and Wikle (2015) refers to the `R` package `spacetime` Pebesma (2012) for all numerical case studies and applications. To wit, one can type `demo("CressieWikle",package="spacetime")` in the RStudio console to launch some of the book's impressive demos. Of course, additional skills are necessary to go from the casual use of R as the mathematician's pocket calculator to an efficient statistical tool for spatio-temporal analysis.

A plethora of approaches exist for building statistical models for spatio-temporal processes $Z(s,t)$, estimating their parameters and performing predictions. It would be impossible to cover all of them in this review. We here limit our exposition to the most commonly used approaches of point estimation, leaving in particular the wide field of Bayesian techniques out of scope. Our aim is not to give a most comprehensive list of available techniques and `R` packages, but to roll up a guiding thread exemplified on a critically large real air pollution data set so as to make this

Soumis au Journal de la Société Française de Statistique
File: main.tex, compiled with jsfds, version : 2009/12/09
date: September 16, 2016

learning phase as painless as possible. The reader will be guided through a detailed start-to-end example using standard models and approaches.

The article is organized as follows: Section 2 presents the air pollution data set and explains how to handle spatio-temporal data, relying on the structured objects from the R package `spacetime` (Pebesma, 2012). Based on a real case study with spatio-temporal pollution data, we give vital advice to import such large databases into R and recall the essentials of projection coordinate systems to work with geolocated data. Section 3 details the various means to conveniently visualize data according to their temporal, spatial or spatio-temporal structures. Probabilistic modeling starts in Section 4 with spatio-temporal covariance theory. The practice of statistical inference with the R packages `CompRandFld` (Padoan et al., 2015) and `gstat` (Pebesma, 2004) is developed on the pollution data example. Once the model is set and inferred, Section 4.3 provides guidelines for prediction and validation and discusses the relative merits of some R packages to achieve such tasks. In addition to this text, we provide extensive R code examples through supplementary material in the R Markdown [1] format that can be freely downloaded from the site: http://informatique-mia.inra.fr/resste/atelier.

2. Handling large spatio-temporal datasets with R

2.1. French pollution data

Ambient air pollution in Europe is assessed through the implementation of monitoring networks distributed across the countries. In addition, observation data are increasingly supplemented by air quality modeling systems which simulate the physical and chemical processes driving the temporal and spatial evolution of atmospheric concentrations.

The data used here as example cover the French territory for the year 2014. Time series of hourly and daily observed concentrations of four pollutants (ozone, nitrogen dioxide, particulate matter PM_{10} and $PM_{2.5}$) were retrieved from the national air quality database. They come from the continuous measurements carried out by the French associations responsible for air quality monitoring (the AASQAs) at 507 stations. Air quality is measured in different types of location characterized by the station environment (rural, suburban, urban) and the type of influence (background, traffic, industrial). For large scale estimation of atmospheric concentrations usually only background stations are considered. Although these data are carefully post processed, there are numerous missing values (42% for the daily database in 2014).

Time series of hourly and daily simulated concentrations of the same pollutants were obtained from the chemistry-transport model *CHIMERE* (Menut et al., 2013), which is implemented in the national system PREV'AIR (www.prevair.org). *CHIMERE* is an Eulerian off-line chemistry-transport model. External forcings are required to run a simulation: meteorological fields, primary pollutant emissions, chemical boundary conditions. Using these input data, *CHIMERE* calculates and provides the atmospheric concentrations of tens of gas-phase and aerosol species over local to continental domains. The key processes affecting the chemical concentrations represented in *CHIMERE* are: emissions, transport (advection and mixing), chemistry and deposition. The data used here were produced at an hourly time step, on a rectangular grid covering France with a spatial resolution of approximately 10 km.

[1] http://rmarkdown.rstudio.com/

To produce the most reliable pollutant maps, the national system PREV'AIR uses the simulated concentrations from *CHIMERE* as a drift for kriging the observations. For now, these maps are produced day by day, without consideration of the time dimension. The challenge taken up in this work is to improve the prediction by proposing a full spatio-temporal statistical model.

2.2. Importing the pollution files: how can `R` *efficiently handle large tables of data?*

The data files provided by Ineris are structured as follows:

- One `cvs` file related to measurement stations provides the attributes (in columns): code, name, country, start date, end date, type of area, classification of the measurement stations (in rows).
- Concerning observations, there are as many `cvs` files as pollutants$\times$observation times : each file is named by a string concatenating the pollutant and the year/month/day or year/month/day/hour of observation and contains as many rows as active measurement stations with their code, longitude, latitude and the record of the corresponding pollutant. For hourly data there are in total $4 \times 365 \times 24 = 35040$ files. Each file is of size 13 kB.
- For the *CHIMERE* model based data, `cvs` files are daily tables with gridded points as rows and longitude, latitude and the four pollutant outputs in columns. There are 365 files each of size 1 MB approximately.

In total, the data set is more than 1 GB. The very first task, not straightforward at first glance, is thus to import this huge amount of data into structured data objects and files.

The table of stations with their attributes such as their coordinates in decimal degrees is imported using the `read.table` command. Importing the other files in a similar way is very inefficient since this would require iterative calls to `rbind` within a loop. The `dataframe` under construction would keep growing iteratively, thereby slowing down the importation.

Instead, we resort to the `data.table` package (Dowle et al., 2015) and its powerful function `rbindlist`. The main stages of this importation routine can be illustrated for a single pollutant. More details can be found in the supplementary material.

```
library(data.table)
######### getting list of files to import
file     <- list.files('~data/OBS')
#########  example for PM10 files
filePM10 <- file[grep('PM10',file)]
#########  get time/date from file names
date_station <- gsub(".csv","",filePM10)
date_station <- gsub("PM10_","",date_station)
## Check if the files have the same length and have the same order for all pollutants
--> see Rmd File at \url{http://informatique-mia.inra.fr/resste/atelier}
######### preparing data as list of dataframe for rbindlist
PM10l <- vector(mode = "list", length = length(filePM10))
for(i in 1:length(PM10l)){
  date  <- strptime(date_station[i],format= "%Y%m%d%H")
  PM10t <- try(read.table(filePM10[i],header = FALSE,sep=" ",na.strings="-999",
               col.names = c("ID","long","lat","PM10"),
               colClasses = c("factor","numeric","numeric","numeric")))
  PM10t <- cbind(PM10t,rep(date,nrow(PM10t)))
  PM10l[[i]] <- PM10t}
```

Soumis au Journal de la Société Française de Statistique
File: main.tex, compiled with jsfds, version : 2009/12/09
date: September 16, 2016

```
######### Using rbindlist for fast row binding of all dataframes
PM10    <- rbindlist(PM10l)
names(PM10)[ncol(PM10)] <- 'date'
PM10$ID <- paste(PM10$ID,PM10$date,sep='_')
######### Using data.table for faster merge
PM10 <- data.table(PM10)
PM25 <- data.table(PM25)  ###(built in the same way as before)
OBS  <- merge(PM10[,.(PM10,ID2,date,long,lat)],
              PM25[,.(PM25,ID2,date1,long1,lat1)],by="ID",all=TRUE)
```

Other methods such as `plyr` or `dplyr` may also be worthwhile to manipulate and join such large files as advocated by Wickham and Francois (2016).

In the rest of the paper, our focus will be on estimation and prediction procedures. For this reason, the example that we will deal with has more workable dimensions (365 days $\times$507 stations $\times$4 pollutants), and we store the resulting tables as `dataframe` objects into Rdata files : (i) the stations with their characteristics (`stations.Rdata`); (ii) the daily observations (`OBS_jour.Rdata`) with station ID, longitude, latitude, date, and the four pollutant records as columns; (iii) the daily prediction table (`CHM.Rdata`) with as columns the longitude, latitude, the four pollutant forecasts and the day of prediction of the 111×101 gridded points ranging (in degrees) from $(41, -5)$ to $(52, 10)$.

2.3. Typing spatio-temporal pollution data with `spacetime`

The package `spacetime` proposes classes and methods for dealing with spatio-temporal data. It considers three formats for organizing data, corresponding to the three types of spatio-temporal data mentioned in the Introduction:

- time-wide format for which different columns reflect different moments in time,
- space-wide format for which different columns reflect different measurement locations or areas,
- long formats for which each record reflects a single time and space combination.

Long formats correspond to the type of spatio-temporal data considered in this paper. They are the most flexible type and they will be used all along this work. The full spatio-temporal information is held in a single column:

```
STATION     TIME        PM10    PM25    NO2     O3
station1    2015-01-01  24.645  .       8.234   67.123
station1    2015-01-02  36.765  .       7.233   89.234
station1    2015-01-03  23.233  .       8.219   90.111
    ...
station2    2015-01-01  32.860  35.233  .       45.222
station2    2015-01-02  33.460  12.231  .       23.433
    ...
```

The package `spacetime` (Pebesma, 2012) proposes four spatial layouts (full grid, sparse grid, irregular layout and simple trajectories). The Space Time Full Data Frame (`STFDF`) layout is convenient to store the well-structured *CHIMERE* model runs based on a regular grid that does not change between time points, while the Space Time Irregular Data Frame (`STIDF`) layout will be appropriate for the daily observations since those are not recorded for the same subset of

stations at every time step. Regular spatio-temporal structures such as the *CHIMERE* model data can be read as STIDF, but they are more efficiently stored as STFDF objects. As an example, the *CHIMERE* data for one single month counts 34.6 MB as STIDF and decreases to 10.8 MB as STFD.

```
load('data/OBS_jour.Rdata')
load('data/CHM.Rdata')
STIDF_jour <- stConstruct(OBS_jour,space=c('long','lat'),time='date',
                          SpatialObj=SpatialPoints(OBS_jour[,c('long','lat')]))
STFDF_jour <- as(STIDF_jour, "STFDF")

CHM_jour <- stConstruct(CHM[CHM$time>= "2013-12-31" & CHM$time <= "2014-02-01",],
                          space=c('lon','lat'),time='time')
CHM_jour <- as(CHM_jour, "STSDF")
```

Since spacetime deals with spatial data *and* time series, it is closely linked to the packages sp and xts that provide classes and methods for spatial data and for time series respectively. spacetime objects inherit from several sp and xts classes and methods: all functions and methods of these packages can be used when spacetime data are coerced to a single location or a single time. To exemplify the compound nature of spacetime, the str command applied to the STIDF object ST_OBS_juin_france shows that its various slots are essentially a concatenation of a dataframe with the pollutant records, a Spatial Point object from the sp package that gives the coordinates of the measurement location and an xts object that encodes a time index (Ryan and Ulrich (2014)). Most statisticians may be familiar with time series data, but the sp package merits some additional explanations of importance. This package allows defining specific classes of dataframe type according to different types of geometries including points, lines, polygons and grids. Advantages for using such spatial data classes are numerous, in particular thanks to the existence of specific spatial methods available for the classes in sp such as geometric transformations to plot planar maps of a surface belonging to a sphere.

One important feature of sp is to provide methods to easily perform geographic projection of data. Transforming from one coordinate reference system (geographic projection) to another one requires the package rgdal to be installed. Such transformations are of utmost importance for spatio-temporal data analysis since the calculation of distances is a prerequisite to compute covariance matrices based on a valid covariance function (*i.e.*, positive definite in the corresponding space). The following example transforms the French 2014 pollution data set from the original longitude and latitude coordinates into the WGS84 (World Geodesic System 1984, the one used by Google Earth and for GPS localization) and then from WGS84 to Lambert93 projection (the official projection for the French Metropolitan area since 2000). The differences between the two projection systems are blatant on Fig1 that displays the French PM_{10} concentration on the 15th of June 2014 for both systems of coordinates.

```
### Add projections
library(sp)
proj4string(CHM_jour)   <- "+proj=longlat +ellps=WGS84 +datum=WGS84+no_defs"
proj4string(STIDF_jour) <- "+proj=longlat +ellps=WGS84+datum=WGS84+no_defs"
```

Other examples are available in the supplementary material. More information on how geographic reference systems work can be found in Burkard (1964).

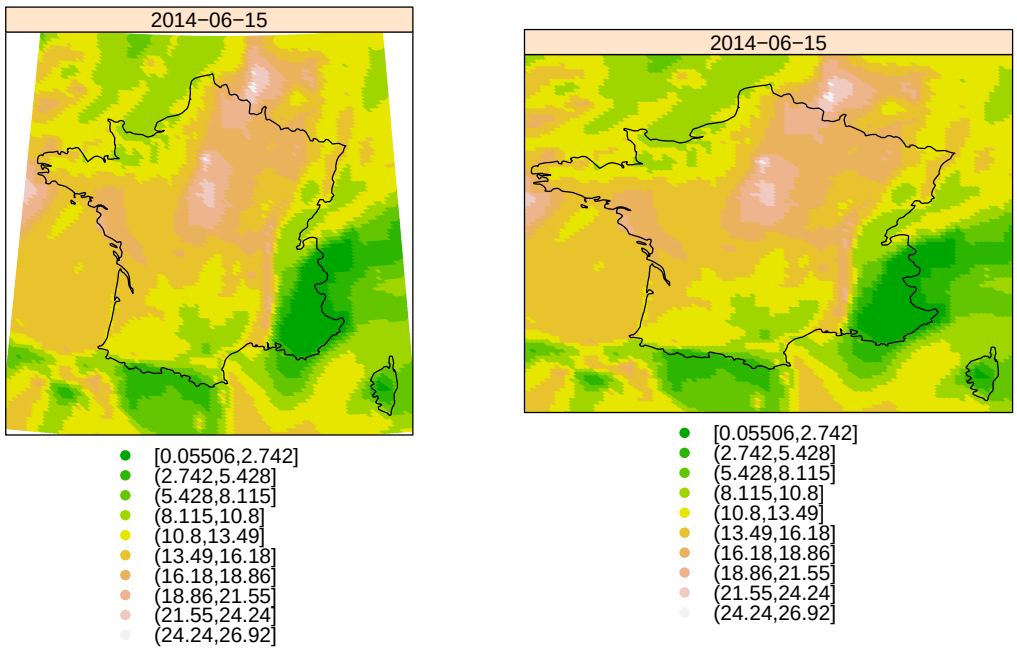

FIGURE 1. *Lambert93 (left) vs WSG84 (right) projection systems for the French PM_{10} concentration forecasts by the* CHIMERE *model on the 15^{th} of June 2014.*

3. Visualizing spatio-temporal data and exploring their dependencies

Any statistical data analysis starts with a visual exploration of the data in order to gain insight into the data set and to guide the statistical modeling of the dependencies in time, in space and over several variables. A wide number of `R` packages are available to plot spatial or temporal data, but only few are of interest for exploring the type of data set we are interested in. The most important packages are listed in Table 3. In this section, we only discuss their specificities and we refer to the supplementary material for a detailed account of how to visualize spatio-temporal data and how to analyze their spatio-temporal dependencies. We also present the packages `shiny` and `leaflet`, which provide interactive tools to render the exploratory analysis easier and user-friendlier.

3.1. Plotting spatio-temporal data

Holding spatio-temporal data in long tables (see Section 2.3) is very helpful for handling the data and also for plotting them. Visualizing spatio-temporal data often consists in displaying change in a given dimension (e.g. space) over the other dimension (e.g. time) to highlight particular features in the data. The different spatio-temporal plot types are defined below. They are illustrated using the function `stplot` of the package `spacetime` by passing a spacetime object as an argument along with the names of columns we want to plot. This function allows users to add multiple layers or symbols using the `sp.layout` argument.

TABLE 1. *Main packages and related functions used to plot and analyze the structures of dependence of (multivariate) spatio-temporal data.*

Spatial location	Time	Number of variables	Vizualisation		Dependencies	
			Package	Function	Package	Function
1	n_t	1	tseries ggplot2	irts autoplot	stats	acf
1	n_t	N	tseries ggplot2	irts autoplot	stats	acf
n_s	1	1	sp plotKML	spplot plotKML	gstat gstat	variogram plot.gstatVariogram
n_s	1	N	sp	spplot	gstat gstat	variogram plot.gstatVariogram
n_s	n_t	1	tseries ggplot2 plotKML spacetime	irts autoplot plotKML stplot	stats gstat gstat	acf variogram plot.StVariogram
n_s	n_t	N	sp	spplot		

– An *animation* (or a movie) is the most intuitive dynamical plot to examine the evolution of the spatial data through time.
– *Space (1-D)/time plots* display data in a space-time cross section such as the Hovmöller diagram, see the left display of Fig. 2. This two-dimensional plot represents space on the x-axis (longitude, station, ...) and time on the y-axis.

```
IDs <- levels(OBS_jour$ID)
sel <- NULL; for(i in 20:24) sel<-c(sel,which(IDs==IDs[i]))
stplot(STFDF_jour[sel, "2014-01-01::2014-01-30",'O3'], mode="xt", scaleX=0,
       col.regions=terrain.colors(100))
```

– *Time series plots* are particularly useful to compare time series associated with different stations, see the right display of Fig. 2.

```
stplot(STFDF_jour[sel,"2014-01-01::2014-01-30",'O3'],mode="ts")
```

– *Multi-panel spatial maps* are used to compare spatial maps for given times or aggregates over time, see Fig 3.

```
world <- getMap(resolution="low")
stplot(STFDF_jour[,"2014-01-01::2014-01-04",'O3'],col.regions=
       brewer.pal(6,"Spectral"),cuts=6,sp.layout=list(world,first=TRUE))
```

Multiple layers can be enhanced by using the package `ggplot2` (Wickham, 2009) or `plotKML` (Hengl et al., 2015) where the latter converts spacetime class objects into KML (Keyhole Markup Language) for use in virtual globes. It also allows an interactive visualization of the data.

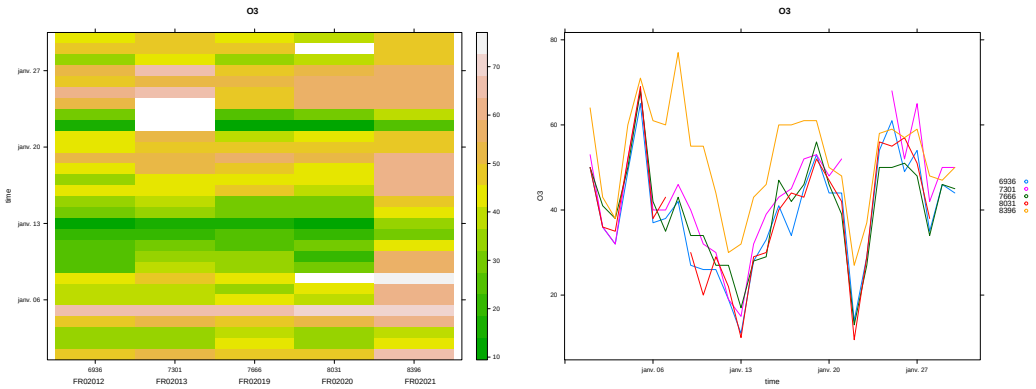

FIGURE 2. *Left: Hovmöller diagram. Right: Time series plots*

Soumis au Journal de la Société Française de Statistique
File: main.tex, compiled with jsfds, version : 2009/12/09
date: September 16, 2016

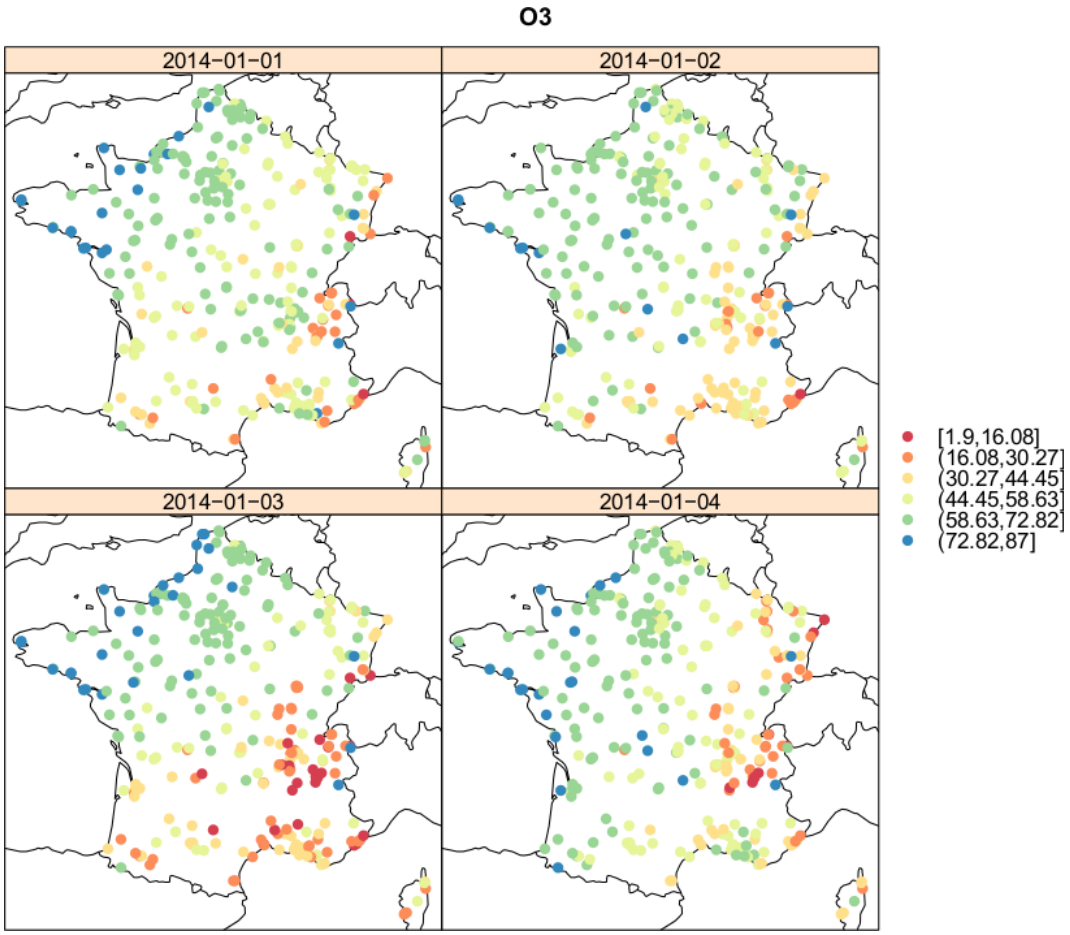

FIGURE 3. *Multi-panel plots*

3.2. *Exploring spatio-temporal dependencies*

The dependence in space and/or time between observations is a major component of statistical modeling of spatio-temporal data. As will be seen in the next section, a good description of the spatio-temporal covariance structure is the key for an efficient prediction. Thus, we shall examine the empirical covariance functions and/or variograms at various space and time lags. For a Gaussian process $\{Z(s,t)\}$, the semi-variogram γ is defined as

$$\gamma((s_1,t_1),(s_2,t_2)) = 0.5\mathbb{E}\left[Z(s_1,t_1) - Z(s_2,t_2)\right]^2, \tag{1}$$

and in our stationary set-up we get a one-to-one link to the covariance function via

$$\gamma((s_1,t),(s_2,t_2)) = \gamma(s_2 - s_1, t_2 - t_1) = \sigma^2 - C(s_2 - s_1, t_2 - t_1)$$

with the sill $\sigma^2 = C(0,0)$. The empirical variogram cloud can be calculated as the empirical equivalent of (1) for all observed space-time lags. To obtain an empirical space-time semi-variogram that can be visually interpreted, one usually groups together space-time lags into a set of space-time bins. More formally, one can define a space-time grid of bin centers $(\tilde{s}_k, \tilde{t}_k)$ resulting in distance classes $D_k = \{(s_i,t_i) : \text{dist}((s_i,t_i),(\tilde{s}_k,\tilde{t}_k)) \leq \varepsilon_k\}$, $k = 1,\ldots,K$, partitioning the set of observed space-time lags (s_i,t_i). The package `gstat` uses the `spacetime` classes and allows us to calculate and plot the sample variogram of the data as shown in the following example and illustrated in Fig. 4.

```
vvst <- variogram(O3~1, STFDF_jour[!is.na(STFDF_jour[,"2014-01-01::2014-12-31","O3"]$O3),
                  "2014-01-01::2014-12-31","O3"], width=25,cutoff=400,tlags=0:5)
plot(vvst)                 # sample variograms at each time lag
plot(vvst,map=FALSE)       # ST-variogram map
plot(vvst,wireframe=TRUE) # ST-variogram wireframe
```

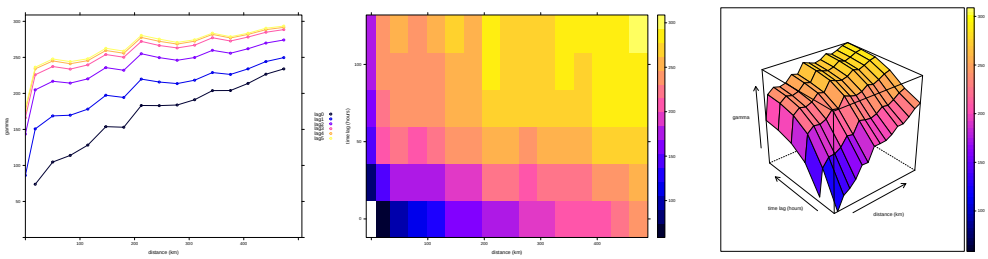

FIGURE 4. *Sample variograms for different time lags (left); spatio-temporal sample variogram map (middle); the same map represented as a wireframe plot (right).*

3.3. *`Shiny`/`Leaflet` application for spatio-temporal data*

To explore and visualize spatio-temporal dynamics easily, an interactive tool may be useful in order to

– switch easily between cross plots of different types by exchanging time, space and variables;

– select specific areas or specific periods;
– zoom in on time or space;
– superpose layers based on different variables and/or time periods;
– add background maps;
– click on a location to display its attributes or some time series at station scale;
– display both maps and tables of statistical results.

The `shiny` package makes incredibly easy to build such interactive web applications. The preceding plots can be integrated in this web tool and linked to widget controls allowing us to parametrize how to sample the dataframe and what and how to plot. An interactive modeling procedure like the one proposed in following sections could also be integrated in a Shiny application. Moreover, when a `shiny` application is coupled with the `leaflet` package, it becomes very easy to propose interactive maps, to add background maps across the web using platform-independent calls (like Web Feature Service - WFS), to navigate and to zoom into the displayed maps.

As a complementary material to this paper, we developed a small `R shiny/leaflet` application based on the INERIS air pollution data to illustrate how such an application can facilitate the exploration and visualization of spatio-temporal data. It implements all types of plots and analysis of statio-temporal dependence done in the preceding section and can be accessed and tested at the following URL: https://resste.shinyapps.io/App1/. The related code is accessible in the supplementary material.

4. Modeling and predicting spatio-temporal processes

We assume that the spatio-temporal random field can be decomposed as

$$Z(s,t) = \mu(s,t) + Y(s,t), \quad (s,t) \in D \times T$$

where $\mu(s,t) = \mathbb{E}[Z(s,t)]$ is the deterministic part, or spatio-temporal trend and $Y(s,t)$ is a zero-mean second-order stationary spatio-temporal random field with covariance function

$$\begin{aligned} C_{ST}(h,u) &= \mathrm{Cov}\{Z(s,t), Z(s+h,t+u)\} \\ &= \mathbb{E}[Y(s,t).Y(s+h,t+u)] \end{aligned}$$

where (s,t) and $(s+h,t+u) \in D \times T$. This statement assumes that all the non-stationarities of the process are accounted in the deterministic part, which is generally quite difficult to verify. The spatio-temporal field $Z(s,t)$, $(s,t) \in D \times T$ is sampled at locations and times (s_i,t_i), $i = 1,\ldots,n$ where n is the number of space-time data. Ideally, measurements are available for all times at each location, but very often data are missing for some times at some locations.

4.1. Modeling the deterministic part

Predicting the spatio-temporal field $Z(s,t)$ at an unsampled spatio-temporal coordinate $(s_0,t_0) \in D \times T$ from available measurements $\{z(s_i,t_i)\}_{i=1,\ldots,n}$, requires a model for the trend $\mu(s,t)$ and a covariance model for the random part $Y(s,t)$. A variety of models exist for modeling the deterministic part, including covariates, parametric or non parametric, linear or non linear. The specificity for modeling the expectation of a spatio-temporal field is that the time can sometimes

be considered as a covariate, and hence is not taken into account in the covariance structure. Several packages propose this option. In the package `SpatioTemporal` (Lindström et al., 2013) the model for $\mu(s,t)$ is

$$\mu(s,t) = \sum_{\ell=1}^{L} \gamma_\ell \mathscr{M}_\ell(s,t) + \sum_{i=1}^{m} \beta_i(s) f_i(t), \tag{2}$$

where $\mathscr{M}_\ell(s,t)$ are spatio-temporal covariates with coefficients γ_ℓ , $\{f_i(t)\}_i$ are temporal basis functions obtained by singular values decomposition, $\beta_i(s)$ are spatially varying coefficients obtained for instance by universal kriging and may depend on covariates. $Y(s,t)$ is therefore a spatial field independent in time and stationary in space. The package options allow for the choice of the basis $\{f_i(t)\}_i$, completion of missing data, maximum likelihood (profile or REML) estimation, prediction and associated variance computation and cross validation.

In the package `Stem` (Cameletti, 2009) the time component of the model is assumed to have a an AR(1) structure. The data are sampled according to a factorial design $n_S \times n_T$ where n_S is the number of spatial stations with measurement device and n_T is the number of time measurements. For $t = 1,\ldots,T$, the model is

$$\begin{aligned} Z_t &= X_t\beta + KY_t + e_t, \qquad e_t \sim \mathscr{N}_{n_S}(0,\Sigma_e), \\ Y_t &= GY_{t-1} + \eta_t, \qquad \eta_t \sim \mathscr{N}_p(0,\Sigma_\eta), \end{aligned} \tag{3}$$

where Z_t are n_S measurements at stations, X_t is a $L \times n_S$ design matrix, accounting for the covariates, K is a $n_S \times p$ matrix and Y_t is a p-dimensional latent process with $Y_0 \sim \mathscr{N}_p(m_0, C_0)$. The innovations e_t and η_t are Gaussian, temporally independent vectors with covariance matrices Σ_e, respectively Σ_η. The package options allow for parameter estimation through the EM algorithm, prediction at unsampled locations and estimation of the variance error by bootstrap.

4.2. Modeling the covariance structure

A major step in fitting a Gaussian model to data is the estimation of the covariance function. For simplicity of exposition, we assume in this section that the spatio-temporal trend has been removed up to a constant and that a stationary model is suitable for the data. Parameters to fix or to estimate are the mean $\theta_{\text{mean}} \equiv \mu$, the global sill $\theta_{\text{sill}} = C_{ST}(0,0)$, the nugget $\theta_{\text{nugget}} = C_{ST}(0,0) - \lim_{u\to 0, h\to 0} C(h,u)$ and parameters related only to the correlation function like the scale, shape, geometric anisotropy with a rotation angle and dilatation parameter, etc. Notice that we can simply estimate θ_{mean} and θ_{sill} empirically through the empirical mean and variance of data.

Covariance functions must be nonnegatively definite, limiting the choice of available models and making the construction of models with realistic features intricate. The definition of spatio-temporal covariance functions that appropriately capture complex interactions between space and time has become a very active field of research. This section gives a practical introduction to functionality for parametric space-time covariance estimation through packages available in `R`. We focus on packages whose space-time estimation capacities are already fully developed and documented at the current stage. We further require a relatively large range of space-time models beyond space-time separability, leading us to consider in more detail the packages `gstat` (directly using R objects of `spacetime` type) and `CompRandFld`. We underline that this selection is by no means exhaustive.

4.2.1. Space-time covariance models

Most of the commonly used space-time covariance models are built by modifying or combining generic covariance models defined for $\mathbb{R}^d$, $d = 1,2,\ldots$. These basic models are usually stationary and isotropic such that $C(s_1,s_2) = C(s_2 - s_1) = C(s_1 - s_2) = C(\|s_2 - s_1\|)$ for $s_1,s_2 \in D$. Commonly known generic models are covariance functions of exponential, powered exponential, Matérn or Cauchy type, amongst many others. In the following, we will assume that some kind of temporal dependence is present in data, excluding the trivial case of independent temporal replications. As discussed in the Introduction section, very simple space-time models with planar $\mathbb{R}^2$ space can be obtained by using a generic covariance C defined over $\mathbb{R}^3$ and by considering one of its dimensions as time. These models $C(s_1,s_2,t)$ are highly simplistic since they ignore the special role of the time arrow. Space-time separable covariance models are constructed using a spatial and a temporal covariance assuming that dependence exists in space and time separately without interaction between the spatial and the temporal domain:

$$C_{ST}(h,u) = C_S(h)\, C_T(u),$$

where $C_S(\cdot)$ is a spatial covariance defined over $\mathbb{R}^2$ and $C_T(\cdot)$ is a temporal covariance defined over $\mathbb{R}^1$. An advantage of such models is that separability leads to a reduced number of parameters, allows separate estimation of the parameters related to space or time and provides faster computation of the inverse and of the determinant of the matrix involved in kriging. However, separability is often an overly simplistic assumption because it is equivalent to conditional independence between $Y(s,t)$ and $Y(s',t')$ given $Y(s',t)$. As a consequence, there is a simple proportionality relationship between $C_{ST}(h,u)$ and $C_{ST}(h',u)$ for two fixed spatial lags h and h'. Separable covariances cannot capture sophisticated interactions between space and time.

Various nonseparable model classes have been developed in the literature, among them models of so-called Gneiting, Iaco-Césare or Porcu type, variants of which are available in the `CompRandFld` package. The Gneiting class (Gneiting, 2002) provides flexible nonseparable isotropic space-time models defined as

$$C_{ST}(h,u) = (\psi_T(u)+1)^{-\delta/2} C_S(h/\sqrt{(\psi_T(u)+1)}), \tag{4}$$

where ψ_T is a function that can be written as $\psi_T(u) = c + \gamma_T(u)$ with $c > 0$ and $\gamma_T(\cdot)$ a temporal variogram function. In Eq. (4), $C_S(\cdot)$ is a spatial covariance function that must be from the class of mixtures of Gaussian covariances for validity (exponential, Matérn, Cauchy, ...).

Certain simpler types of nonseparable constructions have also proven to be useful in practice; some of them are implemented in `gstat`. The relatively simple *metric* model

$$C_{ST}(h,u) = C_{joint}\left(\sqrt{h^2 + (\kappa.u)^2}\right) \quad \kappa > 0,$$

is based on a $\mathbb{R}^3$-covariance C_{joint} modified through a time anisotropy parameter κ. *Product-sum* covariances are nonseparable and defined as follows:

$$C_{ST}(h,u) = kC_S(h)C_T(u) + C_S(h) + C_T(u), \quad k > 0,$$

with a spatial covariance $C_S(\cdot)$ and a temporal covariance $C_T(\cdot)$. Notice that the sill parameter of the spatio-temporal covariance C_{ST} is related to the sill parameter C_S and C_T thanks to the parameter k. Finally, the *sum-metric* models combine the *metric* and *product-sum* constructions.

We emphasize that various kinds of space-time anisotropy can be considered. Purely spatial anisotropy structures can be taken over into space-time models. More specific time-related anisotropy structures may be of interest, for instance describing a certain directional movement in space parametrized by a velocity vector. These are not yet implemented in `R` packages, but can be handled by the user without too much difficulties.

4.2.2. Estimation approaches

Several estimation approaches exist, and it would be impossible to cover all of them in this review. We limit our exposition to the most commonly used approaches of point estimation of stationary space-time correlation functions. We here consider the two most commonly used classes of estimation techniques: weighted least-squares estimation (WLS) based on contrasting the empirical variogram with the parametric theoretical one and likelihood-based techniques. `CompRandFld` implements both of them, whereas `gstat` focuses on WLS. A major complication in the estimation of parametric space-time models comes from the high dimensionality of data. We start by looking at least squares estimation, which can be considered as being relatively robust, in full analogy to the purely spatial context. To apply the WLS approach, we now suppose that an empirical variogram $\hat{\gamma}_k$ has been calculated as in Section 3 and that the parametric variogram γ_θ (and therefore the covariance) depends on a parameter vector θ and takes values $\gamma_{\theta,k}$ at the bin centers $(\tilde{s}_k, \tilde{t}_k)$. We have to solve numerically the optimization problem

$$\theta^\star = \arg\min_\theta \sum_{k=1}^{K} \omega_k \left|\gamma_{\theta,k} - \hat{\gamma}_k\right|^2$$

with weights ω_k, for which different choices can be useful. In particular, $\omega_k = 1$ leads to ordinary least squares, whereas choices such as $\omega_k = |D_k|/\gamma_{\theta,k}^2$ (Cressie, 1993) try to correct for the variance of $\hat{\gamma}_k$ that varies with the number of observations falling into a bin and the true variogram value. Other typical choices overweight relatively small distances in space and time, therefore putting the focus to capturing small distance dependence behavior which is often very important in practice for space-time prediction through kriging.

Working with the classical full likelihood necessitates the inversion of the variance-covariance matrix for the data observation points and the calculation of its determinant, which is most often computationally prohibitive in a space-time context. For a vector x of n standard Gaussian space-time observations and a parametric correlation matrix $C_\theta = \{C((s_{i_1}, t_{i_1}), (s_{i_2}, t_{i_2}))\}_{1 \le i_1, i_2 \le n}$, the likelihood is

$$\theta \mapsto \varphi_{C_\theta}(x) = (2\pi)^{-n/2} |C_\theta|^{-1/2} \exp\left(-0.5 x' C_\theta^{-1} x\right). \tag{5}$$

If n is large (say, larger than around 5000) such that the numerical maximization of (5) is not possible owing to the high computing cost of the determinant $|C|$ and the inverse C^{-1} , we can fall back on composite likelihoods or try covariance tapering.

Covariance tapering (Furrer et al., 2006; Stein, 2013) multiplies the parametric covariance function that we want to estimate with another compactly supported covariance function (the

taper) to obtain a new, valid *tapered* covariance function whose value is exactly 0 for data points that are farther apart than the taper's support in space or in time. This construction yields a sparse structure in the resulting covariance matrix and facilitates computations considerably. Notice however that this approximation technique induces estimation bias which may be rather strong depending on the choice of tapering function, especially if there still is relatively strong dependence in data over long distances.

The idea of composite likelihood (Lindsay, 1988; Varin et al., 2011) is to (falsely) assume independence between certain blocks of the data and therefore requires to maximize the composite likelihood $\theta \mapsto \sum_{\text{blocks}} \varphi_{C_{\text{block}}}(x_{\text{block}})$ to obtain the estimate $\theta^\star$, thus avoiding heavy matrix computations if blocks are relatively small. A classical choice is the pairwise likelihood, where blocks consist of pairs of observations. Three choices of Gaussian likelihoods could be used for pairs and are implemented in `CompRandFld`: the usual bivariate marginal likelihood, the bivariate conditional likelihood and the likelihood of the difference, denoted respectively $l(x_i, x_j)$, $l(x_i \mid x_j)$ and $l(x_i - x_j)$. Similar to the full likelihood, composite likelihood methods lead to asymptotic consistency and asymptotic normality, and classical likelihood-based statistical tests have been adapted to composite likelihood. The number of pair terms in the pairwise likelihood would be prohibitively high in our high-dimensional space-time setting, but we can use a weighted pairwise likelihood and keep only pairs for data points that are relatively close in space and time by applying $0/1$ weights. If such weights are well chosen, the good estimation performance of the pairwise likelihood is usually not impaired, even for multivariate spatio-temporal data (Bourotte et al., 2016). We can moreover shift focus to a good modeling of the dependence at relatively short distances in space and time.

4.2.3. Models and estimation approaches in `gstat` and `CompRandFld`

The package `gstat` is closely intertwined with the `spacetime` package, allowing to conveniently use space-time data structures. Beyond generic spatial covariance models, certain relatively simple space-time model constructions are supported: *separable*, *metric*, *product-sum* and *sum-metric*. Empirical space-time variograms can be calculated and plotted using the functions `variogramST` and `plot.StVariogram`. Spatial variogram models (`vgm`) and space-time variants (`vgmST`) can be defined. Parametric estimation is possible through weighted least squares with different choices for weights (`fit.StVariogram`). Scale parameters related to time anisotropy can be estimated (argument `stAni` of `fit.StVariogram` for a different scale along the time dimension), whereas spatial geometric anisotropy parameters can be fixed manually in the `vgm` object used to construct a `vgmST` model. `gstat` also handles missing data in estimation but not in kriging with its `krigeST` function, assuming that they are non informative. Empirical and theoretical space-time variograms can be compared through a number of dedicated plotting tools.

`CompRandFld` does not apply specific space-time data structures but comes with a well-designed, tidy user interface that makes its use straightforward. The data object with observed values is an array whose dimension depends on the type of data: a vector for spatial data without repetition observed at irregular sites, a square matrix for spatial data observed on a regular grid without repetition, a matrix for a single space-time observation, an array with three dimensions for repeated space-time observations, an array with four dimensions for repeated space-time observations on a regular grid, etc. The current version of `CompRandFld` does not support missing

data and full data must be given for a Cartesian product of spatial sites and time points. Beyond the possibility to use a separable model, it implements flexible classes of space-time covariances like variants of models from the so-called Gneiting, Porcu or Iaco-Cesare classes (Padoan et al., 2015). In each case, the model is parametrized to make appear a nonseparability parameter η. The Gneiting model is available through a power variogram ψ_T and the powered exponential spatial covariance C_S. The nonseparability parameter η is made explicit through the following reparametrization:

$$C_{ST}(h,u) = \sigma^2 g_T(u)^{-1} \exp\left(-\frac{d_S(h)}{g_T(u)^{0.5\eta\theta_{S,\text{shape}}}}\right) \quad \text{with} \tag{6}$$
$$d_T(u) = (u/\theta_{T,\text{scale}})^{\theta_{T,\text{shape}}},$$
$$d_S(h) = (h/\theta_{S,\text{scale}})^{\theta_{S,\text{shape}}},$$
$$g_T(u) = 1 + d_T(u),$$

where σ^2 is the sill and $0 \leq \eta \leq 1$ is the nonseparability parameter with separability for $\eta = 0$. `CompRandFld` provides empirical estimation and parametric weighted least squares based on the variogram (`EVariogram`,`WLeastSquare`), but its main asset certainly are the large variety of likelihood-based estimators (`FitComposite`) including full likelihood, pairwise likelihoods with cut-off distances for $0/1$-weighting (through arguments `maxdist`,`maxtime`) and the tapered likelihood with various choices of space-time separable and nonseparable covariance tapers (through arguments `taper`,`tapsep`). Model selection criteria like AIC or the CLIC for pairwise likelihood are available by setting the boolean argument `varest` of `FitComposite`, and statistical hypothesis tests for nested models can be applied (`Hypotest`), even in the pairwise setting.

Beyond `gstat` and `CompRandFld`, several other packages implement interesting tools related to space-time covariance estimation. For instance, simulation and calculation facilities for a large variety of space-time covariances can be found in the `RandomFields` package, which provides generic models and operators to combine them, multivariate models and specific nonseparable space-time model classes.

4.2.4. Fitting a model to the INERIS data

Air pollution, especially that of non-background measurement stations close to some specific source of pollution (road, industrial site, ...), can show strong nonstationarities both over space and over seasons. Therefore, we here propose to model residuals of observations with respect to the predictions of the chemical-physical *CHIMERE* model as a stationary Gaussian random field. *CHIMERE* assimilates observed data into a model for both forecasts and hindcasts with output given on a $10km \times 10km$ grid. Since the *CHIMERE* model usually does not capture well strong local nonstationarities owing to specific pollution sources, we will only use background stations from the data to avoid complex spatial nonstationarities. To avoid strong temporal nonstationarities and to keep a reasonable amount of data for space-time estimation procedures, we further concentrate our analysis on daily data for a 2014 subperiod spanning January 1 to March 31. We illustrate univariate fitting of a space-time model on the PM_{10} variable. In practice, the model we are fitting here could be useful to correct predictions made by the *CHIMERE* model.

Preparing data

We start by extracting the relevant station IDs.

```
IDs.bg   <- stations$station_european_code[stations$type_of_station=="Background"]
IDs.fr   <- stations$station_european_code[stations$country_name=="France"]
IDs.bgfr <- intersect(IDs.bg,IDs.fr)
```

We now create an object of type `SpatialPoints` for the coordinates of retained sites, which are then transformed from longitude-latitude into the Lambert 93 system. Selected data are stocked into an `STFDF` object `OBS_sel`.

```
idx    <- which(stations$station_european_code %in% IDs.bgfr)
coord  <- cbind(stations$station_longitude_deg,stations$station_latitude_deg)[idx,]
tstart <- "2014-01-01"
tend   <- "2014-03-31"
OBS_jour <- OBS_jour[OBS_jour$ID %in% IDs.bgfr,]
OBS_sel  <- stConstruct(OBS_jour,space=c('long','lat'),time='date',
                        SpatialObj=SpatialPoints(OBS_jour[,c('long','lat')]))
OBS_sel  <- as(OBS_sel,"STFDF")
proj4string(OBS_sel) <- "+init=epsg:4326"
OBS_sel  <- OBS_sel[,paste0(tstart,"::",tend)]
OBS_sel@sp <- spTransform(OBS_sel@sp, CRS("+init=epsg:2154"))
```

Next, we interpolate the *CHIMERE* values from grid points to observation sites through a simple bilinear interpolation with the `interp.surface(...)` function of the `fields` package, and we calculate the residual observations with respect to this mean trend. The residual data are stocked as `PM10res` in `OBS_sel`, and we put the interpolated *CHIMERE* values in a variable `CHM`.

```
CHM_sel  <- CHM[as.character(CHM$time)>=tstart&as.character(CHM$time)<=tend,]
grid.CHM <- unique(cbind(CHM_sel$lon,CHM_sel$lat))
CHM_mat  <- matrix(CHM_sel$PM10, nrow=nrow(grid.CHM))
dim(CHM_mat)
lon.grid <- sort(unique(CHM_sel$lon))
lat.grid <- sort(unique(CHM_sel$lat))
ind.sites.obs <- match(unique(OBS_sel@data$ID),IDs.bgfr)
fun <- function(i){
  tmp <- list(x=lon.grid,y=lat.grid,
              z=matrix(CHM_mat[,i],length(lon.grid),length(lat.grid)))
  interp.surface(tmp, coord[ind.sites.obs,])}
CHM_sites            <- sapply(1:ncol(CHM_mat),fun)
OBS_sel@data$CHM     <- as.numeric(CHM_sites)
OBS_sel@data$PM10res <- OBS_sel@data$PM10-as.numeric(CHM_sites)
```

Estimation with `CompRandFld`

Due to the considerable amount of missing data scattered over the space-time observation points, finding a combination of sites and days with good coverage and no missing data is intricate. We have removed days and sites with many missing data to reach a relatively large matrix of data without missing values for 70 days and 103 sites, stocked in a 70×103 matrix `data` used in the following. The indices of the retained days are given in a vector `times`, and coordinates of retained sites are in a 103×2 matrix `coord.sel`. We start by estimating the mean and the global sill from these data:

```
mean.est <- mean(data)
var.est  <- var(as.numeric(data))
```

yielding values `mean.est=8.02` and `var.est=116`. Next, we estimate an empirical space-time variogram, where the maximal spatial distance is fixed to 500 km and the maximal time lag is 10 days.

```
vgm.emp <- EVariogram(data=data,coordx=coord.sel,coordt=times,cloud=F,
                      maxdist=500,maxtime=10)
```

Owing to the size of the data set with $70 \times 103 = 7210$ values, full maximum likelihood estimation would be computationally extremely heavy. Instead, we will maximize the pairwise likelihood or a WLS contrast. Let us first define a number of more or less complex models. We here check a simple separable model with exponential covariance in space and in time and with or without a nugget effect, and we further consider the Gneiting model (6) with various parameters held fixed.

```
cormod1="exp_exp"
fixed1=list(mean=mean.est,nugget=0,sill=var.est)
start1=list(scale_s=200,scale_t=2)
cormod2="exp_exp"
fixed2=list(mean=mean.est,sill=var.est)
start2=list(scale_s=200,scale_t=2,nugget=0)
cormod3="gneiting"
fixed3=list(sill=var.est,mean=mean.est,nugget=0,power_s=1,power_t=1)
start3=list(scale_s=200,scale_t=2,sep=.5)
cormod4="gneiting"
fixed4=list(sill=var.est,mean=mean.est,power_s=1,power_t=1)
start4=list(scale_s=200,scale_t=2,sep=.5)
cormod5="gneiting"
fixed5=list(sill=var.est,mean=mean.est,nugget=0,power_s=.5,power_t=.5)
start5=list(scale_s=200,scale_t=2,sep=.5)
cormod6="gneiting"
fixed6=list(sill=var.est,mean=mean.est,power_s=.5,power_t=.5)
start6=list(scale_s=200,scale_t=2,sep=.5)
cormod7="gneiting"
fixed7=list(sill=var.est,mean=mean.est,nugget=0)
start7=list(scale_s=200,scale_t=2,sep=.5,power_s=.5,power_t=.5)
cormod8="gneiting"
fixed8=list(sill=var.est,mean=mean.est)
start8=list(scale_s=200,scale_t=2,sep=.5,power_s=.5,power_t=.5, nugget=0)
```

Notice that other nonseparable models implemented in `CompRandFld`, like variants of the Iaco-Cesare and Porcu models, have syntax similar to the Gneiting model; we have to replace the `cormod` character string with `"iacocesare"`, `"porcu"`, etc. We now estimate the first model by pairwise likelihood with the `FitComposite` function. Fixing its argument `varest=TRUE` allows estimating the standard errors and the CLIC. Cut-off distances for the pairs to be considered in the unconditional pairwise likelihood have been fixed to 300 km and 4 days after a visual inspection of the empirical space-time variogram. We run the optimization and compare empirical and parametric variogram estimates:

```
fit <- FitComposite(data=data,coordx=coord.sel,coordt=times,maxdist=300,maxtime=4,
                    corrmodel=cormod1,likelihood="Marginal",type="Pairwise",
                    fixed=fixed1,start=start1)
```

`Covariogram` is a plotting and calculation function for the fitted model variogram and covari-

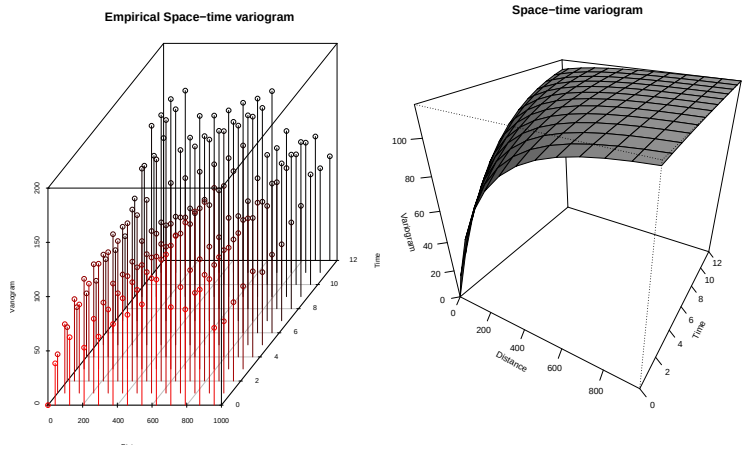

FIGURE 5. *Empirical and fitted variograms for model 1 adjusted with the composite likelihood approach of* `CompRandFld`*, here plotted with the* `Covariogram`*function.*

ance functions; see Figure 5 for the output produced by the following command:

```
Covariogram(fit,vario=vgm.emp,show.vario=T,pch=20)
```

Alternatively, we could use WLS for estimating parameters,

```
fit <- WLeastSquare(data=data,coordx=coord.sel,coordt=times,maxdist=300,maxtime=4,
                    corrmodel=cormod1,fixed=fixed1,start=start1,weighted=T)
```

Finally, we can estimate all models by either WLS or PL through

```
fitsWLS <- fitsPL <- vector("list", 8)
for(i in 1:8){
   fitsWLS[[i]] <- WLeastSquare(data=data,coordx=coord.sel,coordt=times,maxdist=300,
      maxtime=4,corrmodel=get(paste0("cormod",i)),fixed=get(paste0("fixed",i)),
      start=get(paste0("start",i)),weighted=T)
   fitsPL[[i]]=FitComposite(data=data,coordx=coord.sel,coordt=times,maxdist=300,
      maxtime=4,corrmodel=get(paste0("cormod",i)),likelihood="Marginal",type="Pairwise",
      fixed=get(paste0("fixed",i)),start=get(paste0("start",i)),varest=T) }
```

For pairwise likelihood fits, we can extract the index of the best fitting model with lowest CLIC value through `which.min(sapply(fitsPL,getElement,"clic"))`, and here the "most complex" model 8 is retained; that is, the Gneiting model with a nugget where all parameters have been estimated from data through the pairwise likelihood. Notice that the Gneiting models 5,7 and 8 are embedded. To illustrate the possibilities of testing hypotheses, we here test if the nugget of model 8 really is significant (with respect to model 7), and if the shape parameters of model 7 could be equal to 0.5 (model 5). We opt for the pairwise likelihood version of Wald's statistic:

```
fit8 <- fitsPL[[8]];fit7=fitsPL[[7]];fit5=fitsPL[[5]]
HypoTest(fit8,fit7,fit5,statistic="Wald")
```

and get the output

```
Soumis au Journal de la Société Française de Statistique
File: main.tex, compiled with jsfds, version : 2009/12/09
date: September 16, 2016
```

	Num.Par	Diff.Par	Df	Chisq	Pr(>chisq)
fit8	6	NA	NA	NA	NA
fit7	5	1	1	0.122	7.27e-01
fit5	3	2	2	205.3	2.6e-45

The nugget of model 8 is not significant at level 0.05, but models 5 and 7 differ significantly. We could therefore decide to keep model 7. Finally, we remark that `CompRandFld` comes along with the function `Covmatrix` that returns covariance matrices according to a given parametric model:

```
covvals <- Covmatrix(coordx=cbind(0:50*10,0),coordt=0:9,corrmodel=cormod7,
                     param=as.list(c(fit7$fixed,fit7$param)))$covmatrix
```

Estimation with `gstat`

As before, we consider the `PM10res` variable of residual PM_{10} observations with respect to *CHIMERE* output. `gstat` provides the `fit.StVariogram` function to fit parametric variogram models to the empirical space-time variogram through weighted least squares. It returns a spatio-temporal variogram of class `StVariogramModel`. Missing values are managed in the empirical variogram computation. In its arguments, `fit.StVariogram` requires to set the type of model to fit along with initial values and the empirical variogram object. Further arguments like upper and lower bounds, weights of the squared residuals and options to the `R` standard solver `optim` can be passed. We will try fitting candidate models according to the four basic space-time model types supported by `gstat`. To improve the convergence in the automatic fitting procedure implemented through `fit.StVariogram`, giving good initial parameter ranges as first guesses has proven to be mandatory. Including nested structures in the model is however not possible as the automatic fitting procedure will keep only one structure. Instead, nested structures can be fitted by a rather tedious manual fitting approach, which however may significantly improve the goodness of fit. Adjusting first the marginal temporal variogram (at spatial distance 0) and then the spatial ranges is good and common practice when performing manual fitting. In the following, we produce weighted least squares fits when possible and use the manual approach otherwise. For our data, we have stated that our manual variogram fits (with suffix `man` in the following code) appear visually closer to the empirical variogram than the WLS fits (suffix `ls`).

```
vario  <- variogramST(PM10res~1,data=OBS_sel)
sep    <- vgmST("separable",space=vgm(0.9,"Exp",10000,0.1),time=vgm(0.9,"Exp",3.5,0.1),
                sill=40)
sep_ls <- fit.StVariogram(model=sep,object=vario)
sill   <- 250
sep_man<- vgmST("separable",space=vgm(.18,"Exp",1.5e5,0.01,add.to=vgm(.12,"Exp",8e3,.01)),
                time=vgm(80/sill,"Exp",1670,19/sill,add.to=vgm(66/sill,"Exp",85,0)),
                sill=sill)
ps     <- vgmST("productSum",space=vgm(9,"Exp",8e3,1),time=vgm(8,"Exp",106,2),k=2)
ps_man <- vgmST("productSum",space=vgm(12.5,"Exp",2e5,0,add.to=vgm(10,"Exp",9e3,0)),
                time=vgm(35,"Exp",800,9.5,add.to=vgm(36,"Exp",85,0)),k=0.035)
ps_ls  <- fit.StVariogram(model=ps,object=vario)
m_man  <- vgmST("metric",joint=vgm(60,"Exp",8e3,10,
                 add.to=vgm(70,"Exp",1e5,0,stAni=1000)),stAni=150)
sm_man <- vgmST("sumMetric",space=vgm(30,"Exp",20000),
                time=vgm(40,"Exp",70,15),joint=vgm(50,"Exp",1.6e5,0),stAni=1000)
```

We point out that the fit of the metric model is quite poor since the empirical variogram shows

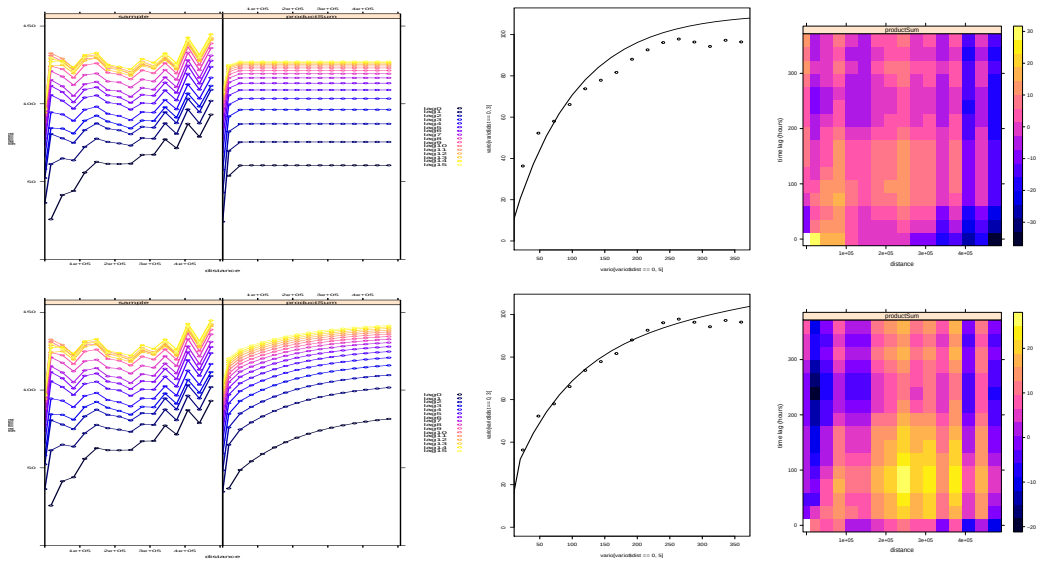

FIGURE 6. *Empirical and fitted variograms for a product-sum model. Upper line: WLS fit with* `fit.StVariogram`*. Lower line: manual fit. Left: empirical vs. fitted variogram for different time lags (*`plot` *arguments* `all=T,map=F`*). Middle: marginal temporal variogram. Right: difference map of space-time variogram values (*`plot` *arguments* `diff=T,map=T`*).*

TABLE 2. *Adjustment scores computed for 6 spatio-temporal variogram models. Sep. = Separable model; ProdSum = Product Sum model. Autofit is the automatic fitting using* `fit.STVariogram`*. Manual indicates manual fit by visual inspection. Metric and SumMetric can only be fitted manually.*

Model	Sep. WLS	Sep. Man.	ProdSum WLS	ProdSum Man.	Metric Man.	SumMetric Man.
No weight	20.56	14.86	8.71	9.94	28.38	11.14
Weighted	66.69	50.73	36.02	35.77	82.37	40.33

that a common sill for each time lag is not a reasonable hypothesis. Various plotting options are available in `gstat`, for instance for confronting empirical and fitted variogram curves for different time lags or for showing maps of differences between empirical and fitted space-time variograms; see the boolean arguments `map`, `all` and `diff` of `gstat`'s plot function for `StVariogram` objects:

```
plot(vario,ps_man,all=T,map=F)
plot(vario,ps_man,map=T,diff=T)
```

Figure 6 shows several plots for WLS and manual fits of the product-sum model, where the quality of the manual fit seems visually more convincing. Finally, to compare the 6 different models fitted in this paragraph, we computed two adjustment scores, namely the mean squared difference and a weighted mean squared difference between the fitted and the empirical variogram, where the weights have been set to be the ratio between the number of pairs and the squared value of the fitted variogram at the corresponding space-time lag. Results are summarized in Table 2 and indicate that the product-sum model yields the best performance.

4.3. Prediction, kriging and cross-validation

4.3.1. Prediction and validation

Assuming that the deterministic part has been removed, either because it is constant or because it has been modeled, we now deal with a zero mean spatio-temporal process Y with a stationary covariance function $C_{ST}(h,u)$. The Best Linear Unbiased Predictor of $Y(s_0,t_0)$ is the so-called Kriging predictor

$$Y^*(s_0,t_0) = \Sigma'_{ST,0}\Sigma_{ST}^{-1}\hat{\mathbf{Y}}$$

where $\hat{\mathbf{Y}} = (\hat{Y}(s_i,t_i))_{i=1,\ldots,n}$ with $\hat{Y}(s_i,t_i) = Z(s_i,t_i) - \hat{\mu}(s_i,t_i)$ the vector of residuals, Σ_{ST} the $n \times n$ matrix with elements $\Sigma_{ST}[ij] = C_{ST}(s_i - s_j, t_i - t_j)$ and $\Sigma_{ST,0}$ the n column vector with elements $\Sigma_{ST,0}[i] = C_{ST}(s_i - s_0, t_i - t_0)$. The prediction variance, also known as the kriging variance, is

$$\begin{aligned}\sigma^2_*(s_0,t_0) &= \mathrm{Var}\{\mathrm{Z}^*(\mathrm{s}_0,\mathrm{t}_0) - \mathrm{Z}(\mathrm{s}_0,\mathrm{t}_0)\} \\ &= C_{ST}(0,0) - \Sigma'_{ST,0}\Sigma_{ST}^{-1}\Sigma_{ST,0}.\end{aligned}$$

These computations require the inversion of the matrix Σ_{ST}, which is an operation of the order of $\mathcal{O}(n^3)$ elementary calculations. When n is large or even when n is moderate, this operation is thus prohibitively long. It is then hardly possible to produce predictions without some form of approximation. Many different approaches were proposed to alleviate the size problem: a) the classical approach of conditioning locally, see e.g. Chilès and Delfiner (2012), b) the fixed rank kriging (Cressie and Johannesson, 2008), c) tapering of the covariance functions (Furrer et al., 2006), d) using a combination of b) and c) (Gneuss et al., 2013; Sang and Huang, 2012) and e) using covariance functions that lead to sparse precision matrices (inverse of the covariance matrix) (Lindgren et al., 2011) as implemented in `R-INLA` (Rue et al., 2009; Blangiardo et al., 2013).

All of the above approaches have their caveats. With a), the shape of the neighborhood must be selected according to the parameters of the covariance, and discontinuities are introduced as the local neighborhood changes. Moreover, the simulated covariance is distorted compared to the desired covariance. Using b) amounts to keep only the low frequency components of the covariance function which can produce overly smoothed realizations. Approach c) is dependent on the range of the taper function compared to the effective range of the covariance function. When the range of the taper function is larger than the effective range, covariance distortion is reduced but so is the sparsity of the tapered covariance matrix. On the contrary, when the range of the taper function is chosen too small, the resulting sparsity is important but so is the distortion of the covariance function. Approach d) is more complex as one has to split the prediction step in two parts corresponding to a long range-short range decomposition. The long range component is estimated by fixed-rank kriging and the short range component is estimated by tapering the covariance of the residuals. Finally, although being extremely efficient in some cases, approach e) has many limitations: first, the Markov property on which it relies has been firmly established only for Matérn covariances in space; second, the smoothness parameter is restricted to a discrete set of values. Third, although well documented and illustrated with spatial data, applications of the SPDE approach for spatio-temporal data based on `R-INLA` are still rare, Blangiardo et al. (2013) and Cameletti et al. (2013) being notable exceptions. The solution adopted for the kriging of the spatio-temporal field will thus depend upon a rather complex combination of size, type and properties of the data set at hand.

4.4. Validation tools

4.4.1. Validation framework

To assess the predictive performances of a model, there exist essentially two frameworks. One can either rely on external validation or internal validation. The former consists in splitting the data set into a learning and a validation subset. The prediction performed at validation locations are compared to the data to compute the prediction scores described below. The latter is based on cross validation (CV) schemes among, where we can cite the leave-one-out scheme and the less expensive K-fold CV. Of the K subsamples, a single subsample is retained as the validation data for testing the model, and the remaining K-1 subsamples are used as training data. The cross-validation process is then repeated K times, with each of the K subsamples used exactly once as the validation data. The leave-one-out scheme can be seen as a n-fold CV. For external validation and K-fold CV, the data sets may be adapted to the spatio-temporal framework: for instance, the same location subset can be taken out at each time step or all locations can be removed for a temporal subset. Cross-validation tools are not provided in the `R` packages dedicated to spatio-temporal statistics. Nonetheless, they can be easily implemented manually.

4.4.2. Prediction scores

Some well-known discrepancy measures are described below:

– Mean square error and Normalized mean square error

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{Z}_i^* - Z_i)^2 \text{ and } \text{NMSE} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{\mathbf{Z}_i^* - Z_i}{\sigma_i^*}\right)^2$$

– Logarithmic score

$$\text{LogS} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{2}\log(2\pi\sigma_i^{2*}) + \frac{1}{2}\left(\frac{\mathbf{Z}_i^* - Z_i}{\sigma_i^*}\right)^2\right)$$

– Continuous Ranked Probability Score

$$\frac{1}{n}\sum_{i=1}^{n}\int_{-\infty}^{\infty}(F_i(y) - \mathbf{1}_{Z_i^* \leq y})dy$$

where $F_i(y) = \mathbb{P}(Z_i \leq y | Z_j, j \neq i)$ and $\mathbf{1}_A$ is the indicator function of event A, equal to 1 if A is verified and equal to 0 otherwise.

RMSE, NMSE and LogS deal with the predicted values, whereas the CRPS deals with predicted distributions under the Gaussian hypothesis. The latter can considered to be robust when the data depart from the Gaussian distribution Gneiting and Raftery (2007).

4.5. Kriging the PM_{10} concentration data

In this section, we use the models estimated in section 4.2.4 on the residuals between the output of *CHIMERE* model and the observations for PM_{10} concentrations. The kriging predictions are computed on the $10km \times 10km$ *CHIMERE* grid. Adding the kriged residuals to *CHIMERE* outputs provides corrected prediction maps. Due to the large number of observations, it is impossible to use the full data set for prediction. We will then have to build spatial and/or temporal neighborhoods to perform the kriging on subsampled data.

4.5.1. Preparing the data

If necessary, we start by reloading previously saved data objects (observation series `OBS_sel`, *CHIMERE* predictions `CHM`, and the complete data subset `data`,`coord.sel`,`times` from the covariance function estimation with `CompRandFld`). We then build the prediction grid as a `SpatialPoints` object, and transform its coordinates to the Lambert 93 format.

```
coord    <- cbind(CHM$lon,CHM$lat)
coordchm <- expand.grid(unique(CHM$lon),unique(CHM$lat))
#  Transform to Lambert 93 coordinate projection for approximately correct spatial
#  geodesic distances
coord.sp <- SpatialPoints(coordchm, proj4string=CRS("+init=epsg:4326"))
coord.sp <- spTransform(coord.sp, CRSobj=CRS("+init=epsg:2154"))
plot(coord.sp@coords/1000, pch=4, cex=.5, asp=1, xlab="(km)", ylab="(km)")
```

4.5.2. Prediction with `CompRandFld`

We first load the fitted spatio-temporal covariance models from the corresponding paragraph of Section 4.2.4 and choose the best one in the CLIC sense.

```
%load("fitsPL.Rdata")
nummod    <- 8
corrmodel <- fitsPL[[nummod]]$corrmodel
param     <- as.vector(fitsPL[[nummod]]$param)
param     <- list(nugget=param[1],power_s=param[2],power_t=param[3],scale_s=param[4] ,
                  scale_t=param[5],sep = param[6], sill= var.est, mean=mean.est)
coords    <- coord.sel
times     <- c(88:90)
dobs      <- data[(nT-lT):nT,]
```

Then we build a subset of the data to perform the prediction. More precisely, we will build the predictions using the last 3 days of data at all observation locations. Note that we identify here the times by their number in the time sequence rather by than the date, for the sake of simplicity.

```
ns <- ncol(data)
nT <- nrow(data)
lT <- 2
times <- c(88:90)
dobs  <- data[(nT-lT):nT,]
```

We now define the prediction locations (the grid built above) and the prediction times. We here choose to predict the PM_{10} residuals at four different times: two retrospective prediction at the ante-penultimate and the last time steps and two days forward.

```
loc <- coord.sp@coords/1000
tt  <- c(88,90:92)
loc_to_pred   <- loc
times_to_pred <- tt
```

We are now ready to perform the (simple) kriging with the function `Kri` of the package `CompRandFld`, specifying prediction locations and times, observation locations and times, the covariance model and its parameters and the observations.

```
pr <- Kri(loc=loc_to_pred,time=times_to_pred,coordx=coords,coordt=times,
          corrmodel=corrmodel, param=param, data=dobs)
```

We can plot the kriging and kriging standard error maps of the PM_{10} residuals using for instance the function `raster` from the package `raster`. The resulting maps are depicted in Fig. 7 at times 91 and 92, that is 04/01/2014 and 04/02/2014. We can finally add the predicted residuals to the *CHIMERE* outputs to get corrected maps and corrected temporal prediction maps using similar R scripts. The resulting maps are depicted in Fig. 8 at dates 91 and 92. The functions from the package `spacetime` could also be used to plot the results, as shown in the supplementary material.

```
europe=readOGR('maps',p4s=NULL,layer='carto_eur')
grx <- unique(CHM$lon)
gry <- unique(CHM$lat)
par(mfrow=c(2,2))
for(i in 3:4){
  ras <-raster(list(x=grx,y=gry,z=matrix(pr$pred[i,],
               nrow=length(grx)),col=tim.colors(64)))
  ras <- disaggregate(ras,fact=c(5,5),method="bilinear")
  ras <- rasterize(europe,ras,mask=T)
  plot(ras,xlab='lon',ylab='lat',xlim = c(-5,10),zlim=c(-20,35),legend.width=3,
       cex.axis = 1.5,axis.args = list(cex.axis = 1.5),cex.lab=1.5,asp=1,
       main = paste("Kriging Time=" , tt[i]),
       legend.args=list(text=expression('     ',paste(PM[10],'
                        (',mu,'g.',m^3,')',sep='')),
       line=1.5,cex=1.5), bty='n', box=F)
  plot(europe[europe$COUNTRY=='France',],xlim=range(grx),ylim=range(gry),add=T)
  ras_sd <- raster(list(x=grx,y=gry,z=matrix(sqrt(pr$varpred[i,]),
                   nrow=length(grx)),col=tim.colors(64)))
  ras_sd <- disaggregate(ras_sd,fact=c(5,5),method="bilinear")
  ras_sd <- rasterize(europe,ras_sd,mask=T)
  plot(ras_sd,xlab='lon',ylab='lat',xlim =c(-5,10),zlim=c(7,11),legend.width=3,
       cex.axis = 1.5,axis.args = list(cex.axis = 1.5),cex.lab=1.5,asp=1,
       main = paste("Kriging Time=" , tt[i]),
       legend.args=list(text=expression('     ',paste(sigma[K],'
                        (',mu,'g.',m^-3,')',sep='')),
       line=1.5,cex=1.5), bty='n', box=F)
plot(europe[europe$COUNTRY=='France',],xlim=range(grx),ylim=range(gry),add=T)
}
```

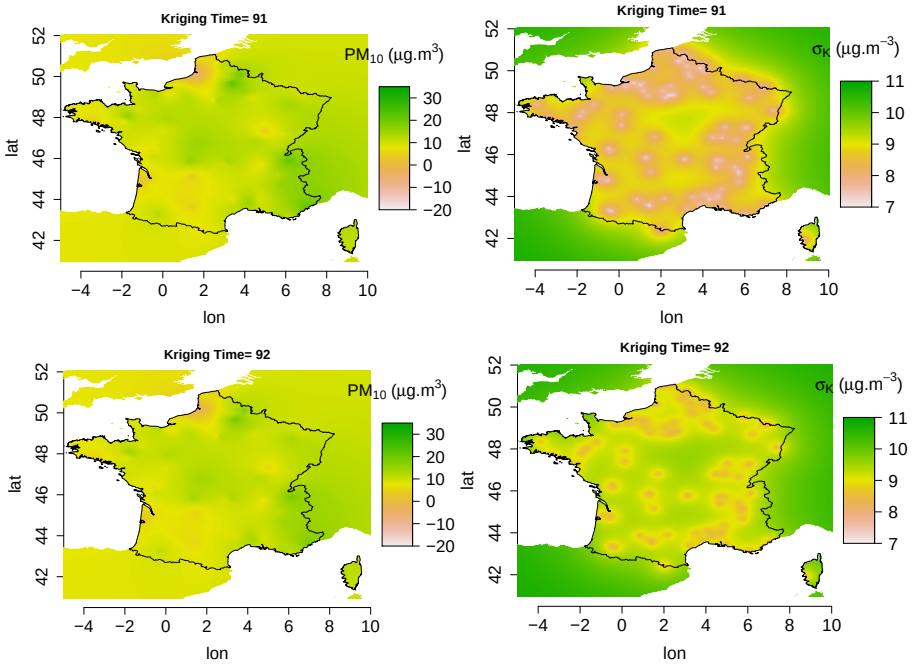

FIGURE 7. *Predicted residuals over France at times 91 and 92 (04/01/2014 and 04/02/2014).*

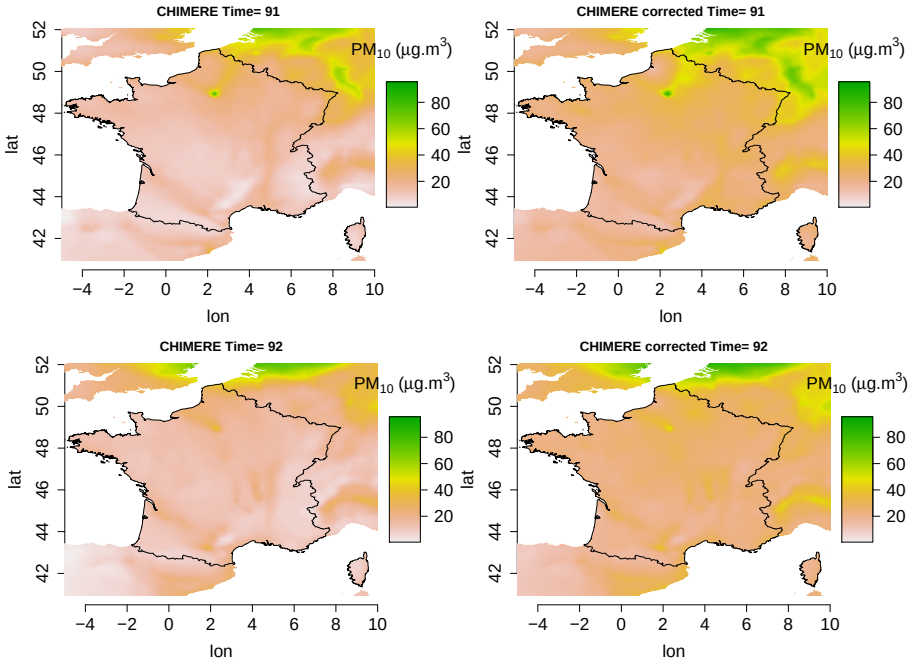

FIGURE 8. *Corrected PM_{10} maps over France at times 91 and 92 (04/01/2014 and 04/02/2014).*

The 8 models fitted by composite likelihood are compared in terms of prediction performances. We computed the root mean squared error (RMSE) between the *CHIMERE* output plus the predicted residual and the actual measure of PM10. We first perform a leave-one-out cross validation at day 90 (30th of March) and then perform a one-day-ahead external validation for the first of April, that is the first date after the learning dataset. In both case we use the data from the two preceding dates at all locations. The results are shown in Table 3. It can be noticed that the model that shows the best prediction performances is model 4, namely the Gneiting model with exponents set to 1 and a nugget effect in both experiments. Nevertheless, all models show similar performances and outperforms the prediction based on *CHIMERE* only for which the RMSE is 14.77. When predicting a-day-ahead, the difference between *CHIMERE* alone and its correction with spatio-temporam kriging is even more striking since for *CHIMERE* alone the RMSE_{EXT} is equal to 19.34.

TABLE 3. *Leave-one-out RMSE and one-day-ahead RMSE_{EXT} computed for 8 estimated spatio-temporal covariance models by leave-one-out cross-validation and external validation.*

Model	1	2	3	4	5	6	7	8
RMSE	10.18	9.68	9.78	9.53	9.71	9.74	9.77	9.75
RMSE_{EXT}	12.85	12.08	12.32	11.97	12.57	12.51	12.51	12.36

4.5.3. Prediction with `gstat`

We first need to reload the data and the fitted spatio-temporal variograms from the corresponding paragraph of Section 4.2.4. It is worth noticing that `gstat` is unable to manage `NA` values with spatio-temporal data. This is surprising since they are accepted for the function `Krige` that performs spatial kriging in the same package. We therefore are restricted to work on the same subsample as in the previous section.

```
load(paste0(DATA,"dataforfits.RData"))
load(paste0(DATA,"timesforfits.Rdata"))
sill          <- 250
separable_fit <- vgmST("separable",space =vgm(.18,"Exp",1.5e5,0.01,
                       add.to=vgm(.12,"Exp",8e3,.01)),
                       time=vgm(80/sill,"Exp",1670,19/sill,
                       add.to=vgm(66/sill,"Exp",85,0)),
                       sill=sill,stAni=50,temporalUnit = "day")
```

Then we build a `STFDF` object containing the data, using the Lambert 93 coordinates defined in the previous paragraph and the prediction grid, which must be a `STF` object. Care has to be taken when building the prediction grid, as an error will occur when running the `krigeST` function if the coordinates and the dates have not the same format as in the `STFDF` object containing the data. We build here the same spatio-temporal prediction grid as in the previous paragraph.

```
temps        <- STFDF_jour@time[times]
data2        <- as.data.frame(as.vector(t(data)))
names(data2) <- 'PM10res'
REScor       <- STFDF(sp=coord.spt,time=temps,data=data2)
tt           <- c(88,90:92)
dates        <- temps <- STFDF_jour@time[tt]
```

```
predGrid     <- STF(coord.spchm,dates)
```

We are now ready to perform ordinary kriging. For simple kriging, the parameter `beta` has to be known and set to the mean value. There are options to define the kriging neighborhood, the default being unique neighborhood. Setting a moving neighborhood requires defining the maximum number of neighboring points `nmax` and setting a spatio-temporal anisotropy `stAni` corresponding to the number of space units equivalent to one time unit. Here we chose to use a unique neighborhood and the product-sum spatio-temporal variogram fitted in Section 4.2.4.

```
krig <- krigeST(PM10res~1,
        data = REScor[,seq(as.Date("2014-03-29"),as.Date("2014-03-31"),by="day")],
        newdata = predGrid, modelList = ProductSum_man)
```

The output of `krigeST` is a `STFDF` object that can be plotted directly using `stplot`. Here, we want to plot the sum of *CHIMERE* and the kriged residuals in the WGS84 coordinate system instead. Therefore, we build another `STFDF` object containing *CHIMERE* outputs for the four considered dates as well as the latter with kriged residuals added.

```
temps   <- STFDF_jour@time[tt]
mydata2 <- as.data.frame(cbind(as.double(CHM_mat[,tt]),
                               as.double(CHM_mat[,tt]+krig@data[[1]])))
names(mydata2) <- c("CHIMERE","CHIMERE corrected")
CHMcorgs       <- STFDF(sp=pts,time=temps,data=mydata2)
```

Finally, we are able to plot the results with `stplot` in the same way as in the previous section. Results are depicted in Figure 9.

As with `CompRandFld`, we can easily implement a cross-validation scheme using the same settings as in the previous paragraph. At least two dates are necessary to create an `STF` object.

```
tt    <- 89:90
dates <- STFDF_jour@time[tt]
times <- c(69,70)
err   <- matrix(NA,ns,6)
for(nummod in 1:6){
  corrmodel <- models[[nummod]]
  for (ista in 1:ns){
    krig <- krigeST(PM10res~1, data = REScor[-ista,times],
                    newdata = STF(coord.spt[ista],dates), modelList = corrmodel)
    err[ista,nummod] <- data[nT,ista]-krig@data[2,]
  }
}
RMSE <- sqrt(apply(err^2,2,mean))
```

The results are summarized in Table 4. The best predictive model is the automatically adjusted product sum model, which has also shown the best performance in terms of adjustment scores (see the corresponding paragraph of Section 4.2.4). It also outperforms some versions of the Gneiting model fitted with `CompRandFld` in the leave-one-out cross-validation schemes but these latter show best one-day-ahead prediction performances overall.

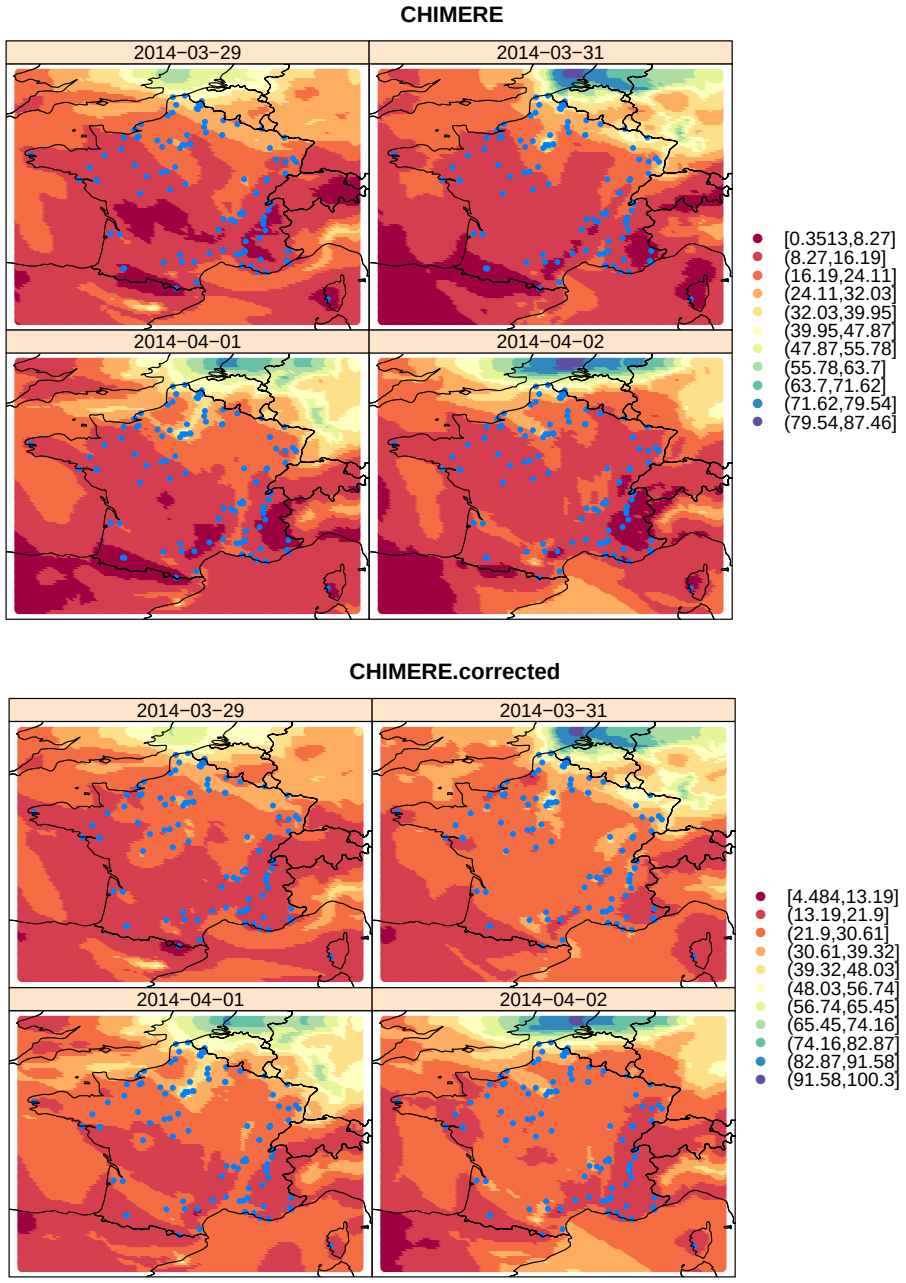

FIGURE 9. CHIMERE *and corrected* PM_{10} *maps over France at four different dates computed with* `gstat`.

5. Discussion

This hands-on review has shown how to perform the analysis of spatio-temporal data, from visual exploration of the data set to the estimation of the spatio-temporal covariance function and to the spatio-temporal prediction with the help of the most complete R packages for spatio-temporal

TABLE 4. *Leav-one-out RMSE and one-day-ahead $RMSE_{ext}$ computed for 6 estimated spatio-temporal covariance models by leave-one-out cross-validation and external validation.. Sep. = Separable model; ProdSum = Product Sum model. Autofit is the automatic fitting using* `fit.STVariogram`*. Manual indicates manual fit by visual inspection. Metric and SumMetric can only be fitted manually*

Model	Sep. Autofit	Sep. Manual	ProdSum Autofit	ProdSum Manual	Metric	SumMetric
RMSE	9.73	9.90	9.62	10.03	9.87	10.43
$RMSE_{EXT}$	12.74	14.35	12.74	13.79	14.70	13.43

data available today. CRAN's "SpatioTemporal" Task View [2] bears witness of the impressive range of tools available for the analysis of space-time data through `R` packages, implemented by a large number of contributors for treating data far beyond the framework of continuous space and continuous Gaussian variables covered in this tutorial. Some of these packages are tightly linked to a specific type of data, or to a specific statistical model. In particular, the packages `SpatioTemporal` (Lindström et al., 2013) and `Stem` (Cameletti, 2009) cover models (2) and (3). The `spate` (Sigrist et al., 2015) package provides functionality for spatio-temporal modeling of large data sets using a spectral SPDE approach.

Our case study on a high-dimensional set of real data has demonstrated that the step from purely spatial to spatio-temporal data analysis is highly nontrivial: the visual exploration of data becomes intricate, and the dependence along the time arrow can be asymmetric and occur through various types (separable or nonseparable covariance function, dynamic model, ...). The huge size of typical data sets calls for efficiently implemented algorithms to stock, manipulate and explore data. Models must be capable to capture the specific characteristics of time-dependent data through suitable space-time covariance structures while remaining amenable to efficient and relatively fast estimation and prediction through kriging. Covariance functions are tied to non-negative definiteness, such that the construction of admissible models and distance functions in the plane or on the sphere for large-region or global data is intricate.

Table 5 summarizes the functionality of the most relevant general purpose `R` packages for estimating space-time covariances. A first observation is that, although data structures and syntax being relatively homogeneous within individual packages, differences across packages are quite profound owing to different package authors with different scientific backgrounds, which is a blessing and a curse at the same time and characteristic for `R`. Therefore, it is often difficult to exchange data and models across packages in a straightforward way, which in practice impedes a simple access to the full range of tools provided by various packages. A second observation is that we could not find any package that assembles all features that are necessary to conduct a real high-dimensional case study. `RandomFields` (Schlather et al., 2016, 2015) is today the most comprehensive package for the analysis and simulation of spatial and spatio-temporal random fields. However, covariance functions are only estimated using weighted least squares and maximum likelihood, thereby limiting its use to data sets of moderate size. In addition, the price to pay for its richness is its complexity and a syntax that may appear unwieldy to some users. `CompRandField` (Padoan et al., 2015) is oriented towards spatio-temporal random fields and proposes most spatio-temporal covariance models, pairwise composite likelihood for their estimation and spatio-temporal tapering for kriging. However, its data structure is rather limited since it can only work with a factorial space×time data design which does not allow

[2] https://cran.r-project.org/web/views/SpatioTemporal.html

any missing values. Moreover, it does not contain any trend modeling functionality. By contrast, `gstat` (Pebesma, 2004) offers a nice specific space-time data structure based on the package `spacetime` (Pebesma, 2012) for handling the data and trend modeling. However, it handles only few spatio-temporal models; in particular, the Gneiting class is missing. Estimation is based only on weighted least squares. A final observation is that (by now) there is no package able to handle multivariate spatio-temporal data.

Modern dynamic and interactive visualization tools like the Shiny/Leaflet combination used in our analysis are an important step towards a more straightforward exploration of space-time data and communication of scientific results. Allowing the use of animated graphics in digital manuscripts of scientific papers is another interesting recent development (Genton et al., 2015).

In summary, we look back at a very enriching experience, which has allowed us to discover a multitude of powerful `R` tools. We are excited to look forward to the many more contributions yet to come from the large number of the developers and package authors that are working on space-time data, whose availability and mass seems to increase at an almost exponential speed thanks to techniques like remote sensing and mobile devices for data collection. If we could wish for specific extensions with respect to available `R` tools, we would be happy to have implementations of a larger choice of covariance functions in `gstat`, more flexible data structures including missing values and trend modeling in `CompRandFld` and efficient maximum pairwise likelihood estimation approaches in `RandomFields`. Air quality agencies usually monitor several pollutants simultaneously. Currently, no `R` package is capable to handle multivariate spatio-temporal data. We thus performed a single variable analysis. Developing a package offering functionality for the analysis and prediction of multivariate spatio-temporal data would therefore be of great interest. In that respect, the recent work in Bourotte et al. (2016), proposing a flexible class for fully non-separable models, offers an interesting starting point.

TABLE 5. *Most important `R` packages for spatio-temporal analysis, along with their main features and limitations.*

Packages	Data.	Cov models	Estimation	Kriging	Big N
`gstat`	`STDF`, `STSDF` or `STIDF` data structure from `spacetime`.	4 classes: *Separable, Product-Sum, Metric, Sum Metric*. Geom. anisotropy	`fit.StVariogram` Weighted Least Sq. only.	`krigeST` Linear model for the trend. No `NA`s.	Moving Neigh only.
`CompRandFld`	Space $\times$ Time data design. No `NA`s at all.	Many classes: *Separable, Product-Sum, Porcu, Gneiting, ...*	Weighted Least Sq. Comp. Lik. Hypothesis testing.	`Kri` No trend model. Tapering. Chordal and Geodesic dist.	Pairwise Comp. Lik., Tapering., Use of `spam`
`RandomFields`	`RFsp`: extension from `sp` package	Comprehensive: *Product-Sum, Gneiting, Porcu, mixtures, ...*	`RFfit` Weighted Least Sq., Max. Lik.	`RFinterpolate` incl. trend modeling	Use of `spam` Tapering

6. Acknowledgments

The data set was kindly provided by INERIS. The "RESSTE Network" is a network of French academics and researchers sharing an interest in developing statistical methods for spatio-temporal data. It is funded by the MIA division of INRA. The following authors contributed to this paper:

Soumis au Journal de la Société Française de Statistique
File: main.tex, compiled with jsfds, version : 2009/12/09
date: September 16, 2016

- Denis Allard, BioSP, INRA, 84914, Avignon, France;
- Maxime Beauchamp, INERIS, Direction des risques chroniques, 60550 Verneuil-en-Halatte, France;
- Liliane Bel, UMR MIA-Paris, AgroParisTech, INRA, Université Paris-Saclay, 75005, Paris, France;
- Nicolas Desassis, MINES ParisTech, PSL Research University, Centre de géosciences, Equipe Géostatistique, 77300 Fontainebleau, France;
- Édith Gabriel, LMA EA2151, Université d'Avignon, 84916 Avignon, France;
- Ghislain Geniaux, Ecodéveloppement, INRA, 84914, Avignon, France;
- Laure Malherbe, INERIS, Direction des risques chroniques, 60550 Verneuil-en-Halatte, France
- Davide Martinetti, Ecodéveloppement, INRA, 84914, Avignon, France;
- Thomas Opitz, BioSP, INRA, 84914, Avignon, France
- Éric Parent, UMR MIA-Paris, AgroParisTech, INRA, Université Paris-Saclay, 75005, Paris, France;
- Thomas Romary, MINES ParisTech, PSL Research University, Centre de géosciences, Equipe Géostatistique, 77300 Fontainebleau, France;
- Nicolas Saby, InfoSol, INRA, 45075 Orléans, France.

References

Blangiardo, M., Cameletti, M., Baio, G., and Rue, H. (2013). Spatial and spatio-temporal models with r-inla. *Spatial and spatio-temporal epidemiology*, 7:39–55.

Bourotte, M., Allard, D., and Porcu, E. (2016). A flexible class of non-separable cross-covariance functions for multivariate space–time data. *Spatial Statistics*.

Burkard, R. K. (1964). Geodesy for the layman. Technical report, DTIC Document.

Cameletti, M. (2009). Stem: Spatio-temporal models in r. *R package version*, 1.

Cameletti, M., Lindgren, F., Simpson, D., and Rue, H. (2013). Spatio-temporal modeling of particulate matter concentration through the spde approach. *AStA Advances in Statistical Analysis*, 97(2):109–131.

Chilès, J. and Delfiner, P. (2012). *Geostatistics: Modeling Spatial Uncertainty, 2nd edition*. John Wiley & Sons.

Cressie, N. (1993). *Statistics for Spatial Data: Wiley Series in Probability and Statistics*. Wiley-Interscience New York.

Cressie, N. and Johannesson, G. (2008). Fixed rank kriging for very large spatial data sets. *J. of the Royal Statist. Society, Series B*, 70:209–226.

Cressie, N. and Wikle, C. K. (2015). *Statistics for spatio-temporal data*. John Wiley & Sons.

Dowle, M., Srinivasan, A., Short, T., with contributions from R Saporta, S. L., and Antonyan, E. (2015). *data.table: Extension of Data.frame*. R package version 1.9.6.

Furrer, R., Genton, M. G., and Nychka, D. (2006). Covariance tapering for interpolation of large spatial datasets. *J. Computnl Graph. Statist.*, pages 502–523.

Genton, M. G., Castruccio, S., Crippa, P., Dutta, S., Huser, R., Sun, Y., and Vettori, S. (2015). Visuanimation in statistics. *Stat*, 4(1):81–96.

Gneiting, T. (2002). Compactly supported correlation functions. *Journal of Multivariate Analysis*, 83(2):493—-508.

Gneiting, T. and Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378.

Gneuss, P., Schmid, W., and Schwarze, R. (2013). Efficient approximation of the spatial covariance function for large datasets - analysis of atmospheric CO_2 concentrations. In *Discussion Paper series recap15*.

Hengl, T., Roudier, P., Beaudette, D., and Pebesma, E. (2015). plotKML: Scientific visualization of spatio-temporal data. *Journal of Statistical Software*, 63(5):1–25.

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. *J. R. Statist. Soc. B*, 73:423–298.

Lindsay, B. G. (1988). Composite likelihood methods. *Contemporary Mathematics*, 80(1):221–239.

Soumis au Journal de la Société Française de Statistique
File: main.tex, compiled with jsfds, version : 2009/12/09
date: September 16, 2016

Lindström, J., Szpiro, A., Sampson, P. D., Bergen, S., and Sheppard, L. (2013). Spatiotemporal: An r package for spatio-temporal modelling of air-pollution. *J stat softw (in press)(http://cran. rproject. org/web/packages/SpatioTemporal/index. html)*.

Menut, L., Bessagnet, B., Khvorostyanov, D., Beekmann, M., Blond, N., Colette, A., Coll, I., Curci, G., Foret, G., Hodzic, A., Mailler, S., Meleux, F., Monge, J.-L., Pison, I., Siour, G., Turquety, S., Valari, M., Vautard, R., and Vivanco, M. G. (2013). Chimere 2013: a model for regional atmospheric composition modelling. *Geoscientific Model Development*, 6(4):981–1028.

Padoan, S. A., Bevilacqua, M., et al. (2015). Analysis of random fields using comprandfld. *Journal of Statistical Software*, 63(i09).

Pebesma, E. (2012). Spacetime: Spatio-temporal data in r. *Journal of Statistical Software*, 51(7):1–30.

Pebesma, E. J. (2004). Multivariable geostatistics in s: the gstat package. *Computers & Geosciences*, 30:683–691.

R Core Team (2013). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Rue, H., Martino, S., and Chopin, N. (2009). Approximate bayesian inference for latent gaussian models by using integrated nested laplace approximations. *Journal of the royal statistical society: Series b (statistical methodology)*, 71(2):319–392.

Ryan, J. A. and Ulrich, J. M. (2014). *xts: eXtensible Time Series*. R package version 0.9-7.

Sang, H. and Huang, J. (2012). A full scale approximation of covariance functions for large spatial data sets. *J. R. Statist. Soc. B*, 74:111–132.

Schlather, M., Malinowski, A., Menck, P. J., Oesting, M., and Strokorb, K. (2015). Analysis, simulation and prediction of multivariate random fields with package RandomFields. *Journal of Statistical Software*, 63(8):1–25.

Schlather, M., Malinowski, A., Oesting, M., Boecker, D., Strokorb, K., Engelke, S., Martini, J., Ballani, F., Moreva, O., Menck, P. J., Gross, S., Ober, U., Christoph Berreth, Burmeister, K., Manitz, J., Morena, O., Ribeiro, P., Singleton, R., Pfaff, B., and R Core Team (2016). *RandomFields: Simulation and Analysis of Random Fields*. R package version 3.1.16.

Sigrist, F., Künsch, H. R., and Stahel, W. A. (2015). spate: An R package for spatio-temporal modeling with a stochastic advection-diffusion process. *Journal of Statistical Software*, 63(14):1–23.

Stein, M. (2013). Statistical properties of covariance tapers. *J. Comput. Graph. Statist.*, 22:866–885.

Varin, C., Reid, N., and Firth, D. (2011). An overview of composite likelihood methods. *Statistica Sinica*, 21(1):5–42.

Wickham, H. (2009). *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York.

Wickham, H. and Francois, R. (2016). *dplyr: A Grammar of Data Manipulation*. R package version 0.5.0.

Soumis au Journal de la Société Française de Statistique
File: main.tex, compiled with jsfds, version : 2009/12/09
date: September 16, 2016