Doble Grado en
Matemáticas y Estadística

TRABAJO FIN DE GRADO

Predicción en datos espaciales y espacio-temporales: Modelos estadísticos e implementación en R

Autor
Montemayor Lago Arenal

Tutora
Ana María MuñozReyes

Sevilla, Junio de 2022

Índice general

Prólogo

El trabajo de fin de grado que se presenta con el título "Predicción en datos espaciales y espacio temporales: Modelos estadísticos e implementación en R" ha sido elaborado durante febrero de 2022 hasta mayo de 2022. Se presenta como parte de los requisitos necesarios para la obtención del doble grado de Matemáticas y Estadística de la Universidad de Sevilla. Ha sido un proceso duradero de investigación y redacción que culmina con la aplicación de muchos de los conocimientos obtenidos durante los años realizados en la facultad de Matemáticas.

Estos años cursando la doble titulación han sido un periodo extenso de aprendizaje que ha formado y moldeado mi persona. He mejorado, profundizado y descubierto no sólo conocimientos técnicos sino personales. Solo guardo buenos recuerdos, ya que de los malos siempre he aprendido y sacado algo positivo.

Agradezco a todas las personas que se han cruzado en mi camino y han dejado parte de su pensamiento y filosofía. Gracias a todo el personal docente por el infinito apoyo y la continua ayuda. Por querer hacernos mejores profesionales y personas. En especial, gracias a Ana María Muñoz Reyes, por ser mi tutora de este proyecto. Gracias también a mis compañeros, por los cuales me he sentido siempre apoyada y arropada ante cualquier circunstancia. Por último, gracias a todas aquellas personas que han creído en mi, aquellos que me han aportado seguridad en los pasos que iba dando y motivación cuando algo parecía que sería difícil.

Muchas gracias.

Resumen

Este trabajo de fin de grado tiene como objetivo recopilar información sobre los procesos de predicción de datos espaciales y espacio temporales fixed rank kriging (FRK) y fixed rank filtering (FRF), ya que no existe información abundante sobre ellos. De esta forma, se quiere dar a conocer estas herramientas de tratamiento de datos, puesto que tienen un enorme potencial analítico.

Este trabajo de fin de grado se divide en tres capítulos. El primero de ellos nos sirve como base teórica para la construcción de los modelos. En él se definen los conceptos necesarios para la correcta comprensión del proyecto, principalmente nociones de probabilidad y procesos estocásticos. Además, se muestra información sobre el tratamiento y almacenamiento de los datos espaciales y espacio temporales, y sobre el Sistema de Información Geográfica.

El segundo capítulo consta del marco teórico. Se definen los modelos SRE y STRE, con los que se trabajarán para elaborar los procedimientos de fixed rank kriging (FRK) y fixed rank filtering (FRF). Con ellos, se mostrará una forma innovadora y eficaz de predecir datos espaciales y espacio temporales.

El tercer capítulo contiene los paquete previos y necesarios para la ejecución del paquete FRK, así como la definición de las diferentes clases y métodos usados en ellos.

El cuarto capítulo se basa en la implementación en R del modelo FRK, ya que para modelo FRF no existe aún librería que lo implemente en R, actualmente se implementa en Matlab. Se trabajará con ejemplos de datos almacenados en paquetes de R. Se llevará a cabo la construcción de los elementos imprescindibles para la estimación y predicción de los datos.

Palabras clave: SRE, STRE, FRK, FRF.

Abstract

The aim of this final degree thesis is to gather information about the spatial and spatiotemporal data prediction processes fixed rank kriging (FRK) and fixed rank filtering (FRF), since there is no abundant information about them. In this way, we want to make these data processing tools known, since they have an enormous analytical potential.

This final thesis is divided into three chapters. The first one serves as a theoretical basis for the construction of the models. We define the concepts necessary for the correct understanding of the project, mainly notions of probability and stochastic processes. In addition, information on the treatment and storage of spatial and spatio-temporal data, and on the Geographic Information System is shown.

The second chapter consists of the theoretical framework. We define the SRE and STRE models. We will work with them to elaborate the fixed rank kriging (FRK) and fixed rank filtering (FRF) procedures. We will show an innovative and efficient way to predict spatial and spatio-temporal data.

The third chapter contains the previous and necessary packages for the execution of the FRK package, as well as the definition of the different classes and methods used in them.

The fourth chapter is based on the R implementation of FRK. Nowdays, there isn't an implementation for FRF model in R. For this purpose, we will work with examples of data stored in R packages. The construction of the essential elements for the estimation and prediction of the data will be carried out.

Keywords: SRE, STRE, FRK, FRF.

Índice de figuras

Capítulo 1

Introducción

1.1. Procesos estocásticos y estacionariedad

El contenido teórico de este capítulo ha sido extraído de Jiménez (2020).

Definición Un proceso estocástico es una familia de variables aleatorias, $\{Y_t, t \in T\}$ ó $\{Y(t), \quad t \in T\}$, definidas sobre un espacio de probabilidad común (Ω, A, P), indexadas por los elementos de un conjunto T. Al conjunto T se le denomina espacio paramétrico o conjunto de índices del proceso.

Una primera clasificación de los procesos estocásticos viene dada por la naturaleza del conjunto T:
Si T es numerable, se dice que es un proceso de parámetro discreto. Ejemplos: $T = \{0, 1, 2, ...\}, \quad T = \{0, \pm 1, \pm 2, ...\}$.
Si T es no numerable, se dice que es un proceso de parámetro continuo. Ejemplos: $T = \{t : -\infty < t < \infty\}$, $T = \{t : t \geq 0\}$, $T = \{t : t \in [0, 1]\}$.

Las variables aleatorias que componen el proceso no son, en general, independientes. Habrá casos en que sí lo sean y otros en los que no.

Definición En el marco estadístico de los procesos estocásticos, una serie temporal $\{Y_1, Y_2, ..., Y_T\}$ se puede interpretar como una realización muestral de un proceso estocástico que se observa únicamente para un número finito de instantes, t = 1, 2, . . . , T, es decir, una serie temporal es una realización parcial de un proceso estocástico de parámetro de tiempo discreto.

Para cada valor del parámetro t, Y_t es una variable aleatoria, por lo que puede calcularse su función de distribución. De esta forma, se puede también calcular la función de distribución conjunta de un vector aleatorio $(Y_{t_1}, Y_{t_2}, ..., Y_{t_n})$.

Así, un proceso estocástico determina un sistema de distribuciones finito dimensionales: $\{F_{t_1}, F_{t_2}, ..., F_{t_n}, t_1, ..., t_n \in T, \quad n \in \mathbb{N}\}$. La implicación contraria ocurre cuando se satisfacen ciertas condiciones.

Teorema de Kolmogorov Se supone que las distribuciones finito dimensionales $\{F_{t_1}, F_{t_2}, ..., F_{t_n}, t_1, ..., t_n \in T, \quad n \in \mathbb{N}\}$ satisfacen

$$\begin{aligned} \lim_{y_n \to \infty} F_{t_1,t_2,...,t_{n-1},t_n}(y_1, y_2, ..., y_{n-1}, y_n) \\ = F_{t_1,t_2,...,t_{n-1}}(y_1, y_2, ..., y_{n-1}), \\ \forall n \in \mathbb{N}, \quad \forall t_1, t_2, ..., t_{n-1}, t_n \in T. \end{aligned} \tag{1.1}$$

Entonces existe un proceso estocástico $\{Y_t, \quad t \in T\}$ tal que sus distribuciones finito dimensionales vienen dadas por (1.1).

Definición Sea $\{Y_t, \quad t \in T\}$ un proceso estocástico tal que todas sus distribuciones finito dimensionales (1.1) son normales multivariantes, entonces se dice que el proceso es un proceso estocástico normal o Gaussiano.

Definición Sea $\{Y(t), \quad t \in T\}$ un proceso estocástico tal que $E(Y_t^2) < \infty, \forall t \in T$. El primer momento de un proceso estocástico viene dado por el conjunto de las medias de todas las variables aleatorias del proceso:

$$E(Y_t) = \mu_t, \quad t \in T.$$

El segundo momento centrado del proceso viene dado por el conjunto de las varianzas de todas las variables aleatorias del proceso y por las covarianzas entre todo par de variables aleatorias:

$$V(Y_t) = E(Y_t - \mu_t)^2 = \sigma_t^2, \quad t \in T,$$
$$cov(Y_t, Y_s) = E(Y_t - \mu_t)(Y_s - \mu_s) = \gamma_{t,s}, \quad \forall t \neq s \in T.$$

La teoría de los procesos estocásticos se utiliza como base para determinar qué procesos modelan los conjuntos de datos con el fin de caracterizar el comportamiento de éstos y servir como base para predecir. La filosofía de la predicción se basa en aprender de las regularidades del comportamiento pasado del modelo y proyectarlo hacia el futuro. Una característica que se le impone a los procesos estocásticos para que sean estables para predecir es la estacionariedad. Se definen dos tipos de estacionariedad:

Definición Estacionariedad estricta. Se dice que un proceso estocástico $\{Y_t, \quad t \in T\}$ es estacionario en sentido estricto sí y sólo si:

$$F_{t_1,t_2,...,t_n} = F_{t_{1+k},t_{2+k},...,t_{n+k}}, \quad \forall t_1, t_2, ..., t_n \in T, \quad \forall k,$$

es decir, si la función de distribución de cualquier conjunto finito de n variables aleatorias del proceso no varía si se desplaza k periodos en el tiempo.

Definición Estacionariedad en covarianza. Sea $\{Y_t, \quad t \in T\}$ un proceso estocástico tal que $E(Y_t^2) < \infty, \quad \forall t \in T$. Se dice que $\{Y_t, \quad t \in T\}$ es estacionario en covarianza, estacionario débil, estacionario de segundo orden o simplemente estacionario sí y sólo si:

1. Todas las variables aleatorias del proceso tienen igual media y es finita:

$$E(Y_t) = \mu < \infty, \quad \forall t.$$

A esta propiedad se le denomina estacionario en media.

2. Todas las variables aleatorias del proceso tienen la misma varianza y es finita:

$$V(Y_t) = E(Y_t - \mu)^2 = \sigma_Y^2 < \infty, \quad \forall t$$

   Es decir, la dispersión en torno a la media (constante) a lo largo del tiempo es la misma para todas las variables del proceso.

3. Las autocovarianzas no dependen del tiempo, solo dependen del número de periodos de separación entre las variables. Es decir, dos variables separadas por k periodos de tiempo poseen igual covarianza lineal que cualesquiera otras dos variables que distan k periodos de tiempo, independientemente del momento concreto de tiempo:

$$cov(Y_t, Y_{t+k}) = E(Y_t - \mu)(Y_{t+k} - \mu) = \gamma_k < \infty, \quad \forall t$$

Propiedad Si un proceso estocástico es estacionario en covarianza y su distribución es normal, entonces es estacionario en sentido estricto.

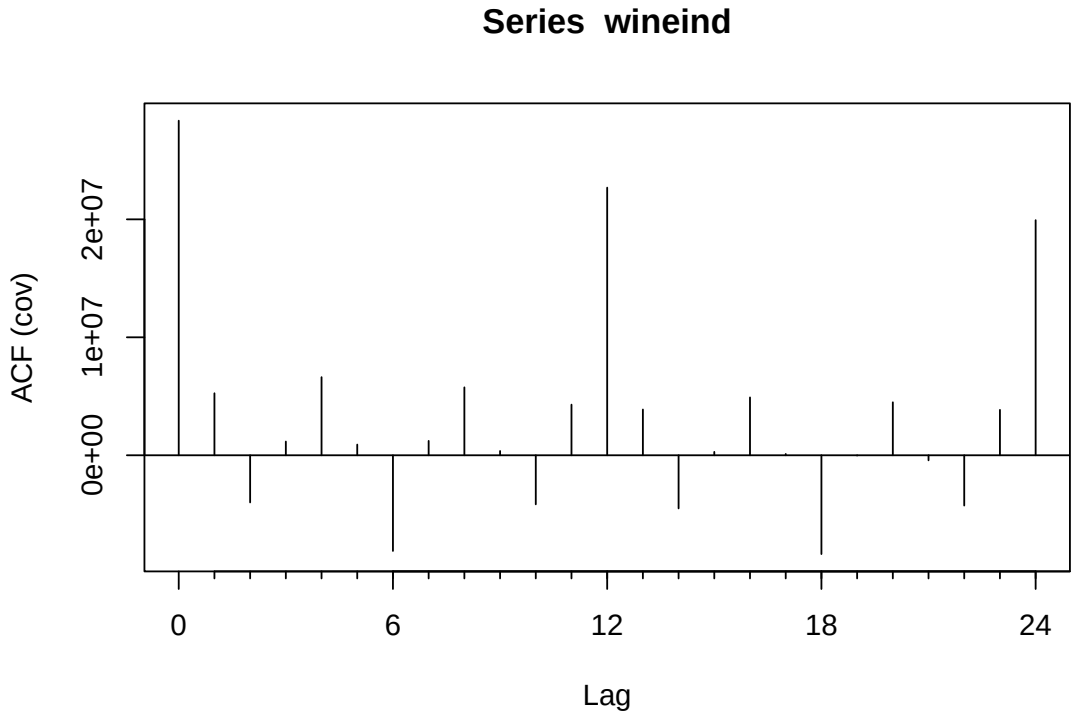
Sea $\{Y_t, \quad t \in T\}$ un proceso estocástico estacionario. Se definen a continuación la función de autocovarianzas y de autocorrelación.

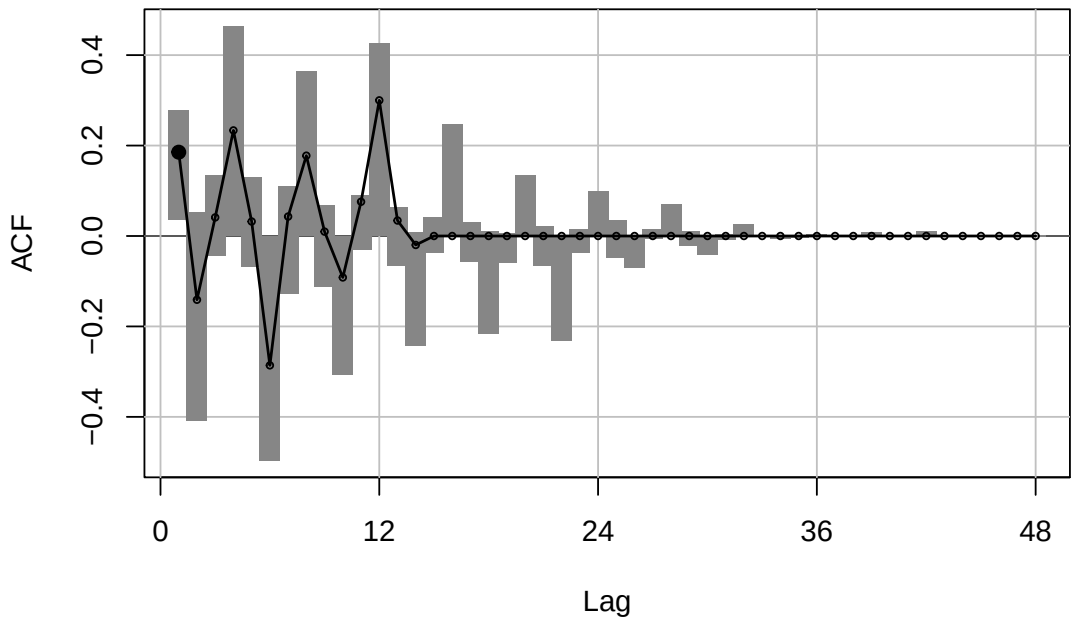
Definición *Función de autocovarianzas (FACV)* La función de autocovarianzas de un proceso estocástico estacionario es una función del parámetro k (retardo), que mide el número de periodos de separación entre las variables:

$$\gamma_k, \quad k = 0, 1, 2...$$

Se puede comprobar que la FACV es una función simétrica: $\gamma_k = E(Y_t - \mu)(Y_{t+k} - \mu) = E(Y_t - \mu)(Y_{t-k} - \mu) = \gamma_{-\mu}$ y además para k=0 se tiene: $\gamma_0 = E(Y_t - \mu)E(Y_t - \mu) = V(Y_t)$.

A continuación se muestran ejemplos de una FACV:





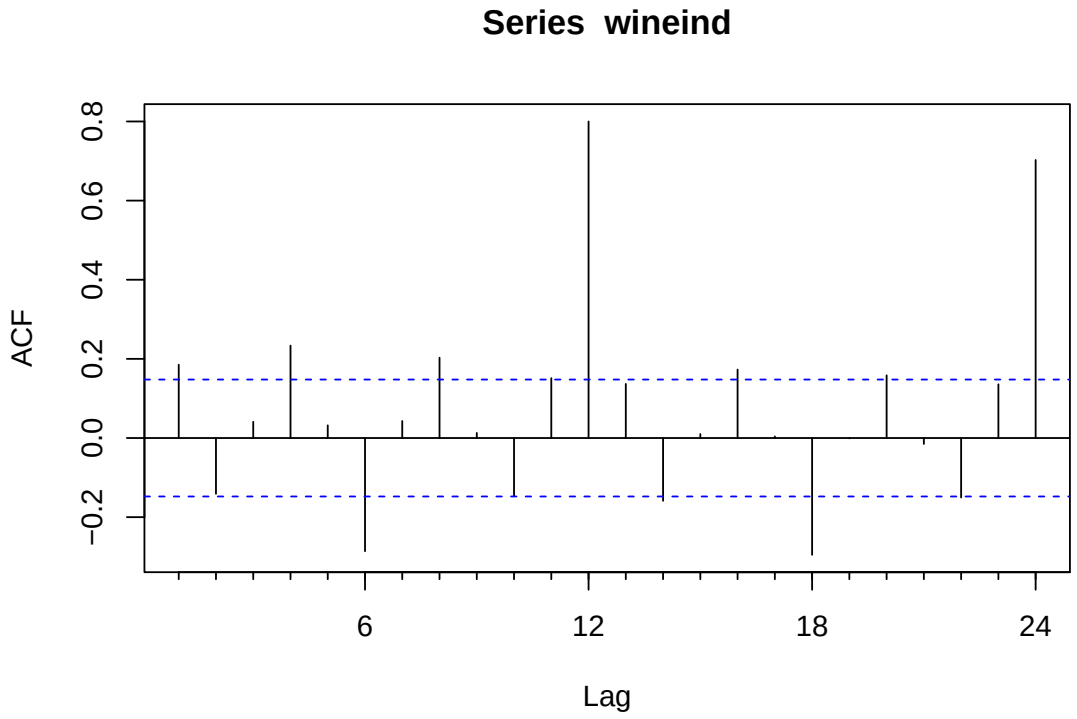
Definición El coeficiente de autocorrelación de orden k de un proceso estocástico estacionario mide el grado de asociación lineal existente entre dos variables aleatorias del proceso separadas por k periodos:

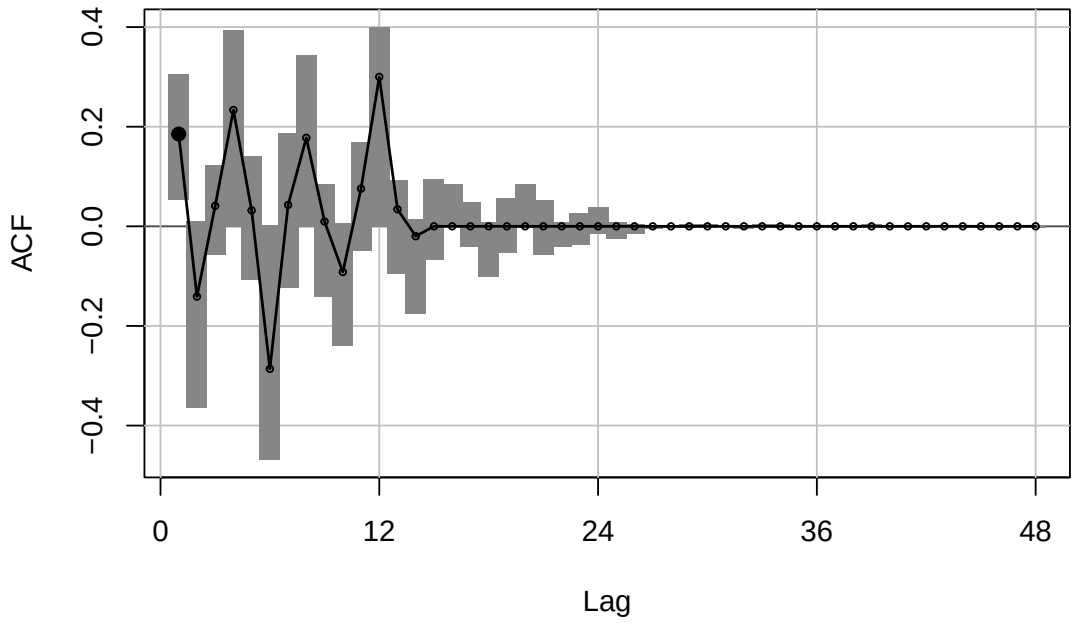
$$\rho_k = \frac{cov(Y_t, Y_{t+k})}{\sqrt{V(Y_t V(Y_{t+k}))}} = \frac{\gamma_k}{\sqrt{\gamma_0 \gamma_0}} = \frac{\gamma_k}{\gamma_0}$$

tal que $|\rho_k| \leq 1, \quad \forall k$ y por ser coeficiente de correlación no depende de unidades.

Definición *Función de autocorrelación (FAC)* La función de autocorrelación de un proceso estocástico estacionario es una función del parámetro k que agrupa los coeficientes de autocorrelación del proceso: $\rho_k, \quad k = 0, 1, 2, ...$ Se representa mediante un correlograma (diagrama de barras).

Véase un ejemplo de FAC:

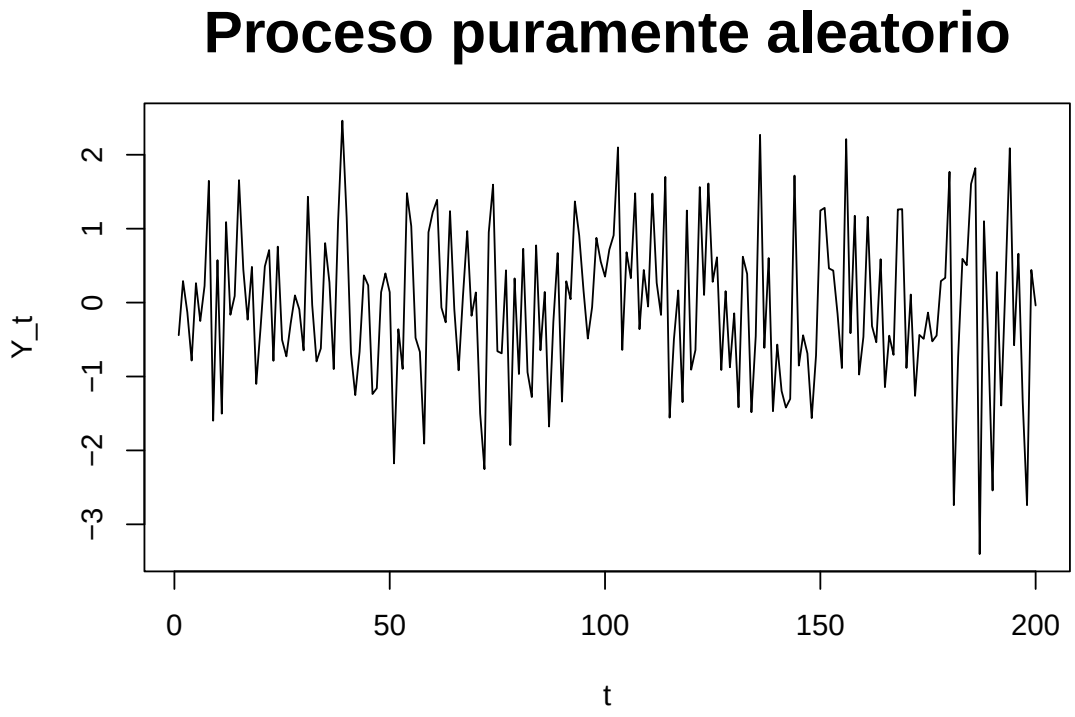




Definición *Proceso puramente aleatorio* Se llama proceso puramente aleatorio a un proceso estocástico formado por variables aleatorias independientes e idénticamente distribuidas. Es fácil comprobar que un proceso puramente aleatorio se corresponde con un proceso estacionario con función de autocovarianzas $\gamma_0 = V(Y_1)$ y $\gamma_k = 0, \quad \forall k \geq 1$.

Véase un ejemplo:

```
y=rnorm(200)
plot(y, type="l", xlab="t", ylab="Y_t",
     main="Proceso puramente aleatorio", cex.main=2)
```



Definición *Paseo aleatorio* Se llama paseo aleatorio de media cero a un proceso estocástico $\{S_t : \quad t = 0, 1, 2, ...\}$ definido por $S_0 = 0$ y $S_t = Y_1 + ... + Y_t, \quad t = 1, 2, ...$ donde las variables Y_t constituyen un proceso puramente estocástico con media 0. Si la distribución de cada Y_t cumple $P(Y_t = 1) = P(Y_t = -1) = \frac{1}{2}$ entonces el proceso se conoce como un paseo aleatorio simple simétrico.

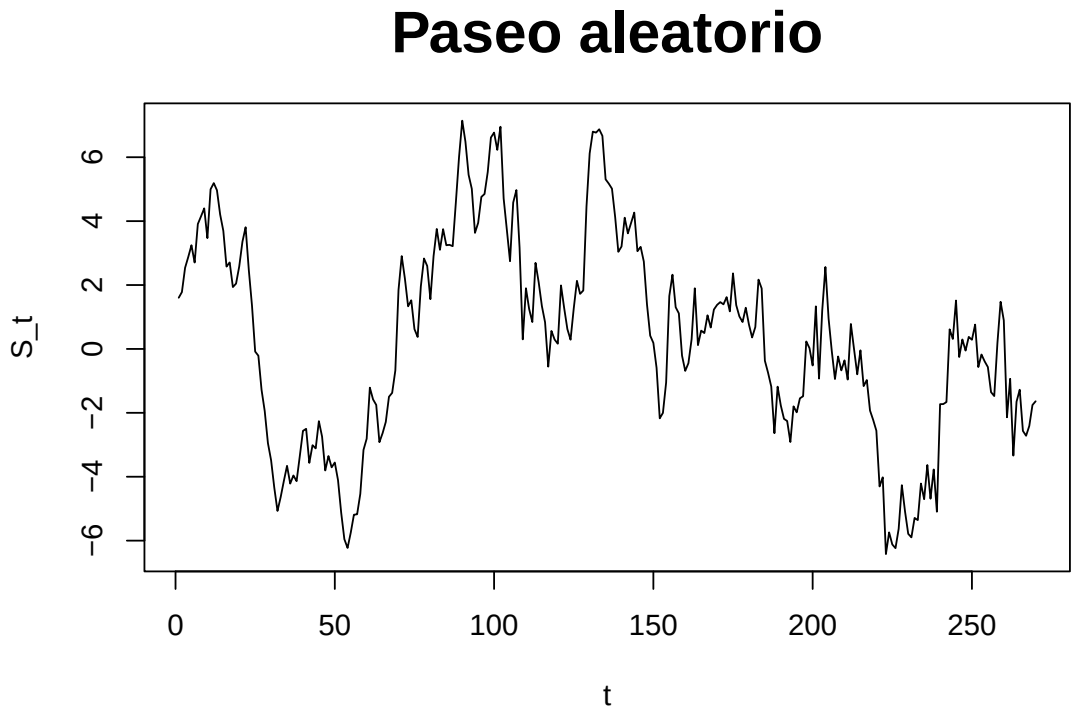
Para este proceso se tiene:

$$E(S_t) = E(Y_1 + ... + Y_t) = 0,$$

$$Cov(S_t, S_{t+k}) = Cov(S_t, S_t + Y_{t+1} + ... + Y_{t+k}) = Cov(S_t, S_t) = tV(Y_1)$$

Como $Cov(S_t, S_{t+k})$ depende de t, se trata de un proceso no estacionario.

```
x = rnorm(270)
s = cumsum(x)
plot(s, type="l", xlab="t", ylab="S_t", main="Paseo aleatorio", cex.main=2)
```

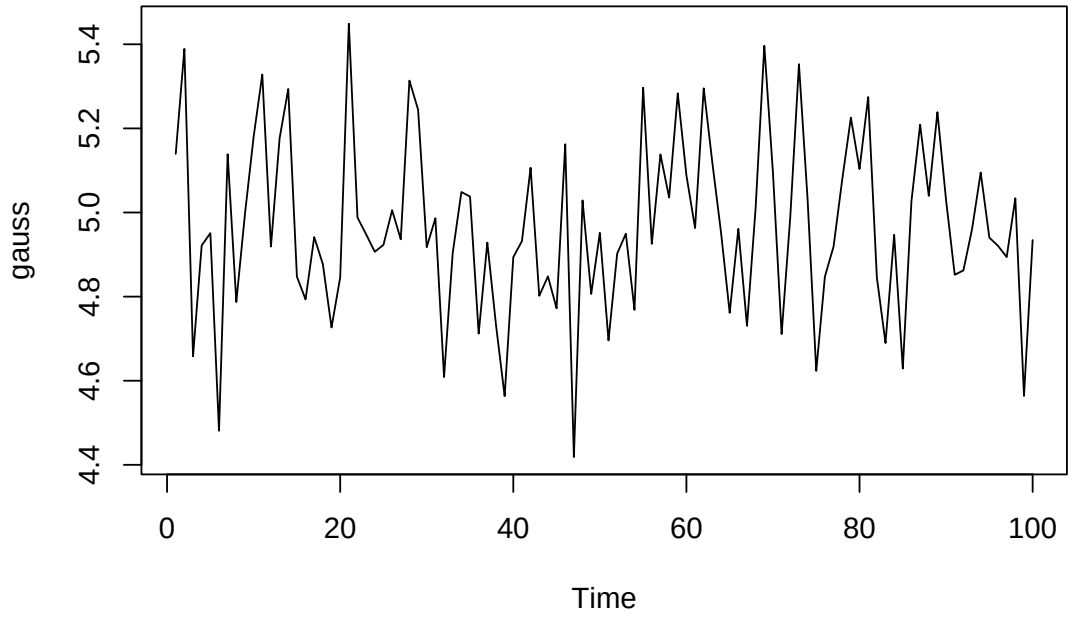


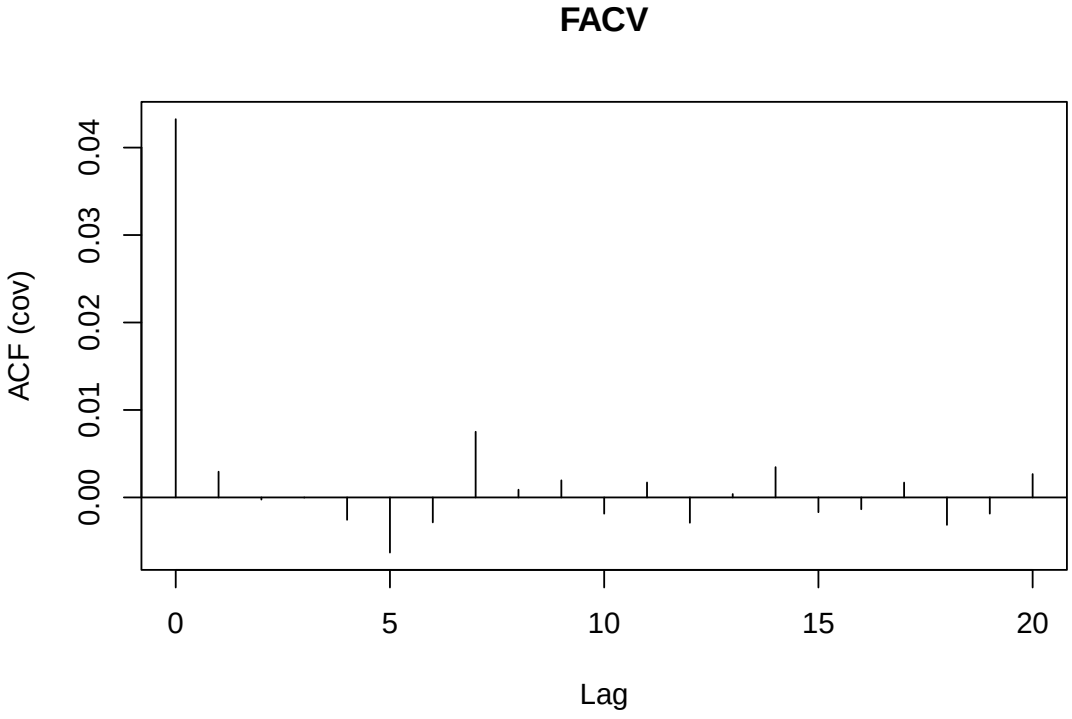
Definición El ruido blanco es un proceso estocástico estacionario α_t, $t = 0, \pm 1, \pm 2, ...$ con media cero $E(\alpha_t) = 0, \quad \forall t$, varianza constante $V(\alpha_t) = \sigma^2, \quad \forall t$ y covarianzas nulas $Cov(\alpha_t, \alpha_s) = 0, \quad \forall t \neq s$. Se representa de la siguiente forma: $\alpha_t \sim RB(0, \sigma^2)$, siendo estacionario si la varianza es finita y su función de autocovarianzas y autocorrelación son de la forma:

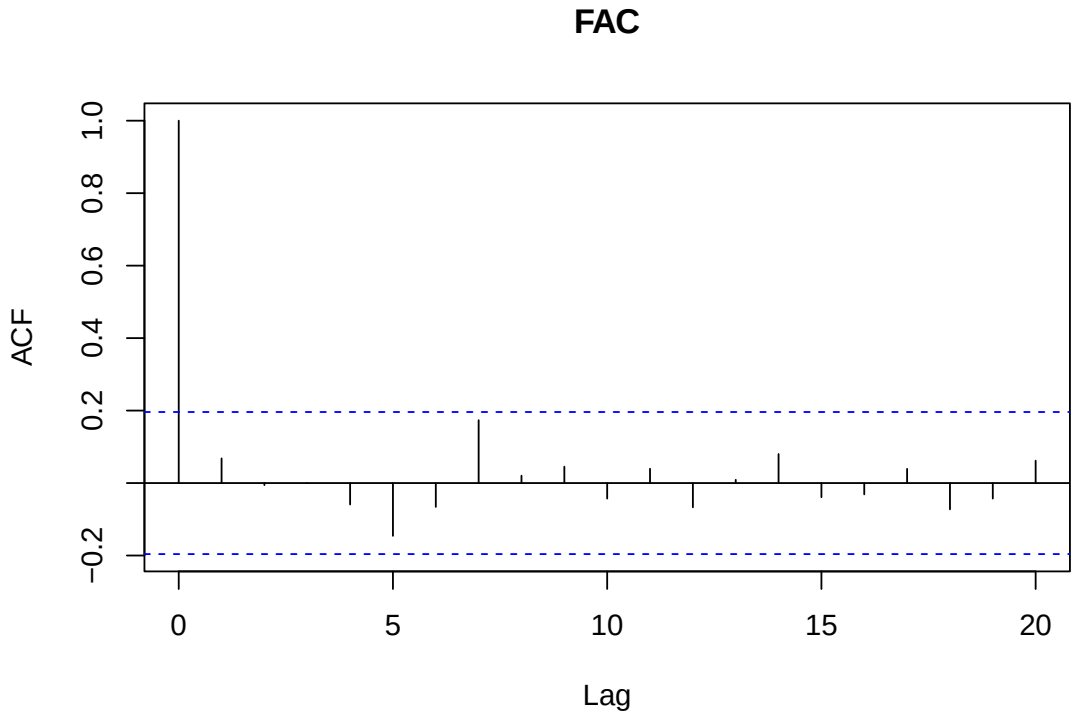
$$\gamma_k = \begin{cases} \sigma^2 & si \quad k = 0 \\ \\ 0 & si \quad k \geq 1 \end{cases}$$

$$\rho_k = \begin{cases} 1 & si \quad k = 0 \\ \\ 0 & si \quad k \geq 1 \end{cases}$$

A continuación se representa un ruido blanco Gaussiano.


gauss
4.4
4.6
4.8
5.0
5.2
5.4
0
20
40
60
80
100
Time


FACV
ACF (cov)
0.00
0.01
0.02
0.03
0.04
0
5
10
15
20
Lag



Definición Un proceso estocástico es un proceso lineal si puede ser representado mediante una sucesión de constantes reales $\{\psi_j\}_{t\in\mathbb{Z}}$ absolutamente sumable, es decir, $\sum_{j=-\infty}^{\infty}|\psi_j|<\infty$ y un proceso ruido blanco $\{Z_t\}_{t\in\mathbb{Z}}$ de tal forma que:

$$Y_t=\sum_{j=-\infty}^{\infty}\psi_j Z_{t-j}$$

Definición El modelo autoregresivo de primer orden AR(1) es un proceso estocástico donde suponemos que el momento presente depende sólo directamente del momento pasado más cercano.
Sea $\{Y_t, \quad t\in Z\}$ un proceso estocástico que satisface las ecuaciones:

$$Y_t=\phi Y_{t-1}+Z_t, \quad t=0,\pm1,\pm2,...$$

donde $\{Z_t, \quad t\in Z\}$ es un proceso de ruido blanco, también llamado vector de innovación, y $|\phi|<1$.
Este proceso es llamado proceso autorregresivo de orden 1 y lo denominaremos AR(1).
Es fácil ver, aplicando reiteradamente la ecuación del modelo, que

$$Y_t=\sum_{j=0}^{\infty}|\phi^j|Z_{t-j},$$

$$\sum_{j=0}^{\infty}|\phi^j|,$$

ya que $|\phi| < 1$, por lo que se trata de un proceso lineal.

Definición Una función o proceso determinista es una función cuyos valores futuros están determinados de manera funcional por valores pasados. Es decir, se puede predecir sin ningún tipo de error.

Proposición Sea $\{Y_t\}_{t\in(Z)}$ es un proceso lineal, esto es,

$$Y_t = \sum_{j=-\infty}^{\infty} \phi_j Z_{t-j}, \quad \forall t \in (Z),$$

donde $\{Z_t\}_{t\in(Z)}$ es un proceso de ruido blanco y $\{\phi_t\}_{t\in(Z)}$ es una sucesión de constantes reales absolutamente sumable, es decir, $\sum_{j=-\infty}^{\infty} |\phi_j| < \infty$. Entonces $\{Y_t\}_{t\in(Z)}$ es un proceso estacionario con:

1. $$E(Y_t) = 0, \quad \forall t \in (Z)$$

2. $$V(Y_t) = \sigma^2 \sum_{j\in(Z)} \phi_j^2 < \infty, \quad \forall t \in (Z)$$

3. $$Cov(Y_t, Y_{t+k}) = \sigma^2 \sum_{j\in(Z)} \phi_j \phi_{j+k} < \infty, \quad \forall t \in (Z).$$

A continuación se mostrará un resultado que ayudará en el desarrollo de los modelos posteriores.
Con este resultado, se podrá afirmar que todo proceso estacionario se corresponde con un proceso lineal o se puede transformar en un proceso lineal si se le resta una componente determinista.

Teorema de descomposición de Wold
Sea $\{Y_t = \sum_{j=0}^{\infty} \Psi_j Z_{t-j} + V_t\}$ un proceso estacionario que no es determinista.
Se tiene:

1. $\Psi_0 = 1$ y $\sum_{j=0}^{\infty} \Psi_j^2 < \infty$.
2. $\{Z_t\}$ es un proceso ruido blanco.
3. $Cov(Z_s, V_t) = 0, \quad \forall s, t.$
4. $\{Z_t\}$ se corresponde con el límite de combinaciones lineales de $\{Y_s\}, \quad s \leq t.$
5. $\{V_t\}$ es un proceso determinista.

1.2. Clasificación datos espaciales

Los datos espaciales se definen como ubicaciones en el espacio. Pero es necesario resaltar que pueden llevar asociados atributos que contengan información.
Por ejemplo, el caudal de los ríos de España. La ubicación de los ríos de España son datos espaciales que llevan asociados el atributo caudal.

Un primera clasificación de los datos espaciales podría ser la siguiente, basándose en su posible representación en el espacio:

1. **Datos vectoriales**
   Los datos vectoriales se presentan por pares de coordenadas relacionadas con algún sistema de referencia, como por ejemplo un sistema cartográfico.
   Ejemplos: ubicaciones de edificaciones o muertes por población.
   Los datos vectoriales se pueden representar en tres tipos de estructuras geométricas:

   a) Puntos
   Los puntos son representaciones del espacio de manera que cada coordenada (x, y) corresponde con una ubicación del espacio.
   Ejemplos: edificio o pozo.

   b) Líneas
   Las líneas poseen dos puntos que definen su estructura. En el caso de que estos dos puntos correspondan con el inicio y fin de la línea, estaríamos tratando con una recta.
   Ejemplo: carretera.

   c) Polígonos
   Los polígonos pueden verse como el área que encierra un conjunto de líneas que comparten puntos de inicio y fin. En este caso los polígonos representan estructuras bidimensionales.
   Ejemplos: lago o país.

2. **Datos rasterizados**
   Los datos raster representan la información a través de píxeles. En este caso se presentan los datos en una distribución de malla o cuadrícula, de tal forma que cada celda corresponde con un atributo.
   Con ellos representamos objetos Ejemplos: imágenes de satélites u otros sensores.

Ahora bien, también se podría clasificar los datos espaciales según el dominio donde se estén midiendo dichos datos.

1. **Datos geoestadísticos**
   Los datos geoestadísticos se corresponde con aquellos datos espaciales que medimos en un dominio D fijo.
   Ejemplo: Temperatura media en Andalucía el mes de julio de 2020.

2. **Datos de área**
   Los datos de área se corresponden con datos espaciales donde el dominio D está dividido en unidades de área, de forma que su suma sea el total D.
   Ejemplo: Riesgos de mortalidad por accidente de tráfico en las Comunidades Autónomas de España durante el periodo 2000-2010.

3. **Datos de procesos puntuales**
   Los datos de procesos puntuales son datos espaciales donde el dominio D es una variable aleatoria. El proceso puntual espacial está formado por el conjunto de índices, que proporcionan las localizaciones de los eventos aleatorios.
   Ejemplo: Posiciones en el cielo de 1000 galaxias.

1.3. Datos espacio temporales

En las últimas décadas, ante la necesidad de analizar diversos procesos de interés en el tiempo, se ha producido un avance en el estudio de modelos que permitan procesar y obtener información de datos espacio temporales. Con esta visión, no sólo se estudiará la componente espacial que indica la localización en un conjunto de puntos, sino además se almacenará dicha componente a lo largo del tiempo.

Uno de los objetivos primordiales en el estudio de datos espacio temporales es el almacenamiento y representación de las observaciones. En esta sección, se abordarán los formatos de almacenamiento más comunes en el uso de las librerías que se verán posteriormente y las diferentes disposiciones de los datos espacio temporales.

1.3.1. Formato de almacenamiento

Los datos espacio temporales generalmente se almacenan en tablas por razones de simplificación, ya que resulta más fácil y sencillo visualizarlos, pudiéndose almacenar además en una sola tabla o en diferentes tablas. Existen tres formas de almacenar los datos espacio temporales en tablas, según Pebesma (2012).

- Formato ancho de tiempo En este formato se almacena en las distintas columnas los momentos del tiempo que estemos midiendo. De esta forma, cada fila corresponde con una localización en la que se mide la variable en momentos de tiempo distintos.
- Formato ancho de espacio En este caso las columnas expresan cada localización y cada fila corresponde a un momento temporal concreto.

```
##   year month day   RPT   VAL   ROS   KIL   SHA  BIR   DUB   CLA   MUL   CLO
## 1   61     1   1 15.04 14.96 13.17  9.29 13.96 9.87 13.67 10.25 10.83 12.58
## 2   61     1   2 14.71 16.88 10.83  6.50 12.62 7.67 11.50 10.04  9.79  9.67
## 3   61     1   3 18.50 16.88 12.33 10.13 11.17 6.17 11.25  8.04  8.50  7.67
## 4   61     1   4 10.58  6.63 11.75  4.58  4.54 2.88  8.63  1.79  5.83  5.88
## 5   61     1   5 13.33 13.25 11.42  6.17 10.71 8.21 11.92  6.54 10.92 10.34
## 6   61     1   6 13.21  8.12  9.96  6.67  5.37 4.50 10.67  4.42  7.17  7.50
##     BEL   MAL
## 1 18.50 15.04
## 2 17.54 13.83
## 3 12.75 12.71
## 4  5.46 10.88
## 5 12.92 11.83
## 6  8.12 13.17
```

Como se pueden observar las localizaciones se encuentran en las columnas, de forma que cada una contiene el dato para esa localización siendo cada fila un periodo de tiempo concreto.

- Formato alargado La información espacio temporal está almacenada en una sola columna, y existen otras columnas que especifican el momento temporal y la localización.

```
##     state year region     pcap     hwy   water    util       pc   gsp
## 1 ALABAMA 1970      6 15032.67 7325.80 1655.68 6051.20 35793.80 28418
## 2 ALABAMA 1971      6 15501.94 7525.94 1721.02 6254.98 37299.91 29375
## 3 ALABAMA 1972      6 15972.41 7765.42 1764.75 6442.23 38670.30 31303
## 4 ALABAMA 1973      6 16406.26 7907.66 1742.41 6756.19 40084.01 33430
## 5 ALABAMA 1974      6 16762.67 8025.52 1734.85 7002.29 42057.31 33749
```

En este último caso, cada fila se corresponde con un momento temporal concreto y una localización concreta.

1.3.2. Disposiciones de los datos espacio temporales gráficamente

Se observa como una unidad espacial que se llamará característica espacial puede distribuirse en un gráfico de diferentes formas, es decir una característica espacial puede tener más de un registro almacenado en diferentes momentos temporales.

Estas características espaciales pueden representarse como puntos, polígonos, líneas o cuadrículas, como ya se ha mencionado en las secciones anteriores. A continuación se definen las posibles opciones, según Pebesma (2021):

- **Cuadrícula completa**
  En este caso las características espaciales $s_i : \quad s = 1, ..., n$ son todas observadas en momentos de tiempo $t_j : \quad j = 1, ..., m$. Se obtiene una malla de tamaño $n \times m$ donde aparecen todas las opciones rellenas con datos.
  Si se fija en una característica espacial concreta, existe una secuencia de momentos temporales con datos. Y ocurre igual a la inversa, si nos fijamos en un momento concreto existen datos de las características espaciales.
  En el caso de que no existiera algún dato simplemente se almacena un valor NA.
  Ejemplos claros de esta forma de almacenamiento podrían ser las imagénes de arcoiris.

- **Cuadrícula dispersa**
  En la cuadrícula dispersa sólo se almacenan las características espaciales y los momentos temporales para los que existen datos.
  Para cada observación z_k existe un índice $\{i, j\}$ donde i se corresponde con la característica espacial y j con el momento temporal.
  Esta forma de almacenamiento es beneficiosa en situaciones como:

  - Conjuntos que contengan muchos valores perdidos.
  - Cuando para un conjunto de características espaciales limitadas se poseen distintos momentos temporales.
  - Cuando para un conjunto de momentos limitados el conjunto de características espaciales varía. Ejemplos de conjuntos de datos que utilicen este tipo de almacenamiento podrían ser la localización de crímenes producidos en un año en una región.

- **Datos no estructurados**
  Cuando los datos que almacenamos no tienen una organización clara con respecto a los momentos temporales y las características espaciales, se almacenan en un formato alargado como vimos con el caso de las tablas alargadas. Podríamos imaginar que se trata de una cuadrícula dispersa donde los ejes se corresponden con los datos sin ninguna limitación.
  Cualquier conjunto de datos se puede almacenar de esta forma, pero tiene claros inconvenientes:

  - Cada dato posee una característica espacial y un momento temporal que deben ser almacenados, y por tanto al no existir una organización se puede llegar a tener un conjunto bastante redundante.
  - Como no existe regularidad, debe proporcionarse de otra forma, y eso conlleva dificultades.
  - La elección de las características espaciales y los momentos temporales será ineficiente. Ejemplo en el que se usaría este almacenamiento sería para casos de enfermedades en una región.

- **Intervalos de tiempo**
  Se pueden clasificar los intervalos de tiempos en tres casos:

  - El tiempo es un instante concreto y la característica espacial existe en ese único instante.
    Ejemplo: un terremoto.
  - El tiempo es un intervalo y la característica espacial no cambia durante todo el intervalo. De esta forma se representa la característica espacial durante ese intervalo.
    Ejemplo: temperatura media de un país.
  - El tiempo es un instante y la característica espacial sí se mueve a través de una trayectoria.
    Ejemplo: movimiento/trayectoria humano.

1.4. Sistema de referencia de coordenadas

El sistema de referencia de coordenadas (CRS) identifica los valores de las coordenadas con los puntos exactos en la superficie de la Tierra o del espacio.

Este sistema es imprescindible cuando trabajamos con datos espaciales, ya que permite unificar los estudios además de aportar veracidad y realidad.

En general existen dos tipos de sistemas de referencia de coordenadas:

- **Geodésico**
  Este CRS se basa en las coordenadas de las tres dimensiones del espacio. Toma como aproximación de la Tierra una elipsoide (global o local), ya que el globo terrestre no es una esfera perfecta.
  Este elipsoide junto con información sobre las coordenadas define el denominado datum.
  Generalmente se asume que se trabaja sobre la superficie terrestre, por lo que para las coordenadas se consideran latitud y longitud.
  Uno de los CRS más usados es el WGS84 (World Geodetic System 1984) en el que se basa el Sistema de Posicionamiento Global (GPS).

- **Proyectado o cartesiano**
  Este CRS se basa en coordenadas dimensionales, tomando como referencia distancias o áreas.
  Un ejemplo claro sería el UTM (Universal Transverse Mercator) que divide la tierra en 60 tramos de longitud numerados y 20 bandas de latitud marcadas con letras. De esta forma, Galicia por ejemplo se encuentra en la cuadrícula 29T.

Para asignar proyecciones a un objeto, ya sea de clase vectorial o ráster, utilizamos la función CRS() del paquete sp, que describiremos más adelante.

```
## [1] "CRS"
## attr(,"package")
## [1] "sp"
```

En el caso de querer cambiar una proyección ya asignada a un objeto, se utiliza la función spTransform del paquete sp.
Y si se quisiera conocer el sistema de referencia de coordenadas que posee un objeto se utilizaría la función st_crs.

```
## Coordinate Reference System:
##   User input: +proj=longlat +ellps=sphere +no_defs
##   wkt:
## GEOGCRS["unknown",
##     DATUM["Unknown based on Normal Sphere (r=6370997) ellipsoid",
##         ELLIPSOID["Normal Sphere (r=6370997)",6370997,0,
##             LENGTHUNIT["metre",1,
##                 ID["EPSG",9001]]]],
##     PRIMEM["Greenwich",0,
##         ANGLEUNIT["degree",0.0174532925199433],
##         ID["EPSG",8901]],
##     CS[ellipsoidal,2],
##         AXIS["longitude",east,
##             ORDER[1],
##             ANGLEUNIT["degree",0.0174532925199433,
##                 ID["EPSG",9122]]],
##         AXIS["latitude",north,
##             ORDER[2],
```

```
##             ANGLEUNIT["degree",0.0174532925199433,
##                 ID["EPSG",9122]]]]

## Coordinate Reference System:
##   User input: +proj=robin +lon_0=0 +x_0=0 +y_0=0 +datum=WGS84 +units=m +no_defs
##   wkt:
## PROJCRS["unknown",
##     BASEGEOGCRS["unknown",
##         DATUM["World Geodetic System 1984",
##             ELLIPSOID["WGS 84",6378137,298.257223563,
##                 LENGTHUNIT["metre",1]],
##             ID["EPSG",6326]],
##         PRIMEM["Greenwich",0,
##             ANGLEUNIT["degree",0.0174532925199433],
##             ID["EPSG",8901]]],
##     CONVERSION["unknown",
##         METHOD["Robinson"],
##         PARAMETER["Longitude of natural origin",0,
##             ANGLEUNIT["degree",0.0174532925199433],
##             ID["EPSG",8802]],
##         PARAMETER["False easting",0,
##             LENGTHUNIT["metre",1],
##             ID["EPSG",8806]],
##         PARAMETER["False northing",0,
##             LENGTHUNIT["metre",1],
##             ID["EPSG",8807]]],
##     CS[Cartesian,2],
##         AXIS["(E)",east,
##             ORDER[1],
##             LENGTHUNIT["metre",1,
##                 ID["EPSG",9001]]],
##         AXIS["(N)",north,
##             ORDER[2],
##             LENGTHUNIT["metre",1,
##                 ID["EPSG",9001]]]]
```

1.5. Sistema de información geográfica

En la década de los años sesenta, se produce el comienzo de una herramienta que nos ayuda hoy en día a comprender y visualizar de manera precisa nuestro mundo.
Con el inicio y auge de dos eventos, como fueron la computación y los conceptos de geografía cuantitativa y computacional, aparecieron los primeros desarrollos sobre sistemas de información geográfica.
Durante estos años se formalizaron las investigaciones referidas a la información geográfica y se impulsó una revolución cuantitativa en este ámbito.
El primer sistema de información geográfico computarizado se desarrolló en el año 1963.

En él se almacenó la información sobre los recursos naturales de todas las regiones de Canadá.

Un sistema de información geográfica (SIG), según García (2021), es una herramienta imprescindible para el tratamiento de datos espaciales y espacio temporales, es decir, una herramienta para trabajar con información georreferenciada.
Son el principal ámbito de aplicación de las bases de datos espaciales y se han convertido en un medio para inspirar a través de los datos.

Los SIG nos permiten realizar una serie de operaciones sobre los datos, como son: + Lectura + Almacenamiento + Gestión y edición + Análisis + Visualización + Creación de mapas

Se debe tener en cuenta que un sistema de información que maneje información espacial tiene que trabajar a la vez con ambas partes de la información: la forma del objeto definida en un plano y sus atributos temáticos asociados.
Por ejemplo, un lago que tiene su correspondiente forma geométrica plasmada en un plano tiene también otros datos asociados como pueden ser los niveles de contaminación.

Además, se deben controlar los componentes que intervienen en un sistema de información geográfica (García 2021).
Los datos son la materia prima de los SIG. Se obtienen de diversas fuentes como pueden ser archivos excel o fotografías aéreas y a partir de ellos podremos analizar y extraer la información necesaria.
El software es el componente que proporciona la potencia para llevar a cabo los procesos de análisis y visualización. Existen tecnologías especifícas como ArcGIS, QGIS o Gvsig, que nos aportan lo necesario para la computación.
El hardware es el componente físico. Cuanta mayor potencia tenga nuestro hardware menos latencia y más rendimiento obtendremos en nuestros estudios.
El equipo humano que lo desarrolla debe contener perfiles SIG, como técnicos, analistas y programadores, expertos en estas tecnologías para el correcto desarrollo del modelo.
Los métodos o procesos que ayudan a implementar de forma adecuada los sistemas en las empresas. Para ello, se rigen por unos procedimientos y reglas que facilitan el buen funcionamiento de los análisis.

Con todo esto, no cabe duda que los SIG se han convertido en una herramienta muy útil para estudiar la información geográfica y obtener resultados que se pueden aplicar al estudio y la mejora de la sociedad.
Entre las aplicaciones que más destacan se encuentran: + Aplicaciones científicas: modelización cartográfica o estudios sobre ciencias ambientales. + Labores relacionadas con la gestión: integración en el Catastro. + Aplicaciones empresariales: localización de áreas de interés según el comportamiento.

Capítulo 2

Marco teórico

2.1. Introducción

Como se ha visto los procesos espaciales y espacio temporales están presentes en la vida cotidiana y en la naturaleza que nos rodea.
Poder predecir estos procesos en localizaciones geográficas y en tiempo concreto ayuda a tomar medidas e intentar resolver conflictos ambientales, sociales y científicos.
Además, no sólo es necesario la predicción, sino también medir el error de dicha predicción para definir la eficacia y seguridad de las predicciones.

Existen muchos métodos para la predicción de datos espaciales y espacio temporales como pueden ser regresiones lineales, interpolaciones o kriging.
Sin embargo, el fixed rank kriging es una nueva visión para la estimación y predición ya que no se basa en modelos de variograma, sino en la construcción de un modelo denominado SRE o STRE, modelo de efectos espaciales aleatorios o modelo de efectos espacio temporales aleatorios, que predice en un dominio discretizado.
La unidad básica de predicción se denomina BAU, unidad básica de área. Las BAUs pueden construirse según las restricciones que se quiera aportar al modelo. Su número también depende de los ajustes que se hagan.

Se definen los modelos de efectos aleatorios espaciales y espacio temporales, así como sus componentes. A partir de ellos se elabora la teoría sobre el FRK, fixed rank kriging, y el FRF, fixed rank filtering. Ambos métodos ayudan al modelaje y predicción de los datos espaciales y espacio temporales, respectivamente.

2.2. Modelo de efectos aleatorios espaciales

Consideramos un proceso espacial $\{Y(s) : s \in D\}$, donde D es un dominio finito, infinitamente numerable o con una medida de Lebesgue positiva en $\Re^d$. Sean las observaciones del proceso las siguientes: $Z(s) = Y(s) + \varepsilon(s)$.

La componente $\{Y(s) = \mu(s) + v(s) : s \in D\}$ se corresponde con un modelo estadístico clásico con componentes: $\mu(s)$ es una función de tendencia determinista que puede

modelarse de numerosas formas. Una de ellas podría ser: $\mu(s) = t(s)'\alpha$, con $t(s)$ un vector de covariables espacialmente referenciadas, generalmente conocidas, y α un vector de coeficientes de regresión, generalmente desconocidos (Zammit-Mangion y Cressie 2017).

La componente $v(s) = \nu(s) + \xi(s)$ está formada por una componente $\nu(s) = \phi(s)_l\eta_l, \quad l = 1, ..., r$, con $\phi(s)$ un vector r dimensional de funciones básicas espaciales predefinidas y η un vector aleatorio gaussiano r dimensional de media 0 y matriz de covarianzas de dimensión $r \times r$ denotada por K.
Y un efecto aleatorio $\xi(s)$ que casi no está correlado espacialmente, es decir, que apenas hay relación lineal e incluso podría asumirse que es nula. En ocasiones se asume que el efecto aleatorio es gaussiano de media 0 y varianza desconocida.

Se observa que las funciones básicas espaciales $\phi(s)$ pueden ser o no ser ortogonales. Aquellas funciones básicas que se tomen deben ser capaces de reconstruir correctamente las realizaciones de $Y(\cdot)$.

Se asume que $E(\phi(s)\eta) = E(\xi(s)) = 0$ entonces $E(v(s)) = 0$. De esta forma, $v(s)$ puede expresarse en términos de una combinación lineal de un número fijado de funciones básicas espaciales que definen el modelo SRE para $v(s)$:

$$v(s) = \sum_{l=1}^{r} \phi(s)_l\eta_l + \xi(s), \quad s \in D$$

Ahora bien, la idea es poder dividir nuestro dominio D en distintas unidades de interés. Por tanto, se discretiza el conjunto D de la siguiente forma: $D^G \equiv \{A_i \subset D : \quad i = 1, .., N\}$, donde N es un número pequeño generalmente mayor que r, que representa el número de partes en el que se divide el dominio D.
Las agrupaciones A_i se denominan como unidades básicas de área (BAUs). Se tiene que son disjuntas y $D = \cup_{i=1}^{N} A_i$.
De esta forma, al discretizar el dominio D, la idea es obtener las realizaciones $\{Y(s), \quad s \in D\}$ mediante la media de las unidades básicas de área (BAUs), es decir:

$$Y_i \equiv \frac{1}{|A_i|} \int_{A_i} Y(s) \cdot ds, \quad i = 1, ..., N.$$

En general las BAUs se suponen con igual área, pero puede haber casos en que no sea así (Zammit-Mangion y Cressie 2017).

Por tanto, siguiendo esta idea, se tiene:

$$Y_i = \mu_i + \nu_i + \xi_i, \quad i = 1, ..., N.$$

El cálculo de la función de tendencia dependerá de cómo se tome su forma. Ya se vio que una estructura usual para esta función es: $\mu(s) = t(s)^{\perp}\alpha$. Por tanto, se tendría $t_i \equiv \left(\frac{1}{|A_i|} \int_{A_i} t(s) \cdot ds\right)^{\perp}, \quad i = 1, ..., N$. Se tomará este caso de aquí en adelante.
El siguiente término se calcula como sigue: $\nu_i \equiv \left(\frac{1}{|A_i|} \int_{A_i} \phi(s) \cdot ds\right)^{\perp} \eta, \quad i = 1, ..., N$.
Se denota $\mathbf{S} \equiv \left(\frac{1}{|A_i|} \int_{A_i} \phi(s) \cdot ds\right)^{\perp}, \quad i = 1, ..., N$ matriz de dimensión $N \times r$.
Por último, $\xi_i \equiv \left(\frac{1}{|A_i|} \int_{A_i} \xi(s) \cdot ds\right)^{\perp}$. Se supone que ξ es una efecto aleatorio gaussiano de media cero y varianza $var(\xi_i) = \sigma^2_{\xi_i}\nu_{\xi,i}$ donde la varianza de ξ es desconocida y los pesos

$\{\nu_{\xi,1}, ..., \nu_{\xi,N}\}$ son conocidos y generados por el investigador en cuestión basándose en la información sobre el dominio que se posee.

Una vez definido el dominio discretizado con sus correspondientes BAUs, se supone que el proceso $Y(\cdot)$ es observado en m localizaciones pudiendo abarcar una o más BAUs y además no tienen por qué ser disjuntas.
Normalmente $m \gg r$, teniendo en cuenta que puede darse tanto el caso en el que las localizaciones observadas sean un número mayor que el número de BAUs $m > N$, como el caso en el que dicho número de localizaciones observadas sea menor que el número de BAUs, $m \leq N$.
De esta forma, se define el dominio de observaciones como $D^O \equiv \{\cup_{i \in c_j} A_i : j = 1, ..., m\}$ con $c_j \neq \emptyset$ un subconjunto en $2^{\{1,...,N\}}$. Se observa fácilmente que $m = |D^O|$ (Zammit-Mangion y Cressie 2017).
Se concluye que cada elemento de D^O es una unión de subconjuntos de D^G, es decir, de BAUs.
Se ilustra esta idea de dominio discretizado y observado con un ejemplo. Sea D nuestro dominio, y se toma el siguiente dominio discretizado: $D^G = \{A_1, A_2, A_3, A_4\}$. Y sea el dominio observado $D^O = \{B_1, B_2\}$ con $B_1 = A_1 \cup A_3 \cup A_4$ y $B_2 = A_2$. Así, $c_1 = \{1, 3, 4\}$ y $c_2 = \{2\}$.

Hasta aquí se han definido todos los elementos del modelo. Esto podría servir para las posteriores secciones de estimación de parámetros y predicción del modelo FRK. Pero se debe ser conscientes que la medida del proceso Y es imperfecto, por ello para mejorar esta medida se calcula sobre el dominio observado el proceso Z:

$$Z_j \equiv Z(B_j) = \left(\frac{\sum_{i=1}^N Y_i w_{ij}}{\sum_{i=1}^N w_{ij}}\right) + \left(\frac{\sum_{i=1}^N \delta_i w_{ij}}{\sum_{i=1}^N w_{ij}}\right) + \epsilon_j, \quad B_j \in D^O$$

donde los pesos w_{ij} dependen de las BAUs y se definen así: $w_{ij} = |A_i| I(A_i \subset B_j), \quad i = 1, ..., N, \quad j = 1, ...m, \quad B_j \in D^O$ siendo $I(\cdot)$ la función indicadora.

La componente $\{\epsilon_i\}$ mide el error del proceso, siendo un vector gaussiano de media cero, varianza $Var(\epsilon_j) = \sigma^2_\epsilon \nu_{\epsilon,j}$ e independiente. El vector $\{\nu_{\epsilon,1}, ..., \nu_{\epsilon,m}\}$ es conocido.
Se denota $\boldsymbol{\epsilon} \equiv (\epsilon_j : \quad j = 1, ..., m)^\perp$ y por tanto $Var(\boldsymbol{\epsilon}) = \boldsymbol{\Sigma}_\epsilon \equiv \sigma^2_\epsilon \mathbf{V}_\epsilon \equiv \sigma^2_\epsilon diag(\nu_{\epsilon,1}, ..., \nu_{\epsilon,m})$ es una matriz diagonal de covarianzas de dimensión $m \times m$.
Podrían darse dos casuísticas: que la matriz $\boldsymbol{\Sigma}_\epsilon$ fuera conocida o que no lo fuera. En este último caso se tendría que estimar la componente σ^2_ϵ mediante alguna técnica. Kang, Liu y Cressie proponen estimarla usando técnicas del variograma (Zammit-Mangion y Cressie 2017).

La componente $\{\delta_i\}$ captura cualquier sesgo en el cálculo de las BAUs. Esta componente tiene media cero y varianza $Var(\delta_i) = \sigma^2_\delta \nu_{\delta,i}$ con el parámetro σ^2_δ desconocido y $\{\nu_{\delta,1}, \nu_{\delta,2}, ..., \nu_{\delta,N}\}$ conocidas. Además se supone la independencia de $\boldsymbol{\delta} \equiv \{\delta_j : \quad j = 1, ..., N\}$ e $\mathbf{Y} \equiv \{Y_i : \quad i = 1, ..., N\}$.

Por último se supone que las observaciones $\mathbf{Z} \equiv \{Z_j : \quad j = 1, ..., m\}^\perp$ son independendientes cuando aparecen condicionadas por $\boldsymbol{\delta}$ e $\mathbf{Y}$.

Ahora bien, como se ha mencionado antes, cada elemento del dominio observado D^O es unión de subconjuntos del dominio discretizado D^G. De esta forma, tiene sentido

construir la siguiente matriz:

$$\mathbf{C}_Z \equiv \left(\frac{w_{i,j}}{\sum_{l=1}^{N} w_{i,j}} : \quad i = 1, .., N; \quad j = 1, ..., m \right).$$

Así, se obtendría la siguiente expresión:

$$\mathbf{Z} = \mathbf{C}_Z \mathbf{Y} + \mathbf{C}_Z \boldsymbol{\delta} + \boldsymbol{\epsilon},$$

donde se ha supuesto que las tres componentes son independientes.
Se puede observar fácilmente que las filas de la matriz $\mathbf{C}_Z$ suman 1.

Con todo lo visto hasta ahora, se puede reescribir el modelo de la siguiente forma:

$$\mathbf{Y} = \mathbf{T}\boldsymbol{\alpha} + \mathbf{S}\boldsymbol{\eta} + \boldsymbol{\xi}, \tag{2.1}$$

con $\mathbf{T} \equiv (t_i : \quad i = 1, ..., N)^{\perp}$ y $\boldsymbol{\xi} \equiv (\xi_i : \quad i = 1, ..., N)^{\perp}$.

Además se reescribe $Var(\boldsymbol{\xi}_Z) = \sigma_\xi^2 \mathbf{V}_{\xi,Z} \equiv \sigma_\xi^2 \mathbf{C}_Z \mathbf{V}_\xi \mathbf{C}_Z^{\perp}$, $Var(\boldsymbol{\delta}_Z) = \sigma_\delta^2 \mathbf{V}_{\delta,Z} \equiv \sigma_\delta^2 \mathbf{C}_Z \mathbf{V}_\delta \mathbf{C}_Z^{\perp}$ y $\mathbf{V}_\delta \equiv diag(\nu_{\delta,1}, ..., \nu_{\delta,N})$ conocido.

Luego, agrupando todas las notaciones anteriores:

$$\begin{aligned} \mathbf{Z} &= \mathbf{C}_Z \mathbf{T}\boldsymbol{\alpha} + \mathbf{C}_Z \mathbf{S}\boldsymbol{\eta} + \mathbf{C}_Z \boldsymbol{\xi} + \mathbf{C}_Z \boldsymbol{\delta} + \boldsymbol{\epsilon} \\ &= \mathbf{T}_Z \boldsymbol{\alpha} + \mathbf{S}_Z \boldsymbol{\eta} + \boldsymbol{\xi}_Z + \boldsymbol{\delta}_Z + \boldsymbol{\epsilon}. \end{aligned}$$

Y teniendo en cuenta $E(\eta) = 0$ y $E(\boldsymbol{\xi}_Z) = E(\boldsymbol{\delta}_Z) = E(\boldsymbol{\epsilon}) = 0$ se tiene:

$$E(\mathbf{Z}) = \mathbf{T}_Z \boldsymbol{\alpha},$$

$$\begin{aligned} Var(\mathbf{Z}) &= Var(\mathbf{S}_Z \boldsymbol{\eta}) + Var(\boldsymbol{\xi}_Z) + Var(\boldsymbol{\delta}_Z) + Var(\boldsymbol{\epsilon}) \\ &= \mathbf{S}_Z \mathbf{K} \mathbf{S}_Z^{\perp} + \sigma_\xi^2 \mathbf{C}_Z \mathbf{V}_\xi \mathbf{C}_Z^{\perp} + \sigma_\delta^2 \mathbf{C}_Z \mathbf{V}_\delta \mathbf{C}_Z^{\perp} + \sigma_\epsilon^2 \mathbf{V}_\epsilon. \end{aligned}$$

2.3. Modelo de efectos aleatorios espacio temporales

Según Cressie et al. (2010), se considera un proceso espacio temporal $\{Y(s,t) : s \in D \subset \Re^d, \quad t \in \{1, 2, ...\}\}$, donde el dominio D puede ser finito, infinitamente numerable o tener una medida de Lebesgue positiva en $\Re^d$. Y se consideran observaciones del proceso de la siguiente forma:

$$Z(s;t) = Y(s;t) + \varepsilon(s;t),$$

donde $\{\varepsilon(s,t) : s \in D, \quad t \in \{1, 2, ...\}\}$ es un ruido blanco gaussiano de media cero y varianza positiva $var(\varepsilon(s;t)) = \sigma_\varepsilon^2 v_t(s) > 0$, con $\sigma_\varepsilon^2 > 0$. Además $E(\varepsilon(s;t)\varepsilon(r;u)) = 0$ siempre que $s \neq r$ y $t \neq u$.

Se supone que el proceso espacio temporal posee la siguiente estructura:

$$Y(s;t) = \mu_t(s) + v(s;t),$$

donde $\mu_t(\cdot)$ es una función de tendencia y $v(\cdot;t)$ es un proceso espacial que sigue un modelo de efectos aleatorios de media cero.

Se define a continuación cada componente del proceso espacio temporal.

La función de tendencia puede estar modelada por numerosas funciones. Una de las opciones más usadas es la siguiente: $\mu_t(\cdot) = X_t(\cdot)'\beta_t$ siendo $X_t(\cdot) = ()$ un vector de covariables conocido y beta un vector de coeficientes que generalmente son desconocidos.

La componente $v(\cdot;t)$ sigue un modelo espacial de efectos aleatorios para cualquier t fijado, como vimos en el epígrafe anterior (Cressie y Johannesson 2008):

$$v(s;t) = S_t(s)'\eta_t + \xi(s;t)$$

El primer término de este modelo representa la variación espacial para cada t fijado, siendo $S_t(\cdot) = (S_{1,t}(\cdot), S_{2,t}(\cdot), ..., S_{r,t}(\cdot))'$ un conjunto de r funciones espaciales conocidas y $\eta_t = (\eta_{1,t}, \eta_{2,t}, , ..., \eta_{r,t},)'$ un vector aleatorio gaussiano de media 0 con matriz de covarianzas de dimensión $r \times r$ dada por K_t.

El segundo término del modelo, $\xi(\cdot;\cdot)$, representa la variabilidad. Es un ruido blanco gaussiano con media cero y varianza σ^2_ξ , independiente de η_t.

Se observa que las funciones básicas $S_t(\cdot)$ no tienen porqué ser constantes en el tiempo. Es decir, cambian según el tiempo, aunque en muchas ocasionas se toman invariantes de forma que aporta estabilidad temporal al proceso $v(\cdot;\cdot)$.
Además, tampoco tienen porqué ser ortogonales.
Estas dos condiciones son elegidas, dependiendo del conjunto de funciones que se tome.

En el caso de que no se fije el tiempo, se observa que nuestra componente $v(s;t)$ se corresponde con un modelo espacio temporal de efectos aleatorios. Se asume que η_t, $\xi(\cdot;\cdot)$ y $\varepsilon(\cdot;\cdot)$ son independientes unos de otros. En este caso, la componente η_t sigue la siguiente fórmula recursiva:

$$\eta_{t+1} = H_{t+1}\eta_t + \zeta_{t+1}, \quad t = 1, 2, ... \tag{2.2}$$

Como se puede ver, esta fórmula define claramente un proceso autorregresivo de primer orden, donde H_{t+1} es la matriz autorregresiva de primer orden de dimensión $r \times r$ y el vector de innovación ζ_{t+1},independiente de η_t, tiene media cero y matriz de varianza de innovación $U_{t+1} = var(\zeta_{t+1})$.

Encontrarse con una fila de ceros en la matriz H_{t+1} indica que la componente correspondiente de η_{t+1} no crece dinámicamente desde η_t , pero sí que tiene dependencia en otras componentes de η_{t+1} a través de la componente ζ_{t+1}. Es decir, η_{t+1} no depende del pasado, y cuando lo hace es a través de la componente ζ_{t+1}.

Ahora bien, ya se han definido todas las componentes de cada proceso, de tal forma que el proceso de los datos seguiría un modelo espacio temporal de efectos mixtos:

$$Z(s;t) = \mu_t(s) + S_t(s)'\eta_t + \xi_(s;t) + \varepsilon(s;t); \quad s \in D, \quad t = 1, 2, ..$$

donde se observa que $\mu_t(\cdot)$ es determinista, es decir se conocen los parámetros, y el resto de componentes son estocásticas, por tanto se usan parámetros que se corresponden con

variables aleatorias. Estos parámetros tienen componentes estimadas.

Propiedades del modelo STRE
A continuación se definen algunas de las propiedades del modelo espacio temporal de efectos aleatorios.

- Las covarianzas cruzadas del vector aleatorio gaussiano η_{t_i}:

$$K_{t_1,t_2} = cov(\eta_{t_1}, \eta_{t_2}); \quad t_1, t_2 = 1, 2, ... \tag{2.3}$$

  Se observa que $K_{t,t} = K_t$.

- De (2.2) se obtiene:

$$K_{t_1,t_2} = K_{t_1}(H_{t_2}H_{t_2-1}...H_{t_1+1})', \quad para \quad t_1 < t_2 \qquad y$$
$$K_{t+1} = H_{t+1}K_tH'_{t+1} + U_{t+1}.$$

- Por tanto, de ambas expresiones se obtiene la siguiente igualdad que se denota por L_{t+1}:

$$L_{t+1} = K_{t,t+1} = K_tH'_{t+1}; \quad t = 1, 2, ... \quad ,$$

  donde se puede ver que la matriz L_{t+1} representa las covarianzas cruzadas en las componentes $\{\eta_t : t = 1, 2, ...\}$ del modelo espacio temporal de efectos aleatorios.

- De esta forma, basándose en H_{t+1}, se puede calcular las matrices U_{t+1} y K_1. Esto hace que se pueda calcular finalmente los efectos aleatorios $\{\eta_t : t = 1, 2, ...\}$.

El proceso de datos $Z(\cdot;\cdot)$ es observado en un número finito de momentos, n_t, y en las correspondientes localizaciones $\{s_{1,t}, .., s_{n_t,1}\}$ en un tiempo fijado t. Esto produce un vector n_t-dimensional de datos en el momento fijado t: $Z(t) = (Z(s_{1,t}; t), Z(s_{2,t}; t), ..., Z(s_{n_t,t}; t))', \quad t = 1, 2, ...$

Como ya se ha comentado, el objetivo es predecir el proceso espacio temporal $\{Y(s, t) : s \in D \subset \Re^d, t \in 1, 2, ...\}$, concretamente centrándose en predecir la componente $v(\cdot;\cdot)$, puesto que es la componente aleatoria del modelo. Se recuerda que la otra componente de $Y(s; t)$ es determinista.

Según el momento temporal que se quiera predecir, se diferencian tres casos:

- **Filtrado**: Como se conocen los datos en las localizaciones observadas en los momentos concretos,$Z(t) = (Z(1), Z(2), ..., Z(t))'$, se pasa a predecir $Y(s_0; t)$ a pesar de si hubiera o no un dato correspondiente a esa misma localización $Z(s_0; t)$. Esto se conoce como filtering o filtrado.

- **Suavizado**: En el caso de encontrarse prediciendo la variable $Y(s_0; u), \quad u \in \{1, 2, ..., t - 1\}$ a partir de las observaciones $Z(1), ..., Z(t)$ conocidas. Este proceso es conocido como smoothing o suavizado.

- **Predicción**: Y por último si se quisiera predecir la variable $Y(s_0; u), \quad u \in \{t+1, t+2, ...\}$ a partir de las observaciones $Z(1), ..., Z(t)$ conocidas el proceso es conocido como forecasting o predicción.

2.4. FRK datos espaciales

El objetivo es predecir para distintos valores del parámetro s. Se observa que es necesario únicamente predecir las componentes aleatorias del proceso, ya que el resto de componentes serán conocidas.

Se utiliza para ello las observaciones $Z(t)$ y se asume que la función de tendencia es conocida (ya sea definida como en el ejemplo del epígrafe anterior o de otra forma) y que $v(s) = \phi(s)\eta + \xi(s)$ sigue un modelo SRE.

2.4.1. Estimación

Como se ha visto anteriormente, las componentes aleatorias del modelo $Y(s) = \mu(s) + \phi(s)\eta + \xi(s), \quad s \in D$ son $\{\eta\} \quad y \quad \{\xi\}$. En este caso, se tendría que estimar ambas componentes, de tal forma que la complejidad aumenta bastante.

Por ello, se debe elegir una de las dos componentes para estimar. Así, si se decide modelar los errores sistemáticos del interior de las BAUs ($\boldsymbol{\delta}$) y en cuyo caso se fija $\sigma^2_\xi = 0$, o se estima el efecto aleatorio $\boldsymbol{\xi}$ y por tanto se fija $\sigma^2_\delta = 0$ (Zammit-Mangion y Cressie, 2017).
La estimación es simétrica para ambos casos, por lo que en este estudio se realiza la estimación de $\boldsymbol{\xi}$.

Ahora bien, se distinguen dos tipos de FRK (Zammit-Mangion y Cressie, 2017) dependiendo de la matriz de covarianzas K de la componente η:

- Cuando la matriz K no depende de ningún parámetro y por tanto se estima mediante métodos probabilísticos. Este caso es denominado FRK-V y los parámetros a estimar son $\theta = \{\alpha, \sigma^2_\xi, \sigma^2_\delta, K\}$.
- Cuando se impone alguna condición en términos de parámetros en la varianza de η, la matriz de covarianzas depende de dicho parámetro $K = K_0(\vartheta)$. En este caso, se debe estimar el parámetro $\theta_0 = \{\alpha, \sigma^2_\xi, \sigma^2_\delta, \vartheta\}$ y esta estimación se denomina FRK-M.

Se tiene en cuenta que dependiendo de lo que se quiera estimar uno de los parámetros lo suponemos 0, ya sea σ^2_ξ o σ^2_δ.

Es importante destacar que generalmente el vector de funciones básicas espaciales $\phi(\cdot) = (\phi_1(\cdot), ..., \phi_r(\cdot))^\perp$ está suficientemente suavizado y el tamaño de las BAUs es suficientemente pequeño (Zammit-Mangion y Cressie, 2017) de esta forma se puede aproximar la matriz S de la siguiente manera:

$$S \approx (\phi(s_i) : \quad i = 1, ..., N)^\perp$$

donde las localizaciones s_i se corresponden con los centroides de las BAUs.

Para la estimación se usará el algoritmo de maximización de la esperanza. Este algortimo se usa para encontrar estimadores de máxima verosimilitud de parámetros. El

proceso se divide en dos etapas diferenciadas: calcular la probabilidad condicionada respecto a los parámetros y maximizar dicha función con respecto a dichos parámetros, según Zammit-Mangion y Cressie (2017).

Sea $\mathbf{Z} = \mathbf{T}_Z\boldsymbol{\alpha} + \mathbf{S}_Z\boldsymbol{\eta} + \boldsymbol{\xi}_Z + \boldsymbol{\delta}_Z + \boldsymbol{\epsilon}$ el modelo y se denomina θ a los parámetros que se deben estimar, ya sea en el caso FRK-V o en FRK-M mencionados anteriormente.
Se define la función de probabilidad $L_c(\theta) \equiv [\boldsymbol{\eta}, \mathbf{Z}|\theta]$, donde $[\cdot]$ denota la distribución de probabilidad de sus argumentos, en este caso θ.

El primer paso es calcular la esperanza condicionada:

$$Q(\theta|\theta^{(l)}) \equiv E(lnL_c(\theta)|Z, \theta^{(l)}),$$

siendo $\theta^{(l)}$ un estimador de los parámetros.

El segundo paso es maximizar dicha esperanza condicionada para así obtener las estimaciones de los parámetros:

$$\theta^{(l+1)} = arg\max_{\theta} Q(\theta|\theta^{(l)}).$$

Para llevar a cabo el primer paso se necesita conocer la función de distribución condicionada a los parámetros θ de la variable η.

Se supone que dicha distribución condicionada sigue una normal, como Katzfuss y Cressie (2011) anuncia:

$$\eta|Z, \theta^{(l)} \sim \mathcal{N}(\mu_\eta^{(l)}, {\Sigma_\eta}^{(l)}),$$

donde los parámetros de la distribución normal se estiman de la siguiente forma:

$$\mu_\eta^{(l)} = \Sigma_\eta^{(l)} S_Z^{\perp} (D_Z^{(l)})^{-1} \left(Z - T_Z\alpha^{(l)}\right),$$

$$\Sigma_\eta{}^{(l)} = \left(S_Z^{\perp}\left(D_Z^{(l)}\right)^{-1} S_Z + \left(K^{(l)}\right)^{-1}\right)^{-1},$$

$$D_Z^{(l)} \equiv (\sigma_\xi^2)^{(l)} V_{\xi,Z} + \Sigma_\epsilon \tag{2.4}$$

Con todo lo anterior, recordando que se está desarrollando el caso en el que se supone $\sigma_\varepsilon^2 = 0$, la estimación de los parámetros sería la siguiente.

Se estima σ_ξ^2 mediante la solución de la siguiente ecuación, según Zammit-Mangion y Cressie (2017):

$$tr\left(\left(\Sigma_\epsilon + \left(\sigma_\xi^2\right)^{(l+1)} V_{\xi,Z}\right)^{-1} V_{\xi,Z}\right) =$$

$$tr\left(\left(\Sigma_\epsilon + \left(\sigma_\xi^2\right)^{(l+1)} V_{\xi,Z}\right)^{-1} V_{\xi,Z}\left(\Sigma_\epsilon + \left(\sigma_\xi^2\right)^{(l+1)} V_{\xi,Z}\right)^{-1}\Omega\right) \tag{2.5}$$

donde

$$\Omega \equiv S_Z\Sigma_\eta^{(l)}S_Z^{\perp} + S_Z\mu_\eta^{(l)}\mu_\eta^{(l)^{\perp}}S_Z^{\perp} - 2S_Z\mu_\eta^{(l)}\left(Z - T_Z\alpha^{(l+1)}\right)^{\perp} + \left(Z - T_Z\alpha^{(l+1)}\right)\left(Z - T_Z\alpha^{(l+1)}\right)^{\perp}.$$

Para la estimación de $\alpha^{(l+1)}$ se sustituye en la siguiente expresión la estimación de σ_ξ^2 calculada en (2.5):

$$\alpha^{(l+1)} = \left(T_Z^{\perp}\left(D_Z^{(l+1)}\right)^{-1} T_Z\right)^{-1} T_Z^{\perp}\left(D_Z^{(l+1)}\right)^{-1}\left(Z - S_Z\mu_\eta^{(l)}\right).$$

Si se trata de FRK-V, faltaría calcular el parámetro K:

$$K^{(l+1)} = \Sigma_\eta^{(l)} + \mu_\eta^{(l)} \mu_\eta^{(l)^\perp}$$

Si en cambio se trata de FRK-M, faltaría calcular el parámetro ϑ del que depende la matriz $K = K(\vartheta)$:

$$\vartheta^{(l+1)} = arg\,\max_{\vartheta} |K_0(\vartheta)^{-1}| - tr\left(K_0(\vartheta)^{-1}\left(\Sigma_\eta^{(l)} + \mu_\eta^{(l)}\mu_\eta^{(l)^\perp}\right)\right).$$

Ahora bien, existen casos en los que se simplifican los cálculos (Zammit-Mangion y Cressie, 2017).

- Cuando las matrices $V_{\xi,Z}$ y Σ_ϵ son diagonales, de esta forma sólo es necesario calcular la diagonal de la matriz Ω.
- Cuando las matrices $V_{\xi,Z}$ y Σ_ϵ son proporcionales a la matriz identidad. En este caso, la estimación de σ_ξ^2 queda reducida a la siguiente fórmula tomando como constantes de proporcionalidad γ_1 y γ_2, y recordando que $Z = (Z_1, ..., Z_m)$:

$$\left(\sigma_\xi^2\right)^{(l+1)} = \frac{1}{\gamma_1}\left(\frac{tr(\Omega)}{m} - \gamma_2\right)$$

Para la convergencia del algoritmo EM se calcula la siguiente función en cada iteración:

$$ln\left[Z|\alpha^{(l)}, K^{(l)}, \left(\sigma_\xi^2\right)^{(l)}\right] = -\frac{m}{2}ln(2\pi) - \frac{1}{2}ln|\Sigma_Z^{(l)}| - \frac{1}{2}\left(\mathbf{Z} - \mathbf{T}_Z\alpha(l)\right)^\perp \left(\Sigma_Z^{(l)}\right)^{-1}\left(\mathbf{Z} - \mathbf{T}_Z\alpha(l)\right)$$

donde $\Sigma_Z^{(l)} = \mathbf{S}_Z\mathbf{K}^{(l)}\mathbf{S}_Z^\perp + \mathbf{D}_Z^{(l)}$ y $\mathbf{D}_Z^{(l)}$ definida en (2.4).

2.4.2. Predicción

Para la predicción se toma un conjunto que contengan las regiones BAUs que queremos predecir y se predice el proceso Y en dichas regiones.

Sea $D^P = \{\tilde{B}_k : \quad k = 1, ..., N_P\}$ el dominio que se quiere predecir, que está formado por regiones BAUs calculadas usando el dominio discretizado $D^G = \{A_i \subset D : \quad i = 1, ..., N\}$, de tamaño N_P que indica el número de BAUs a predecir. Se observa que claramente $N_P = |D^P|$ (Zammit-Mangion y Cressie 2017).

Se toma ahora el siguiente proceso evaluado en las BAUs $\widetilde{B}_k$ siguiendo la idea que ya se vio en la construcción teórica del modelo de efectos aleatorios espaciales:

$$Y_{P,k} \equiv Y_P(\widetilde{B}_k) = \left(\frac{\sum_{i=1}^N Y_i\tilde{w}_{ij}}{\sum_{i=1}^N \tilde{w}_{ij}}\right), \quad \widetilde{B}_k \in D^P$$

donde se define $\mathbf{Y}_P = (Y_{p,k} : \quad k = 1, ..., N_p)^\perp$ y los pesos se corresponden con:

$$\tilde{w}_{ik} = |A_i| I(A_i \subset \widetilde{B}_k); \quad i = 1, ..., N; \quad k = 1, ..., N_P; \quad \widetilde{B}_k \in D^P.$$

De nuevo, se puede construir la matriz $\mathbf{C}_P \equiv \left(\frac{\tilde{w}_{ik}}{\sum_{l=1}^{N}\tilde{w}_{lk}}\right): \quad i = 1, ..., N; \quad k = 1, ..., N_P$ que análogamente a lo ya visto las filas de dicha matriz suman 1.

Por tanto, se puede expresar el proceso $\mathbf{Y}_P$ de la siguiente forma:

$$\mathbf{Y}_P = \mathbf{C}_P\mathbf{Y} = \mathbf{T}_P\alpha + \mathbf{S}_P\eta + \boldsymbol{\xi}_P =$$

$$\mathbf{C}_P\mathbf{T}\alpha + \mathbf{C}_P\mathbf{S}\eta + \mathbf{C}_P\xi.$$

Se observa que $Var(\xi_P) = \sigma_\xi^2\mathbf{V}_{\xi,P} \equiv \sigma_\xi^2\mathbf{C}_P\mathbf{V}_\xi\mathbf{C}_P^\perp$.

Las regiones a predecir $\{\tilde{B}_k\}$ pueden solaparse, al igual que ocurre con las observaciones.
Se debe destacar que en ocasiones no todas las regiones $\tilde{B}_k$ contienen una BAU. Cuando esto ocurra, se supondrá que contiene una BAU si el centroide de dicha BAU pertenece a la región $\tilde{B}_k$.

Dicho esto, sea l^* el número de iteraciones necesarias para las estimaciones de los parámetros mediante el algoritmo EM.
Se denotarán a los parámetros estimados como:

$$\hat{\mu}_\eta \equiv \mu_\eta^{(l^*)}, \quad \hat{\Sigma}_\eta \equiv \Sigma_\eta^{(l^*)}, \quad \hat{\alpha} \equiv \alpha^{(l^*)},$$

$$\hat{K} \equiv K^{(l^*)}, \quad \hat{\sigma}_\xi^2 \equiv \left(\sigma_\xi^2\right)^{(l^*)} \quad y \quad \hat{\sigma}_\delta^2 \equiv \left(\sigma_\delta^2\right)^{(l^*)}$$

Por tanto, la predicción se corresponderá para ambos casos con la siguiente expresión:

$$\hat{\mathbf{Y}}_P = E(\mathbf{Y}_P|\mathbf{Z}) = \mathbf{C}_P\hat{\mathbf{Y}}.$$

Y la varianza se estimaría según la siguiente expresión:

$$\Sigma_{Y_P|Z} = Var(\mathbf{Y}_P|\mathbf{Z}) = \mathbf{C}_P\Sigma_{Y_P|Z}\mathbf{C}_P^\perp$$

Véanse estas predicciones según los casos posibles que se han visto.

- Caso $\sigma_\xi^2 = 0$
  Bajo las hipótesis que se han ido asumiendo en el desarrollo, se tiene que $[\mathbf{Y}_P|\mathbf{Z}]$ sigue una distribución normal de la forma $N(\mathbf{Y}_P, \Sigma_{Y_P|Z})$.
  Se tiene que las predicciones son:

  $$\hat{\mathbf{Y}}_P = E(\mathbf{Y}_P|\mathbf{Z}) = \mathbf{T}_P\hat{\alpha} + \mathbf{S}_P\hat{\mu}_\eta$$

  $$\Sigma_{Y_P|Z} = Var(\mathbf{Y}_P|\mathbf{Z}) = \mathbf{S}_P\hat{\Sigma}_\eta\mathbf{S}_P^\perp$$

- Caso $\sigma_\delta^2 = 0$
  Sean $\mathbf{W} \equiv (\eta^\perp, \xi^\perp)$ y $\Pi \equiv (\mathbf{S}, \mathbf{I})$, entonces (2.1) puede escribirse como $\mathbf{Y} = \mathbf{T}\alpha + \Pi\mathbf{W}$.
  Esto daría lugar a los siguientes cálculos de la predicción:

  $$\hat{\mathbf{Y}}_P = E(\mathbf{Y}_P|\mathbf{Z}) = \mathbf{T}\hat{\alpha} + \Pi\hat{W}$$

$$\Sigma_{Y_P|Z} = Var(\mathbf{Y}_P|\mathbf{Z}) = \Pi\Sigma_W\Pi^{\perp}$$

donde

$$\hat{W} \equiv \Sigma_W\Pi^{\perp}\mathbf{C}_Z^{\perp}\Sigma_{\epsilon}^{-1}\left(\mathbf{Z} - \mathbf{T}_Z\hat{\alpha}\right),$$
$$\Sigma_W \equiv \left(\Pi^{\perp}\mathbf{C}_Z^{\perp}\Sigma_{\epsilon}^{-1}\mathbf{C}_Z\Pi + \Lambda^{-1}\right)^{-1}$$

con $\Lambda \equiv bdiag(\hat{K}, \hat{\sigma}_{\xi}^2\mathbf{V}_{\xi})$ una matriz de bloque diagonal.

2.5. FRF datos espacio temporales

En este apartado se van a calcular las predicciones de nuestro conjunto de observaciones $Z(1), Z(2), ..., Z(t)$ para los distintos casos que se han visto en secciones anteriores. Se recuerda que se poseen observaciones para muchos instantes de tiempo.

2.5.1. Filtrado

El objetivo es predecir $Y(s;t) = \mu_t(s) + S_t(s)'\eta_t + \xi(s;t)$ para un valor $s = s_0$ utilizando $Z(s_0,1), Z(s_0,2), ..., Z(s_0,t)$ que en adelante se denotará como $Z(1), ..., Z(t)$. Luego se quiere predecir $Y(s_0;t)$.
Para ello, se deben estimar las componentes aleatorias del modelo: η_t y $\xi(s_0;t)$, puesto que las componentes $\mu_t(s_0)$ y $S_t(s_0)'$ son funciones conocidas en el valor s_0 para cualquier t.
Para la estimación de ambas componentes se va a calcular la esperanza condicionada al conjunto de observaciones $Z(s_0,1), Z(s_0,2), ..., Z(s_0,t)$ de la variable.

2.5.1.1. Estimaciones

- **Estimación de η_t**

La estimación de la componente η_t es la siguiente, partiendo de unos valores iniciales $\hat{\eta}_{0|0}$ y $P_{0|0}$:

$$\hat{\eta}_{t|t} \equiv E(\eta_t|Z(1), ..., Z(t)) = \hat{\eta}_{t|t-1} + G_t\{Z(t) - \mu_t - S_t\hat{\eta}_{t|t-1}\}, \quad t = 1, 2, ...$$

donde

$$\hat{\eta}_{t|t-1} = H_t\hat{\eta}_{t-1|t-1},$$

siendo H_t la matriz de la ecuación (2.2) definida en el epígrafe anterior y G_t se define como

$$G_t = P_{t|t-1}S_t'(S_tP_{t|t-1}S_t' + D_t)^{-1} = P_{t|t-1}S_t'(D_t^{-1} - D_t^{-1}S_t\{P_{t|t-1}^{-1} + S_t'D_t^{-1}S_t\}S_t'D_t^{-1}) \quad (2.6)$$

con $D_t \equiv \sigma_{\xi}^2 I_{n_t} + \sigma_{\varepsilon}^2 V_t$,$V_t \equiv diag(\nu_t(s_{t,1}), ..., \nu_t(s_{t,n_t}))$ y $P_{t|t-1} = H_tP_{t-1|t-1}H_t' + U_t$ con U_t una matriz de dimensiones $r \times r$ tal que $U_t \equiv var(\zeta_t)$ según Cressie et al. (2010).

Por último, se tiene que

$$P_{t|t} \equiv E((\hat{\eta}_{t|t} - \eta_t)(\hat{\eta}_{t|t} - \eta_t)') = P_{t|t-1} - G_t S_t P_{t|t-1}. \tag{2.7}$$

La matriz $P_{t|t}$ es la matriz de errores cuadráticos medios de predicción, según Cressie et al. (2010).

- **Estimación de $\xi(s_0; t)$**

Caso $s_0 \in \{s_{1,t}, ..., s_{n_t,t}\}$
La estimación de $\xi(s_0; t)$ es la siguiente:

$$\hat{\xi}_{t|t}(s_0) \equiv E(\xi(s_0; t)|Z(1), ..., Z(t))$$
$$= c_t(s_0)'(S_t P_{t|t-1} S_t' + D_t)^{-1}(Z(t) - \mu_t - S_t \hat{\eta}_{t|t-1})$$

donde $c_t(s_0) \equiv cov(Z(t), \xi(s_0; t)) = \sigma_\xi^2 (I(s_0 = s_{1,t}), ..., I(s_0 = s_{n_t,t}))'$ ya que $Z(t)$ y $\xi(s_0; t)$ son independientes y $I(\cdot)$ es una función indicadora, como Cressie et al. (2010) define en su artículo.

Caso $s_0 \neq \{s_{1,t}, ..., s_{n_t,t}\}$
En este caso, la estimación del término $\xi(s_0; t)$ se reduce a 0, ya que $c_t(s_0) \equiv cov(Z(t), \xi(s_0; t)) = 0$ (Cressie et al., 2010).

2.5.1.2. Predicción y error estándar

Predicción FRF

– Caso $s_0 \in \{s_{1,t}, ..., s_{n_t,t}\}$
Por tanto, el estimador FRF (Cressie et al., 2010) es:

$$\hat{Y}(s_0; t)^{FRF} \equiv \hat{Y}(s_0; t|t)$$
$$= \mu_t(s_0) + S_t(s_0)' E(\eta_t|Z(1), ..., Z(t)) + E(\xi(s_0; t)|Z(1, ..., Z(t))), \quad s_0 \in D$$

– Caso $s_0 \neq \{s_{1,t}, ..., s_{n_t,t}\}$

$$\hat{Y}(s_0; t)^{FRF} = \mu_t(s_0) + S_t(s_0)' \hat{\eta}_{t|t}$$

Error estándar FRF

– Caso $s_0 \in \{s_{1,t}, ..., s_{n_t,t}\}$
Para calcular el error estándar FRF (Cressie et al., 2010) se utiliza la raíz cuadrada del cuadrado de la media del error de predicción:

$$\sigma(s_0; t)^{FRF} \equiv \sqrt{E(Y(s_0; t) - \hat{Y}(s_0; t)^{FRF})^2}$$
$$= \sqrt{E(S_t(s_0)'(\eta_t - \hat{\eta}_{t|t}) + \xi(s_0; t) - \hat{\xi}_{t|t}(s_0))^2}$$
$$= \sqrt{S_t(s_0)' P_{t|t} S_t(s_0) + \sigma_\xi^2 - c_t(s_0)'(S_t P_{t|t-1} S_t' + D_t)^{-1} c_t(s_0) - 2 S_t(s_0)' G_t c_t(s_0)}$$

– Caso $s_0 \neq \{s_{1,t}, ..., s_{n_t,t}\}$

$$\sigma(s_0; t)^{FRF} = \sqrt{S_t(s_0)' P_{t|t} S_t(s_0) + \sigma_\xi^2}$$

2.5.2. Suavizado

Se quiere encontrar la predicción óptima de $Y(s;u) = \mu_u(s) + S_u(s)'\eta_u + \xi(s;u)$ *para* $u \in \{1,...,t-1\}$. Se recuerda que las observaciones $Z(1),...,Z(t)$ son conocidas. De nuevo, se estiman las componentes aleatorias.

2.5.2.1. Estimaciones

- **Estimación de η_t**

$$\hat{\eta}_{u|t} \equiv E(\eta_u|Z(1),...,Z(t))$$
$$= \hat{\eta}_{u|u} + J_u\{\hat{\eta}_{u+1|t} - \hat{\eta}_{u+1|u}\}$$

donde

$$P_{u|t} \equiv E\left[(\hat{\eta}_{u|t} - \eta_u)(\hat{\eta}_{u|t} - \eta_u)'\right]$$
$$= P_{u|u} + J_u(P_{u+1|t} - P_{u+1|u})J_u', \quad u = 1,...,t-1 \quad y$$
$$J_u \equiv P_{u|u}H_{u+1}'P_{u+1|u}^{-1}$$

- **Estimación de $\xi(s_0;t)$**

Caso $s_0 \in \{s_{1,u},...,s_{n_u,u}\}$

$$\hat{\xi}_{u|t}(s_0) \equiv E(\xi(s_0;u)|Z(1),...,Z(t))$$
$$= \hat{\xi}_{u|u}(s_0) + M_u(s_0)'(\hat{\eta}_{u+1|t} - \hat{\eta}_{u+1|u})$$

donde

$$M_u(s_0) \equiv P_{u+1|u}^{-1}H_{u+1}R_{u|u}(s_0) \quad y$$
$$R_{u|u}(s_0) = -G_t c_t(s_0)$$

Caso $s_0 \neq \{s_{1,u},...,s_{n_u,u}\}$
En este caso, la estimación del término $\xi(s_0;u)$ se reduce a 0, ya que de nuevo $c_u(s_0) \equiv cov(Z(t),\xi(s_0;u)) = 0$ (Cressie et al., 2010).

2.5.2.2. Predicción y error estándar

Predicción FRF

– Caso $s_0 \in \{s_{1,u},...,s_{n_u,u}\}$ Por tanto, el estimador FRF (Cressie et al., 2010) es:

$$\hat{Y}(s_0;u)^{FRF} \equiv \hat{Y}(s_0;u|t)$$

$$= \mu_u(s_0) + S_u(s_0)'\hat{\eta}_{u|t} + \hat{\xi}_{u|t}(s_0), \quad u \in \{1,...,t-1\}$$

– Caso $s_0 \neq \{s_{1,u}, ..., s_{n_u,u}\}$

$$\hat{Y}(s_0; u|t)^{FRF} = \mu_u(s_0) + S_u(s_0)'\hat{\eta}_{u|t}$$

Error estándar FRF

– Caso $s_0 \in \{s_{1,u}, ..., s_{n_u,u}\}$

$$\sigma(s_0; u|t)^{FRF} \equiv \sqrt{E(Y(s_0; u|t) - \hat{Y}(s_0; u|t)^{FRF})^2}$$
$$= \sqrt{E(S_u(s_0)'(\eta_u - \hat{\eta}_{u|t}) + \xi(s_0; u) - \hat{\xi}_{u|t}(s_0))^2}$$
$$= \sqrt{S_u(s_0)'P_{u|t}S_u(s_0) + Q_{u|t}(s_0) + 2S_u(s_0)'R_{u|t}(s_0)}, \quad u = 1, ..., t-1$$

donde

$$Q_{u|t} \equiv var(\xi(s_0; u)|Z(1), ..., Z(t))$$
$$= Q_{u|u}(s_0) + M_u(s_0)'(P_{u+1|t} - P_{u+1|u})M_u(s_0), \quad y$$
$$Q_{u|u}(s_0) = \sigma^2_\xi - c_u(s_0)'(S_u P_{u|u-1} S'_u + D_u)^{-1} c_u(s_0)$$

– Caso $s_0 \neq \{s_{1,u}, ..., s_{n_u,u}\}$

$$\sigma(s_0; u|t)^{FRF} = \sqrt{S_u(s_0)'P_{u|t}S_u(s_0) + \sigma^2_\xi}$$

2.5.3. Predicción

Este último caso se centra en encontrar la predicción óptima de $Y(s; u) = \mu_u(s) + S_u(s)'\eta_u + \xi(s; u) \quad para \quad u \in \{t+1, t+2, ...\}$. Las observaciones $Z(1), ..., Z(t)$ son conocidas de nuevo, y se estiman las componentes aleatorias del modelo como se han visto en los otros dos casos anteriores.

2.5.3.1. Estimaciones

- **Estimación de η_t**

$$\hat{\eta}_{u|t} \equiv E(\eta_u|Z(1), ..., Z(t))$$
$$= \left(\prod_{i=t+1}^{u} H_i\right)\hat{\eta}_{t|t}$$

donde

$$P_{u|t} = \left(\prod_{i=t+1}^{u} H_i\right) P_{t|t} \left(\prod_{i=t+1}^{u} H_i\right)' + U_u + \sum_{i=t+1}^{u-1} \{\left(\prod_{j=i+1}^{u} H_j\right) U_i \left(\prod_{j=i+1}^{u} H_j\right)\}$$

Se observa que cuando u=t+1 el término correspondiente al sumatorio es cero, según Cressie et al. (2010).

- **Estimación de $\xi(s_0; t)$**

En este caso, al hacer estimación para parámetro u > t , se tiene la independencia de los términos $\xi(\cdot; u)$ y $Z(\cdot; 1), ..., Z(\cdot; t)$ (Cressie et al., 2010) precisamente porque tomamos valores para estimar u mayores que t. De esta forma, la estimación de $\xi(s_0; t)$ es trivialmente cero por independencia.

2.5.3.2. Predicción y error estándar

Predicción FRF

Por tanto, el estimador FRF (Cressie et al., 2010) es:

$$\hat{Y}(s_0;u)^{FRF} \equiv \hat{Y}(s_0;u|t)$$

$$= \mu_u(s_0) + S_u(s_0)'\hat{\eta}_{u|t}, \quad u \in \{1,...,t-1\}$$

siendo de nuevo H_t la matriz de la ecuación $\eta_{t+1} = H_{t+1}\eta_t + \zeta_{t+1}$.

Error estándar FRF

$$\sigma(s_0;u|t) = \sqrt{S_u(s_0)'P_{u|t}S_u(s_0) + \sigma^2_\xi}, \quad u = t+1, t+2, ...$$

Capítulo 3

Paquetes previos

3.1. Paquete Spacetime

El paquete spacetime basa su desarrollo en dos paquetes (Pebesma 2012): sp y xts. Del paquete sp toma las clases para los datos espaciales y del paquete xts toma las clases para objetos temporales.

El uso del paquete xts resulta de ayuda por diversos motivos: + Gracias al paquete zoo, que no se verá en este estudio pero en el que se basan muchas de las funciones del paquete xts, se extiende la funcionalidad y se tienen buenas herramientas para la agregación en el tiempo usando funciones de agregación aleatorias. + El paquete xts soporta diferentes tipos de representación del tiempo y la fecha, como son: **Date, POSIXt, timeDate, yearmon** y **yearqtr**.

Realmente, para el almacenamiento de datos espacio temporales se usan dataframes, pero los objetos temporales son almacenados con clases del paquete xts y los objetos espaciales con clases del paquete sp.

Por ejemplo, para interpolación espacial se utilizan las clases derivadas de la clase 'Spatial' del paquete sp. Además, esta librería posee una orden para dibujar dichos objetos: spplot, que se utiliza más adelante. Históricamente, el análisis de datos geográficos se centró en el paquete sp, que ha sido el más utilizado para manejar datos espaciales.

Es necesario remarcar que en las clases del paquete spacetime se registran los tiempos de comienzo y finalización de los datos (Pebesma 2012). Este paquete puede representar el tiempo como instantes o como intervalos. En el caso de no especificar el tiempo de finalización en los objetos, supone que el tiempo se identifica con instantes de tiempo. Si no especificamos el tiempo de finalización pero se está tratando con intervalos, la interpretación sería suponer que los intervalos se desarrollan de manera consecutiva y que el último intervalo tiene el mismo tamaño que el intervalo inmediatamente anterior.

```
methods(class="st_Construct")
```

```
## no methods found
```

3.1.1. Creación de objetos

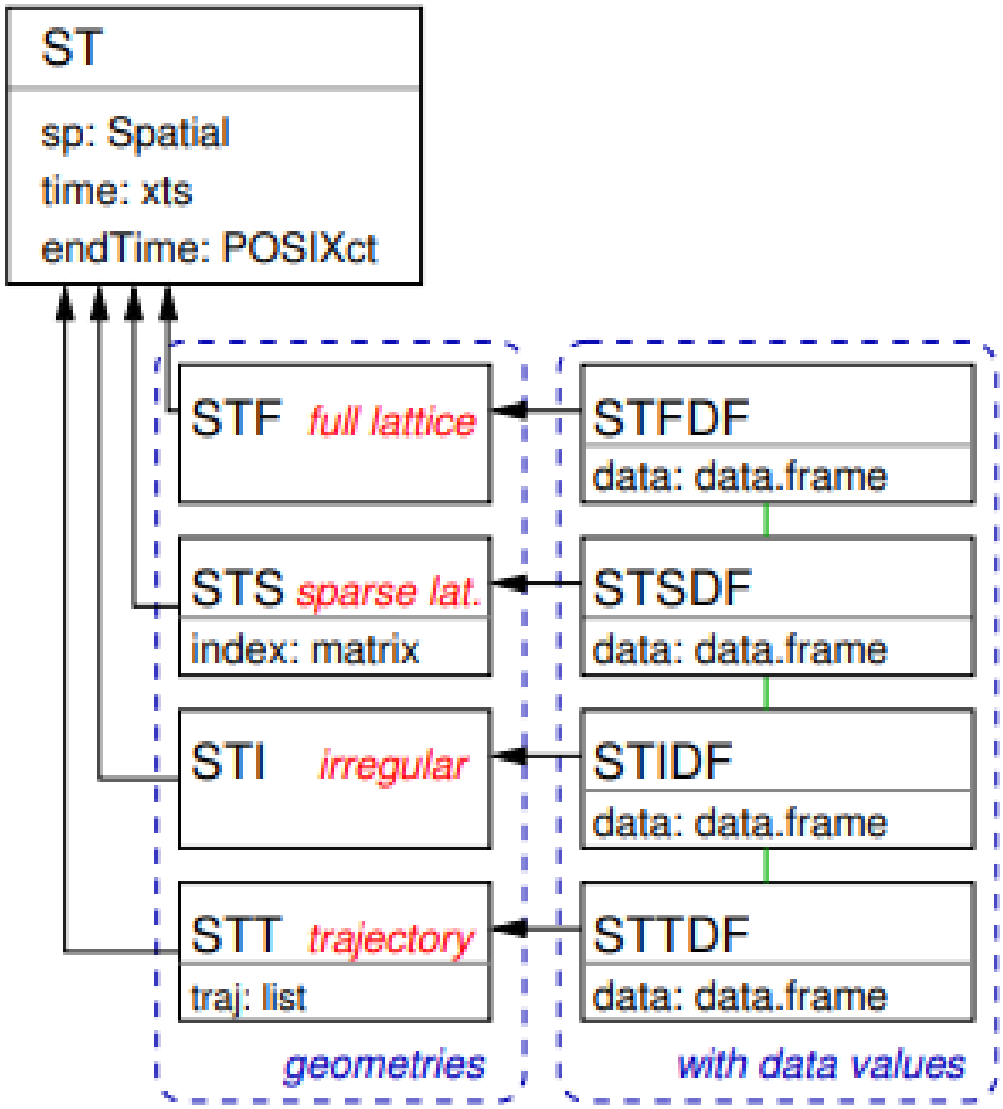
Se recuerdan las 4 formas de almacenamiento de datos espacio temporales que se vieron en el capítulo 1: cuadrícula completa, cuadrícula dispersa, datos no estructurados y trayectorias.

Para cada uno de estos tipos existe un objeto específico, de la clase stConstruct, en el paquete spacetime que lo modela. Todos ellos necesitan los siguientes argumentos: conjunto espacial, conjunto temporal y los valores de los datos contenido en un dataframe.

Además, se verá que se pueden convertir los objetos espacio temporales en otras clases como 'Spatial', 'matrix', 'dataframe' o 'xts'. Es decir, expresar el objeto espacio temporal en un objeto puramente espacial, en una matriz o dataframe, o en un objeto puramente temporal.

Se ilustra cómo el paquete spacetime se basa en el paquete xts para su componente temporal con una imagen:

Figura 3.1: Esquema equivalencia clases



Cuadrícula completa: 'STFDF'

Véase un ejemplo sencillo para captar la idea de las clases. Existen muchas otras opciones de proporcionar los datos temporales y espaciales para crear el objeto espacio temporal, pero aquí se verá solo una de ellas.

Se ilustra el ejemplo con datos del ratio de desempleados por estado de EEUU durante los años 1970-1986.

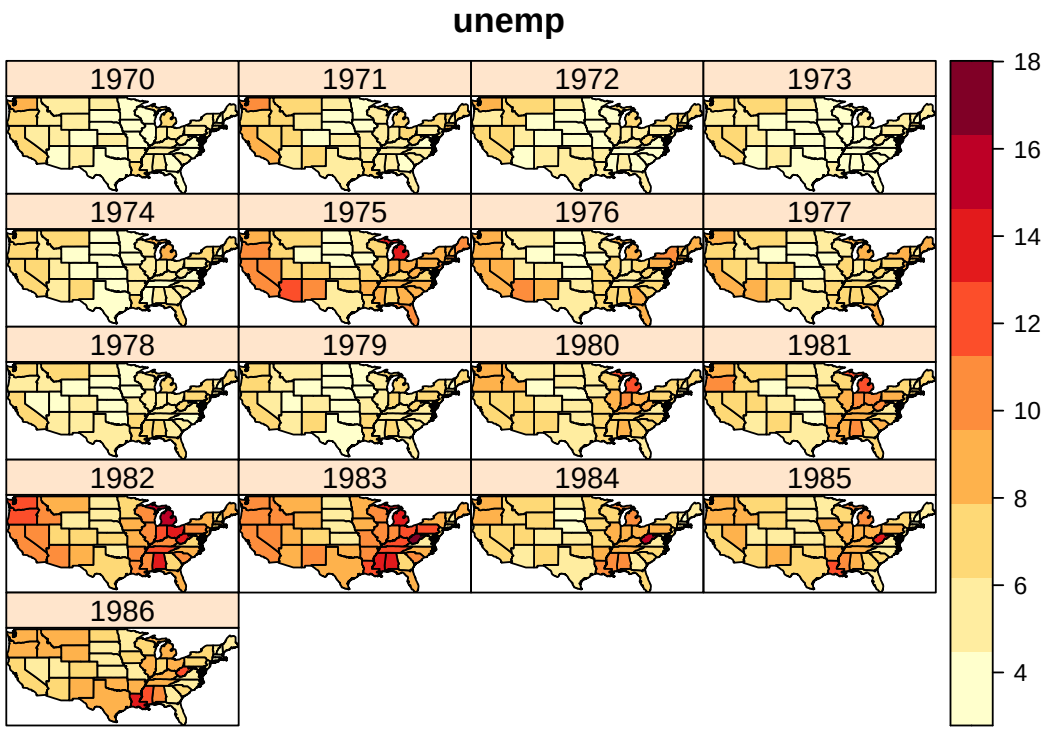
Los datos se encuentran en una cuadrícula completa y con distribución geométrica de polígonos correspondiendo a los estados.

```
estados.m = map("state", plot=FALSE, fill=TRUE)
IDs = sapply(strsplit(estados.m$names, ":"), function(x) x[1])

estados = map2SpatialPolygons(estados.m, IDs=IDs)
```

```
years = 1970:1986
time = as.POSIXct(paste(years, "-01-01", sep=""), tz = "GMT")
```

Se observa que los datos se encuentran almacenados en formato alargado.
Y se calcula el objeto espacio temporal.

```
data("Produc")

Produc.st = STFDF(estados[-8], time,
                  Produc[order(Produc[,2], Produc[,1]),])

stplot(Produc.st[,,"unemp"], years,
       col.regions = brewer.pal(9, "YlOrRd"),cuts=9)
```



Cuadrícula dispersa: 'STSDF'
En este caso se elabora manualmente un conjunto de puntos espaciales y un objeto temporal.

```
sp = cbind(x = c(0,0,1), y = c(0,1,1))
row.names(sp) = paste("point", 1:nrow(sp), sep="")

sp = SpatialPoints(sp)
time_stsdf = xts(1:4, as.POSIXct("2010-08-05")+3600*(10:13))
```

```r
m = c(10,20,30) # means for each of the 3 point locations

mydata = rnorm(length(sp)*length(time_stsdf),mean=rep(m, 4))
IDs = paste("ID",1:length(mydata))
mydata = data.frame(values = signif(mydata,3), ID=IDs)

stfdf = STFDF(sp, time_stsdf, mydata)
stsdf = as(stfdf, "STSDF")
stsdf
```

```
## An object of class "STSDF"
## Slot "data":
##    values    ID
## 1    9.41  ID 1
## 2   20.60  ID 2
## 3   31.60  ID 3
## 4   10.40  ID 4
## 5   20.20  ID 5
## 6   30.20  ID 6
## 7   10.50  ID 7
## 8   20.30  ID 8
## 9   31.30  ID 9
## 10  10.00 ID 10
## 11  20.50 ID 11
## 12  29.60 ID 12
##
## Slot "index":
##       [,1] [,2]
##  [1,]    1    1
##  [2,]    2    1
##  [3,]    3    1
##  [4,]    1    2
##  [5,]    2    2
##  [6,]    3    2
##  [7,]    1    3
##  [8,]    2    3
##  [9,]    3    3
## [10,]    1    4
## [11,]    2    4
## [12,]    3    4
##
## Slot "sp":
## SpatialPoints:
##        x y
## point1 0 0
## point2 0 1
## point3 1 1
## Coordinate Reference System (CRS) arguments: NA
```

```
##
## Slot "time":
##                      [,1]
## 2010-08-05 10:00:00    1
## 2010-08-05 11:00:00    2
## 2010-08-05 12:00:00    3
## 2010-08-05 13:00:00    4
##
## Slot "endTime":
## [1] "2010-08-05 11:00:00 CEST" "2010-08-05 12:00:00 CEST"
## [3] "2010-08-05 13:00:00 CEST" "2010-08-05 14:00:00 CEST"
```

Datos no estructurados: STIDF

Se va a tratar con un conjunto de datos contenido en el paquete cshapes que muestra las fronteras entre los países. Dicho conjunto de datos es un dataset almacenado en el sistema de información geográfica. En este caso, de nuevo, geométricamente se corresponde con polígonos.

```
cs = cshp()
names(cs)
```

```
##  [1] "gwcode"       "country_name" "start"        "end"          "status"
##  [6] "owner"        "capname"      "caplong"      "caplat"       "b_def"
## [11] "fid"          "geometry"
```

```
st = STIDF(geometry(as(cs, "Spatial")),
           as.POSIXct(cs$start), as.data.frame(cs), as.POSIXct(cs$end))

pt = SpatialPoints(cbind(7, 52), CRS(proj4string(st)))
as.data.frame(st[pt,,1:5])
```

```
##          V1       V2 sp.ID                time             endTime timeIndex
## 1 12.617005 51.62395    84 1886-01-01 00:09:21 1919-06-27 01:00:00        62
## 2 11.435140 51.35379    85 1919-06-28 01:00:00 1920-02-09 00:00:00        63
## 3 11.455945 51.31940    86 1920-02-10 00:00:00 1938-09-29 01:00:00        64
## 4 20.912593 54.19775    87 1938-09-30 01:00:00 1945-05-07 02:00:00        65
## 5  9.414213 50.57591    89 1949-09-21 01:00:00 1990-10-02 01:00:00        67
## 6 13.995306 53.96573    88 1990-10-03 01:00:00 2019-12-31 01:00:00        66
##   gwcode            country_name      start        end      status
## 1    255       Germany (Prussia) 1886-01-01 1919-06-27 independent
## 2    255       Germany (Prussia) 1919-06-28 1920-02-09 independent
## 3    255       Germany (Prussia) 1920-02-10 1938-09-29 independent
## 4    255       Germany (Prussia) 1938-09-30 1945-05-07 independent
## 5    260 German Federal Republic 1949-09-21 1990-10-02 independent
## 6    260 German Federal Republic 1990-10-03 2019-12-31 independent
```

Trayectorias: STTDF

Para mostrar la clase 'STTDF' se trabajará con un conjunto de datos dentro del paquete adehabitatLT que muestra el movimiento de animales.
En este paquete se utiliza la clase 'ltraj' para almacenar un objeto que contiene trayectorias.

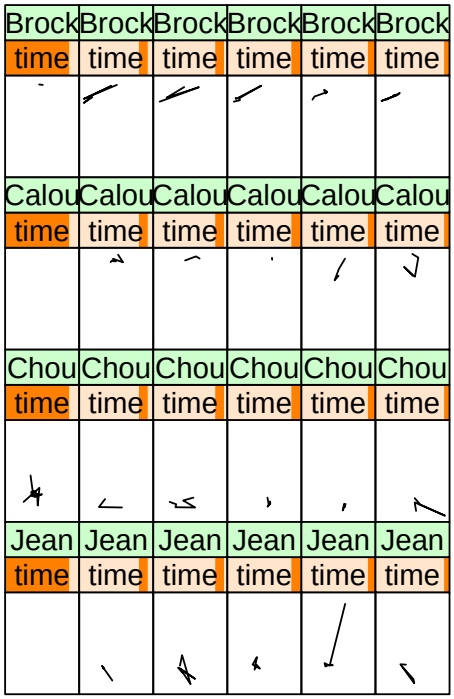
```
data("puechabonsp")

locs = puechabonsp$relocs #objeto de clase SpatialPointsDataFrame
xy = coordinates(locs)
```

Se va a convertir el formato de la columna Date (920729) a un objeto con la siguiente estructura: año-mes-dia.
Con esta reconversión, se define el objeto de clase 'ltraj' que contendrá nuestras trayectorias.
Se define una función que se usará como criterio para separar la trayectoria en tramos.
Se añade la opción nextr=T para al dividir en tramos, cuando aparezca una rama que cumpla el criterio se incluya en el tramo correspondiente.

```
data("puechabonsp")

locs = puechabonsp$relocs
xy = coordinates(locs)

date_habitant = as.character(locs$Date)
date_habitant = as.POSIXct(strptime(as.character(locs$Date),
                                    "%y%m%d", tz = "GMT"))

trajectories = as.ltraj(xy, date_habitant, id = locs$Name)
foo = function(dt) dt > 100*3600*24
l2 = cutltraj(trajectories, "foo(dt)", nextr = TRUE)
```

Por último, una vez construido el objeto de trayectorias correctamente, se construye el objeto que contendrá los datos espacio temporales.

```
sttdf = as(l2, "STTDF")
stplot(sttdf, by="time*id")
```



Se observa como se pueden obtener los valores almacenados en estas clases mediante índices, de forma similar a como se toman de una matriz o dataframe: air_quality/stfdf[i,j] ó air_quality/stfdf[i,j,k], siendo i representa la característica espacial, j el momento temporal y k la variable.

3.1.2. Representación gráfica

La orden stplot permite distintas formas de representar los objetos espacio temporales (Pebesma 2012).

- Gráfico multipanel

Para cada etapa temporal aparece un gráfico, no hay espacio entre ellos y generalmente aparece una leyenda con las indicaciones. Para el objeto STFDF aparecen tres opciones según la información que aporten los ejes: + Ambos ejes representan las características espaciales. + Eje x denota el tiempo y eje y el valor correspondiente y cada panel corresponde con un característica espacial diferente. + Eje x denota el tiempo y eje y el valor correspondiente y cada panel corresponde con cada variable de los datos. En estos últimos dos casos, los valores vienen conectados por líneas, como en los gráficos usuales, pudiéndose cambiar usando la opción type=“p”.

- Gráfico espacio-tiempo

Se representa el espacio en el eje x y el tiempo en el eje y.

- Gráfico animado

Se representa una secuencia de gráficos spplots para cada momento temporal. Para ello se usará el parámetro animate de la orden stplot, indicando el tiempo en segundos para realizar la pausa entre los componentes de la secuencia.

- Gráfico series temporales

Este tipo de gráficos son muy comunes debido a la gran cantidad de conjuntos que se toman como series temporales. Existen muchas formas de representar gráficamente una serie temporal mediante paquetes como ggplot2 o lattice.

3.2. Paquete sf

El paquete sf es un paquete que se encuentra en desarrollo con la intención de mejorar las clases del paquete sp y llegar a sustituirlo a corto plazo.
Se centra principalmente en trabajar con objetos más simples.
No se utilizá durante este desarrollo, pero se ilustrará un breve resumen del potencial que posee.

Los objetos de este paquete se basan en extensiones de dataframes, de tal forma que son objetos que poseen mínimo una columna denominada característica simple de geometría conteniendo la geometría de cada observación.
Esta nueva visión hace que cada fila se considere una característica simple, especificando tanto la geometría como otras variables de interés, según Pebesma (2018).

Se ilustra con un ejemplo. Se cargan unos datos contenidos en el paquete sf y se ve la clase de los mismos.

```
nc = st_read(system.file("shape/nc.shp", package="sf"), quiet = TRUE)
head(nc)
```

```
## Simple feature collection with 6 features and 14 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -81.74107 ymin: 36.07282 xmax: -75.77316 ymax: 36.58965
## Geodetic CRS:  NAD27
##    AREA PERIMETER CNTY_ CNTY_ID        NAME  FIPS FIPSNO CRESS_ID BIR74 SID74
## 1 0.114     1.442  1825    1825        Ashe 37009  37009        5  1091     1
## 2 0.061     1.231  1827    1827   Alleghany 37005  37005        3   487     0
## 3 0.143     1.630  1828    1828       Surry 37171  37171       86  3188     5
## 4 0.070     2.968  1831    1831   Currituck 37053  37053       27   508     1
## 5 0.153     2.206  1832    1832 Northampton 37131  37131       66  1421     9
## 6 0.097     1.670  1833    1833    Hertford 37091  37091       46  1452     7
##   NWBIR74 BIR79 SID79 NWBIR79                       geometry
## 1      10  1364     0      19 MULTIPOLYGON (((-81.47276 3...
## 2      10   542     3      12 MULTIPOLYGON (((-81.23989 3...
```

```
## 3     208  3616     6     260 MULTIPOLYGON (((-80.45634 3...
## 4     123   830     2     145 MULTIPOLYGON (((-76.00897 3...
## 5    1066  1606     3    1197 MULTIPOLYGON (((-77.21767 3...
## 6     954  1838     5    1237 MULTIPOLYGON (((-76.74506 3...
```

```
str(nc)
```

```
## Classes 'sf' and 'data.frame':   100 obs. of  15 variables:
##  $ AREA     : num  0.114 0.061 0.143 0.07 0.153 0.097 0.062 0.091 0.118 0.124 ...
##  $ PERIMETER: num  1.44 1.23 1.63 2.97 2.21 ...
##  $ CNTY_    : num  1825 1827 1828 1831 1832 ...
##  $ CNTY_ID  : num  1825 1827 1828 1831 1832 ...
##  $ NAME     : chr  "Ashe" "Alleghany" "Surry" "Currituck" ...
##  $ FIPS     : chr  "37009" "37005" "37171" "37053" ...
##  $ FIPSNO   : num  37009 37005 37171 37053 37131 ...
##  $ CRESS_ID : int  5 3 86 27 66 46 15 37 93 85 ...
##  $ BIR74    : num  1091 487 3188 508 1421 ...
##  $ SID74    : num  1 0 5 1 9 7 0 0 4 1 ...
##  $ NWBIR74  : num  10 10 208 123 1066 ...
##  $ BIR79    : num  1364 542 3616 830 1606 ...
##  $ SID79    : num  0 3 6 2 3 5 2 2 2 5 ...
##  $ NWBIR79  : num  19 12 260 145 1197 ...
##  $ geometry :sfc_MULTIPOLYGON of length 100; first list element: List of 1
##   ..$ :List of 1
##   .. ..$ : num [1:27, 1:2] -81.5 -81.5 -81.6 -81.6 -81.7 ...
##   ..- attr(*, "class")= chr [1:3] "XY" "MULTIPOLYGON" "sfg"
##  - attr(*, "sf_column")= chr "geometry"
##  - attr(*, "agr")= Factor w/ 3 levels "constant","aggregate",..: NA NA NA NA NA NA
##   ..- attr(*, "names")= chr [1:14] "AREA" "PERIMETER" "CNTY_" "CNTY_ID" ...
```

Se puede observar como en la columna geometry, que es del tipo sfc (simple feature geometry list column), aparece un atributo denominado sf_column que contiene la geometría del objeto. Se puede acceder directamente con st_geometry.

```
st_geometry(nc)
```

```
## Geometry set for 100 features
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -84.32385 ymin: 33.88199 xmax: -75.45698 ymax: 36.58965
## Geodetic CRS:  NAD27
## First 5 geometries:
```

Los comandos que contiene este paquete son los siguientes:

```
methods(class="sf")
```

```
##  [1] $<-                  [                    [[<-
##  [4] aggregate            anti_join            arrange
##  [7] as.data.frame        cbind                coerce
## [10] dbDataType           dbWriteTable         distinct
## [13] dplyr_reconstruct    filter               full_join
## [16] group_by             group_split          identify
## [19] idw                  initialize           inner_join
## [22] krige                krige.cv             left_join
## [25] merge                mutate               plot
## [28] print                rbind                rename
## [31] right_join           rowwise              sample_frac
## [34] sample_n             select               semi_join
## [37] show                 slice                slotsFromS3
## [40] st_agr               st_agr<-             st_area
## [43] st_as_s2             st_as_sf             st_as_sfc
## [46] st_bbox              st_boundary          st_buffer
## [49] st_cast              st_centroid          st_collection_extract
## [52] st_convex_hull       st_coordinates       st_crop
## [55] st_crs               st_crs<-             st_difference
## [58] st_drop_geometry     st_filter            st_geometry
## [61] st_geometry<-        st_inscribed_circle  st_interpolate_aw
## [64] st_intersection      st_intersects        st_is
## [67] st_is_valid          st_join              st_line_merge
## [70] st_m_range           st_make_valid        st_nearest_points
## [73] st_node              st_normalize         st_point_on_surface
## [76] st_polygonize        st_precision         st_reverse
## [79] st_sample            st_segmentize        st_set_precision
## [82] st_shift_longitude   st_simplify         st_snap
## [85] st_sym_difference    st_transform         st_triangulate
## [88] st_union             st_voronoi           st_wrap_dateline
## [91] st_write             st_z_range           st_zm
## [94] summarise            transform            transmute
## [97] ungroup
## see '?methods' for accessing help and source code
```

Cabe destacar que los objetos geométricos son del tipo sfg (simple feature geometry). Existen 17 tipos, todos se pueden ver en los comandos de arriba, aunque algunos de los más utilizados son: st_point, st_multipoint, st_polygon, st_multipolygon, etc.

Se debe mencionar que los objetos sfg y sfc pueden combinarse mediante la función st_sfg.
Se trabajará con objetos del tipo sfg y sfc, o con dataframes, ambos se pueden convertir a un objeto de clase sf mediante la función st_sf o st_as_sf respectivamente.

Por último, se mencionará que aunque se pueda representar gráficamente objetos de este paquete con la función plot, el paquete sf trabaja sin problemas con el paquete tidyverse y además posee comandos para interactuar con los paquetes dplyr, tidyr y ggplot2.

Capítulo 4

Paquete FRK

4.1. Descripción del paquete

El paquete FRK es una herramienta para modelar y predecir datos espaciales y espacio temporales. De hecho, este paquete apareció para trabajar con grandes cantidades de datos.
Para hacerse una idea, se puede trabajar hasta con unos pocos cientos de millones de puntos espaciales en una máquina personal y hasta unos pocos millones de datos espaciales en máquinas con más capacidad de memoria.

El modelo utiliza un conjunto de funciones básicas, que poseen la información de covarianza. Esta representación de funciones básicas facilita el modelado de datos espaciales y espacio temporales. El método naturalmente permite funciones de covarianza anisotrópicas no estacionarias.

La discretización del dominio espacial en las denominadas unidades básicas de área (BAU) facilita el uso de observaciones con diferentes soportes (es decir, soportes de área y puntos de referencia, potencialmente simultáneamente) y la predicción sobre regiones arbitrarias especificadas por el usuario.

Este paquete también admite la inferencia sobre varias variedades, como el plano 2D y la esfera 3D, y proporciona funciones auxiliares para modelar, ajustar, predecir y trazar con relativa facilidad.

La versión 2.0.0 y superior del paquete FRK también admite el modelado de datos no gaussianos, aunque el desarrollo que se expone se base en datos gaussianos, mediante el empleo de un marco de modelo mixto lineal generalizado espacial (GLMM) para atender a las distribuciones Poisson, binomial, binomial negativa, gamma y gaussiana inversa (Zammit-Mangion y Sainsbury-Dale, 2012).

El paquete FRK se apoya en los paquete spacetime y sp.
Las principales clases de este paquete se encuentran recogidas en la siguiente tabla:

Figura 4.1: Clases sp y spacetime

Class	Description
`Basis`	Defines basis functions on a specified manifold.
`Basis_obj`	A virtual class that other basis classes inherit from.
`manifold`	A virtual class that other manifold classes inherit from.
`measure`	Defines objects that compute distances on a specified manifold.
`plane, real_line, sphere, STplane, STsphere, STmanifold`	Subclasses that inherit from the virtual class `manifold`.
`SRE`	Defines the spatial-random-effects model, which is used to do FRK.
`TensorP_Basis`	Tensor product of two basis functions.

Table 1: Important class definitions in **FRK**.

Este paquete tiene diversas aplicaciones (Zammit-Mangion y Sainsbury-Dale, 2012) siendo algunas de ellas las siguientes:

- Integración de múltiples observaciones con relativa facilidad.
- Obtección de las predicciones de millones de localizaciones, es decir, capaz de trabajar con grandes cantidades de datos, mediante el uso de técnicas algebraicas sin necesidad de una simulación condicional.
- Distinción entre el efecto aleatorio y el error de medida con respecto al cálculo de las BAUs.

La componente temporal se incluye añadiendo otra dimensión, como ya se ha visto en el capítulo anterior.
Una de las claves de los modelos SRE y STRE es el uso y definición de las funciones básicas espaciales y espacio temporales, que se puede generar automáticamente con el paquete FRK.

Además, para la visualización de objetos se usa el método 'plot_spatial_or_ST', que acepta no solo objetos espaciales sino también de clase 'STFDF'.

Se van a abordar ejemplos de predicción de datos usando el modelo FRK que se han visto teóricamente en el capítulo anterior. Para ello, se observa en primer lugar que existen dos métodos para hacerlo según el nivel de control que se quiera tener sobre los datos:

Método 1.
Utilizar directamente las funciones contenidas en el paquete FRK. De este modo se tienen menos control sobre las características de los datos, ya que el modelo SRE se produce automáticamente según carguemos los datos. Se suele usar para casos más sencillos.

Este método es más simple que el que se verá más adelante, se utiliza el objeto FRK para generar un modelo SRE con las estimaciones y predicciones. Únicamente serían imprescindibles los siguientes parámetros: la fórmula que relaciona la variable dependiente y el conjunto de datos.

Pero existen muchos otros argumentos con los cuales se dan indicaciones sobre las funciones básicas y las BAUs, que se generarán automáticamente.

Es necesario comentar que las covarianzas no vendrán suministradas por los datos sino por las BAUs. Además, la intersección de los datos con las BAUs nunca debería ser nula.

El número de BAUs que creará automáticamente el objeto en caso de no poseerlas depende del número de datos espaciales disponibles en el conjunto a predecir.
Existe la posibilidad de apoyarse en funciones geométricas del paquete INLA para la construcción de las BAUs y las funciones básicas. Sin embargo, este paquete no se encuentra en los repositorios del CRAN de R y debe añadirse manualmente.

Por tanto, se resumen algunos de los argumentos del objeto FRK:

- Basis: para indicar las funciones básicas en el caso de construirlas o tenerlas.
- BAUs: para indicar las correspondientes BAUs, que deben estar almacenadas en objetos de clase STIDF, STFDF, SpatialPolygonsDataFrame o SpatialPixelsDataFrame.
- Est_error: cálculo del error de la estimación de la varianza. Sólo puede aplicarse en caso de que se suponga la variable de respuesta como gaussiana.
- Vgm_model: crea un objeto de clase variogramModel del paquete gstat que contiene el variograma. Sólo puede aplicarse en caso de que se suponga la variable de respuesta como gaussiana.
- K_type: parametrización usada para la matriz de covarianzas de las funciones básicas. Dependiendo del algoritmo usado para la estimación de los parámetros la parametrización será de un tipo u otro.
- N_EM: número máximo de iteraciones al aplicar el algoritmo EM.
- Method: método para la estimación de los parámetros usado. Actualmente las opciones son: EM y TMB.
- Reponse: cadena que supone la distribución que la variable respuesta posee.
- Pred_time: vector de tiempo que indica en qué momentos se quiera realizar la predicción.
- Covariances: argumento lógico. True indica que deben devolverse las predicciones de las covarianzas. Sólo puede aplicarse si el método de estimación es el algoritmo EM.
- Regular: argumento con dos posibles opciones: 0 y 1. En caso de 0 indica que las funciones básicas son colocadas en una función de densidad de datos y en el caso de 1 las funciones básicas son colocadas en una cuadrícula.

Método 2.
Para un mayor control y uso más avanzado se seguiría un procedimiento más detallado mediante el cual se construyen manualmente ciertos elementos del modelo como la disposición de las BAUs y las funciones espaciales básicas.

Véanse los pasos a seguir para implementar este método:

- Primer paso
  Se cargan los datos y los alojamos en un objeto de clase 'STFDF' o 'STIDF' si se utiliza la librería spacetime o de clase 'SpatialPointsDataFrame' o 'SpatialPolygonsDataFrame' si se utiliza la librería sp.

- Segundo paso
  Se construye una preddición de las BAUs con ayuda de la función auto_BAUs. Mediante esta función se discretizada el dominio. Además contiene argumentos como manifold que permiten escoger el espacio sobre el que se calculan las BAUs (esfera o plano).
  Las BAUs deben tener clase 'Spatial', 'ST' pixel o corresponderse con polígonos (SpatialPixelsDataFrame, SpatialPolygonsDataFrame).

- Tercer paso
  Se construyen las funciones básicas. En el caso de datos espaciales sólo habrá que calculas las funciones básicas espaciales. En el caso de datos espacio temporales habrá que calcular las funciones básicas espacio temporales, que generalmente se calculan realizando el producto con la función 'Tensor_Product' entre las funciones básicas espaciales y temporales calculadas independientemente.
  Para la construcción se utiliza la función auto_basis. Contiene argumentos como type que nos permiten elegir la distribución de las funciones básicas que se calculan: gaussiana, exponencial, etc.
  Las funciones básicas están construidas sobre los espacios $\mathbb{R}$ o $\mathbb{S}^2$.

- Cuarto paso
  Una vez calculado lo anterior, se utiliza la función SRE para calcular el modelo SRE. Tiene como argumentos principalmente: la fórmula con la que se modela la variable dependiente, los datos, las funciones básicas y las BAUs.

- Quinto paso
  Se estiman los parámetos contenidos en el modelo SRE usando SRE.fit.

- Sexto paso
  Por último se predice usando la función predict, obteniendo como resultado en caso de datos espaciales un objeto de clase 'SpatialPolygon' o 'SpatialPolygonDataFrame', y en caso de datos espacio temporales un objeto de clase 'STFDF'.

La función predict permite un argumento, type = c("link", "mean"), que obtiene las predicciones tanto para el proceso $Y(\cdot)$ como para la tendencia $mu(\cdot)$.

Se debe tener en cuenta ciertas necesidades para el uso del paquete FRK en un conjunto de datos.

Los datos pueden pertenecer a las clases 'SpatialPointsDataFrame'(puntos) 'SpatialPolygonsDataFrames'(polígonos), 'STIDF' o 'STFDF'.
Los sistemas de referencias de coordenadas de las BAUs y los datos deben ser compatibles en nuestro espacio donde se construyen las funciones básicas.

Para el cálculo de distancias se utilizan la distancia euclídea o la distancia ortodrómica (distancia sobre la superficie terrestre).

Es interesante saber que se está investigando para implementar en el paquete FRK otras opciones sobre el espacio donde se desarrollan las funciones básicas y la distancia que se usa.

4.2. Implementación datos espaciales

4.2.1. Método 1

Para la implementación del primer método se va a utilizar un conjunto de datos contenido en el paquete sp: meuse.
Este conjunto contiene la polución del suelo del río Meuse con muestras de concentración de metales pesados en una región de Holanda.

Además, viene incorporado con un conjunto de datos meuse.grid que contiene las BAUs con la información de las covarianzas, de tal forma que se mostrarán las dos formas posibles de implementar el FRK: aportando las covarianzas y sin aportarlas.

El objetivo de este ejemplo es analizar la distribución de la concentración de zinc.

4.2.1.1. Sin covarianzas

Como se vio en la sección anterior, se va a especificar con el argumento regular que las funciones básicas espaciales se distribuyan en una función de densidad. Además, se calculan mediante funciones como loglik o info_fit características del modelo SRE.

```
data("meuse", package="sp")
coordinates(meuse) = ~x+y

class(meuse)
```

```
## [1] "SpatialPointsDataFrame"
## attr(,"package")
## [1] "sp"
```

```
f_meuse_notCov = log(zinc) ~ 1

S_meuse_notCov = FRK(f=f_meuse_notCov, data=list(meuse),
                     regular=0, nonconvex_hull=FALSE )
```

```
## Constructing BAUs...
## Generating basis functions...
## Automatically choosing number of functions...
## Modelling using 382 basis functions.
## Constructing SRE model...
## Normalising basis function evaluations at BAU level...
## Fitting variogram for estimating measurement error...
## Binning data...
## Binned data in 0.37 seconds
## Fitting SRE model...
##   |                                                                      |
```

```
# summary(S_meuse_notCov)

BAUs_observadas = observed_BAUs(S_meuse_notCov)
# BAUs observadas
BAUs_no_observadas = unobserved_BAUs(S_meuse_notCov)
# BAUs no observadas

info_fit(S_meuse_notCov) # información de ajuste
```

```
## $method
## [1] "EM"
##
## $num_iterations
## [1] 24
##
## $sigma2fshat_equal_0
## [1] 0
##
## $converged
## [1] 1
##
## $plot_lik
## $plot_lik$x
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
##
## $plot_lik$llk
##  [1] -120.0248 -105.2435 -102.5000 -101.8082 -101.5577 -101.4217 -101.3397
##  [8] -101.2782 -101.2313 -101.1942 -101.1641 -101.1366 -101.1124 -101.0908
## [15] -101.0715 -101.0540 -101.0381 -101.0234 -101.0100 -100.9974 -100.9857
## [22] -100.9747 -100.9643 -100.9544
##
## $plot_lik$ylab
## [1] "log likelihood"
##
## $plot_lik$xlab
## [1] "EM iteration"
```

```
##
##
## $time
##    user  system elapsed
##   32.56    1.48   34.80
```

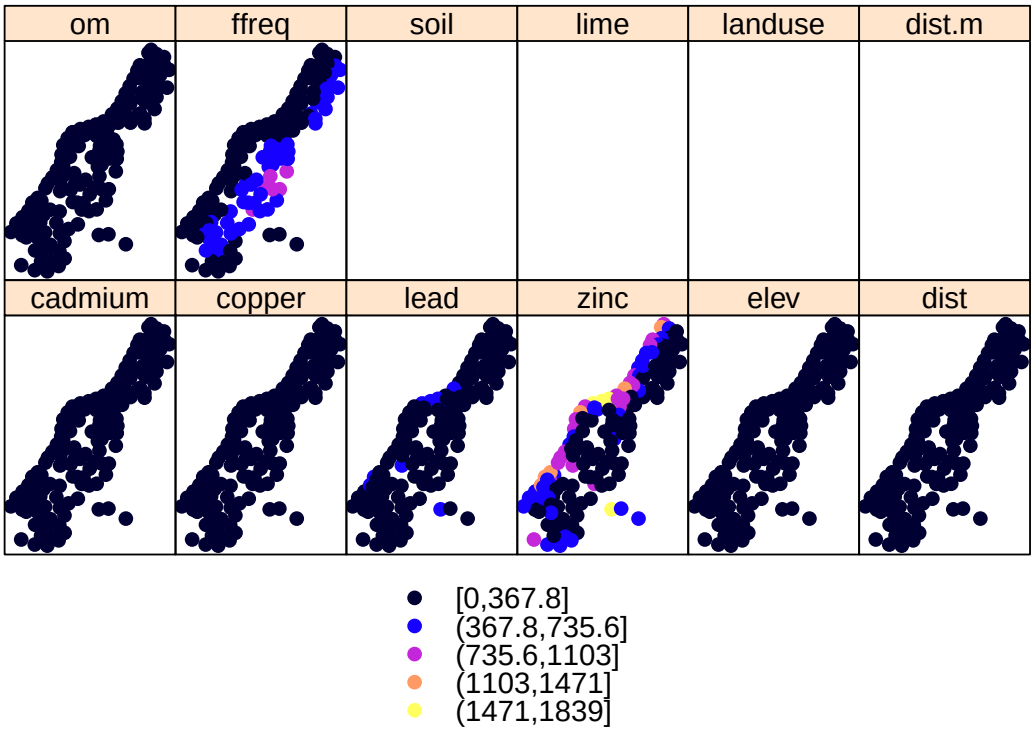
```
loglik(S_meuse_notCov) # probabilidad condicionada
```

```
## [1] -100.9408
```

```
coef(S_meuse_notCov) # coeficientes de regresión estimados
```

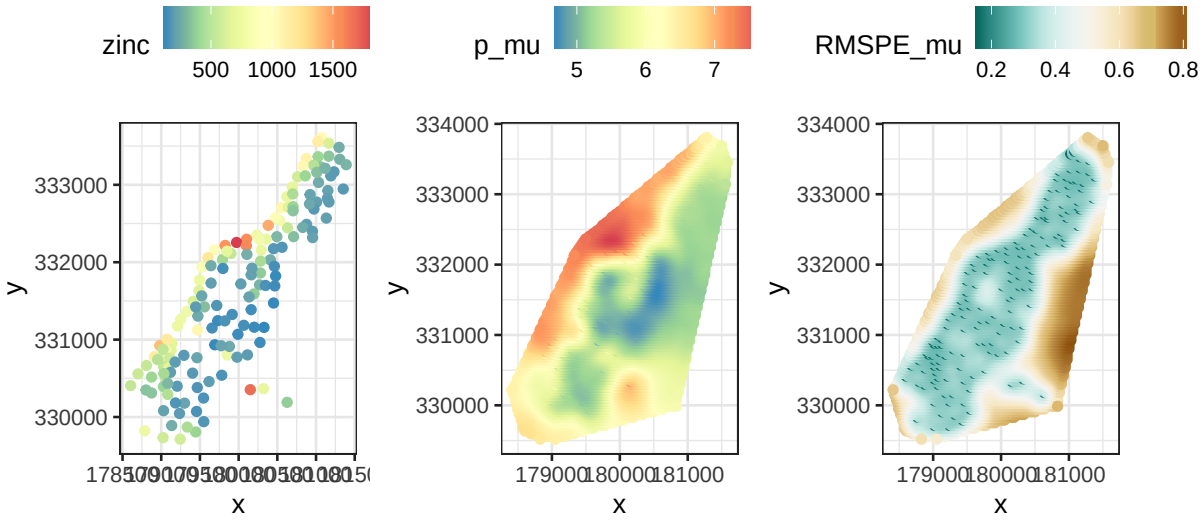
```
## Intercept
##  5.922536
```

```
spplot(meuse)
```



Ahora se observan las estimaciones calculadas y se realiza la predicción de las localizaciones y la predicción de los errores estándars.
Es importante añadir el argumento obs_fs, con el cual se podrá elegir la estimación suponiendo $\sigma^2_\varepsilon = 0$ dando el valor FALSE (caso que se desarrolla en la teoría anterior) o suponiendo $\sigma^2_\xi = 0$ dando el valor TRUE.

```
Pred_meuse_notCov = predict(S_meuse_notCov, obs_fs=F, type = c("link", "mean"))
# summary(Pred_meuse_notCov)

plotlist = plot(S_meuse_notCov, Pred_meuse_notCov)
ggpubr::ggarrange(plotlist = plotlist, nrow = 1, align = "hv", legend = "top")
```



4.2.1.2. Con covarianzas

Se considera como covarianza la raíz cuadrada de la distancia desde el centroide de la BAU hata el punto más cercano del río Meuse.

Se utiliza la función BAUs_from_points para convertir un objeto de clase 'SpatialPointsDataFrame' en un objeto de clase 'SpatialPolygonsDataFrame' construyendo una BAU alrededor de cada punto espacial.

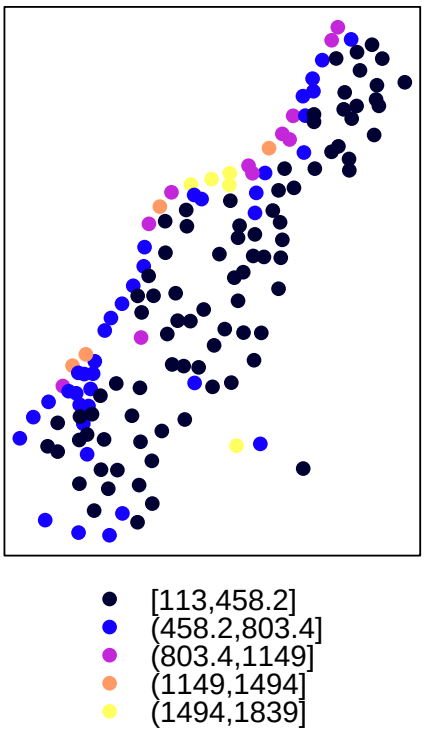
Se supone que las 10 primeras observaciones del conjunto de datos meuse están perdidas.

```
data("meuse")
meuse[1:10, "zinc"] = NA

meuseNew = subset(meuse, !is.na(zinc))
meuseNew = meuseNew[, c("x", "y", "zinc")]
coordinates(meuseNew) = ~x + y

meuse$zinc = NULL
```

```
coordinates(meuse) = c("x", "y")
meuse.gridNew = BAUs_from_points(meuse)
class(meuseNew)
```

```
## [1] "SpatialPointsDataFrame"
## attr(,"package")
## [1] "sp"
```

```
spplot(meuseNew)
```



Ahora se procede a calcular el objeto FRK que contiene el modelo SRE con las estimaciones y se predice.

```
f_meuse_Cov = log(zinc) ~ 1 + sqrt(dist)
S_meuse_Cov = FRK(f = f_meuse_Cov, data = list(meuseNew),
                  BAUs = meuse.gridNew, regular = 0)
```

```
## Assuming fine-scale variation is homoscedastic
## Generating basis functions...
## Automatically choosing number of functions...
## Modelling using 390 basis functions.
## Constructing SRE model...
## Averaging over polygons...
## Normalising basis function evaluations at BAU level...
```

```
## Fitting variogram for estimating measurement error...
## Binning data...
## Binned data in 0.61 seconds
## Fitting SRE model...
##   |                                                                      |
```

```
# summary(S_meuse_Cov)

Pred_meuse_Cov = predict(S_meuse_Cov, obs_fs = FALSE,
                         type = c("link", "mean"))

plotlist_Cov = plot(S_meuse_Cov, Pred_meuse_Cov)
#ggpubr::ggarrange(plotlist = plotlist_Cov, nrow = 1)
```

4.2.2. Método 2

Para la implementación del segundo método descrito se va a usar un conjunto de datos almacenado en el paquete FRK.
Este conjunto es 'AIRS_05_2003', que contiene la medición de dióxido de carbono procedente de la Sonda Infrarroja Atmosférica de la NASA durante el 1 de mayo de 2003 y el 15 de mayo de 2003. Las mediciones se llevaron a cabo a las 1:30pm entre 60 grados sur y 90 grados norte.

Se va a implementar ambos métodos en el plano y en la esfera, superficies permitidas en el paquete FRK.

4.2.2.1. Sobre superficie plana

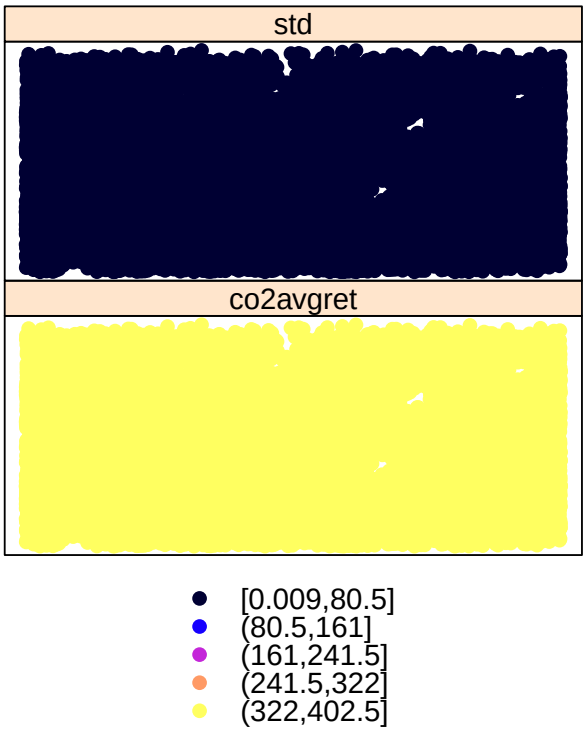
- **Primer paso**
  Se cargan los datos y se realizan algunas transformaciones.
  Se va a tomar los tres primeros días del conjunto de datos.
  Por comodidad, se renombra la columna co2std a std.
  Por último, se toman las columnas relevantes para el ejemplo y se asigna el sistema de referencia de coordenadas.

```
data(AIRS_05_2003) # se cargan los datos

AIRS_05_2003 =
dplyr::filter(AIRS_05_2003,day %in% 1:3) %>%
  # se toman los tres primeros días
dplyr::mutate(std=co2std) %>%
  # se renombra la columna co2std
dplyr::select(lon,lat,co2avgret,std)
# se toma las columnas que son de interés
```

```
coordinates(AIRS_05_2003) = ~lon+lat
# se convierte a objeto sp espacial
class(AIRS_05_2003)
```

```
## [1] "SpatialPointsDataFrame"
## attr(,"package")
## [1] "sp"
```

```
spplot(AIRS_05_2003)
```

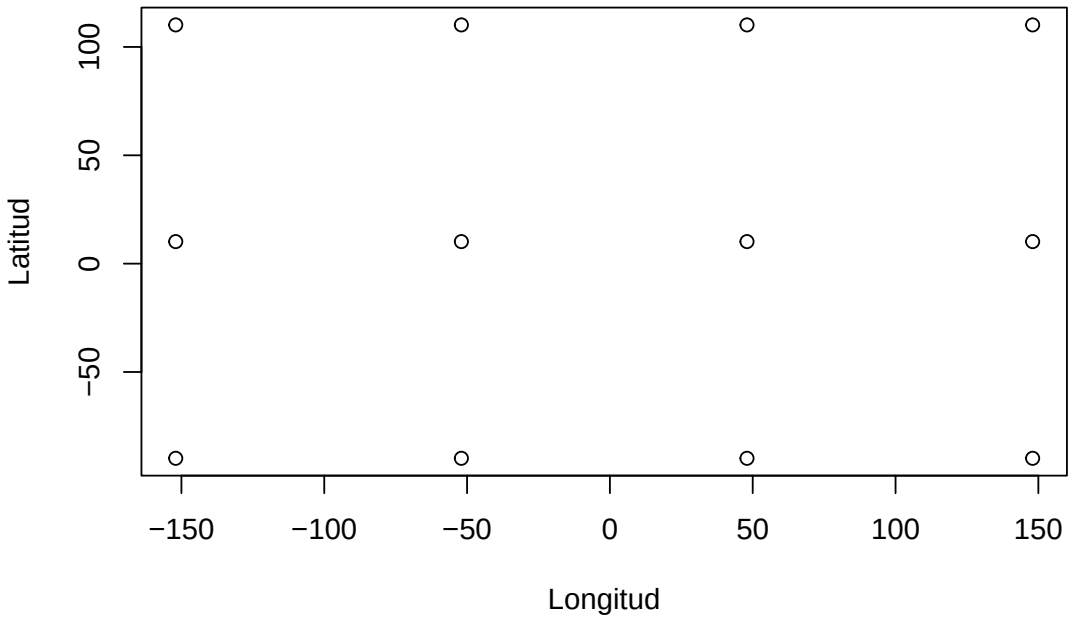


- **Segundo paso**
  Con la función auto_BAUs se generan las BAUs sobre el plano. Se crea una cuadrícula o malla de tipo rectangular con el argumento type="grid" y especificando el tamaño de cada celda con el argumento cellsize.

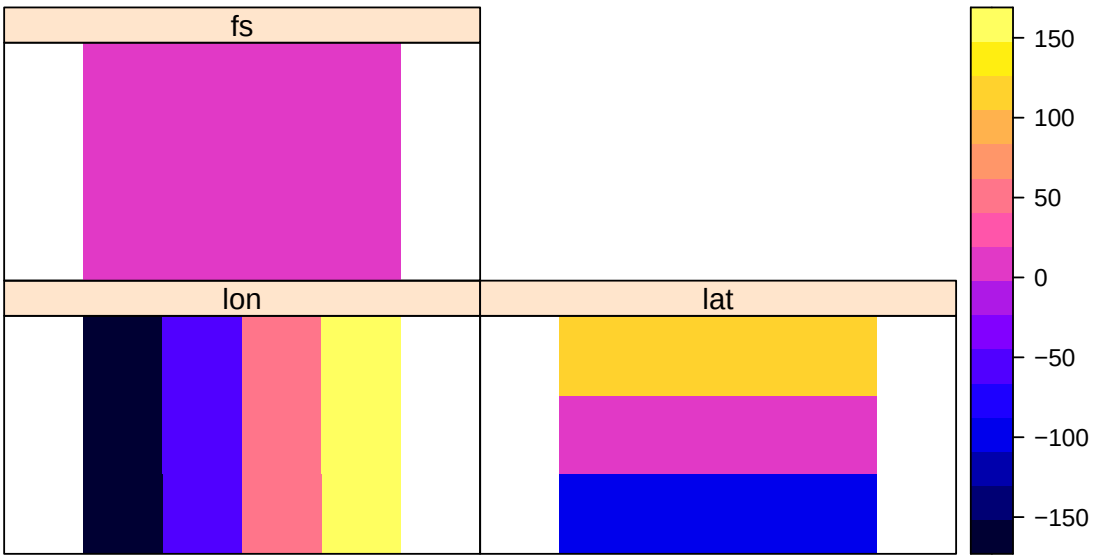
```
BAUs_AIRS_plane = auto_BAUs(manifold =plane(),
                            cellsize=c(100,100),
type = "grid", # malla hexagonal
data = AIRS_05_2003,
convex=-0.05, # border buffer factor
nonconvex_hull=FALSE)
class(BAUs_AIRS_plane)
```

```
## [1] "SpatialPixelsDataFrame"
## attr(,"package")
## [1] "sp"
```

```
BAUs_AIRS_plane$fs = 1

plot(BAUs_AIRS_plane$lon,BAUs_AIRS_plane$lat,
     ylab="Latitud", xlab="Longitud")
```




```
spplot(BAUs_AIRS_plane)
```

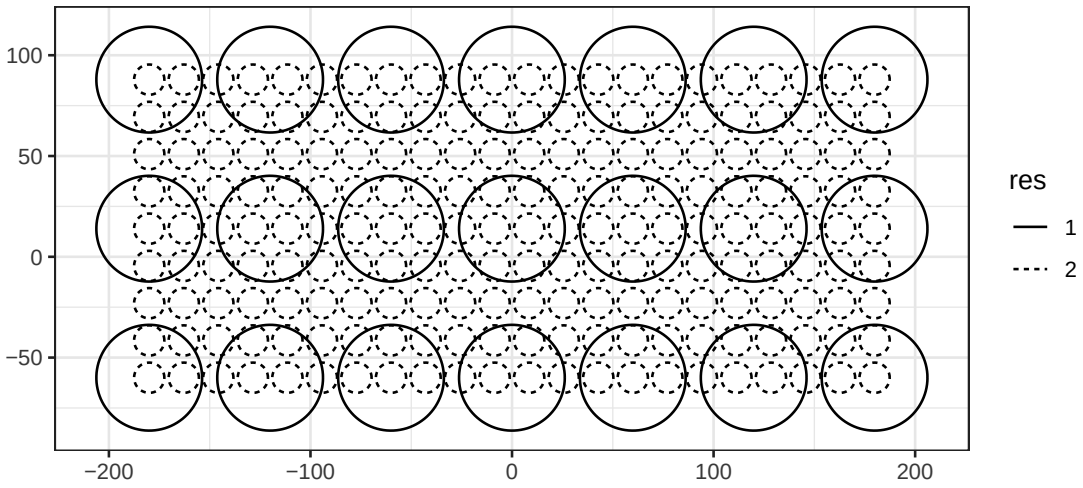


- **Tercer paso**
  Ahora se van a construir las funciones básicas espaciales.
  Para las funciones básicas ya se ha comentado que existen varios tipos: cuadradas, gaussianas, exponenciales,...). En este desarrollo se tomarán funciones básicas gaussianas:

```
BF_AIRS_plane = auto_basis(manifold = plane(),
data=AIRS_05_2003, # conjunto de datos
nres = 2, # número de resoluciones
type = "Gaussian") # función bicuadrada
class(BF_AIRS_plane)
```

```
## [1] "Basis"
## attr(,"package")
## [1] "FRK"
```

Para representarlas se utiliza el comando show_basis. En el caso del plano sólo soporta las funciones básicas de forma local. Cada función básica se muestra como un círculo con el diámetro igual al parámetro de escala de la función. El tipo de línea muestra la resolución.



- **Cuarto paso**
  Para el cálculo del modelo se usa la formula tal que la variable cov2avgret, que contiene la cantidad de CO2, tiene un gradiente latitudinal, por tanto se toma latitud como covariable.

Se especifica también la opción est_error = FALSE ya que el error viene suministrado por los datos.

Como el conjunto de datos que se tiene es bastante grande, puede darse que diferentes puntos se encuentren en la misma BAU. Para esto, se utiliza el argumento average_in_BAU = TRUE, que calcula la media de cada BAU.

```
f_AIRS = co2avgret ~ lat + 1 # fórmula

S_AIRS_plane = SRE(f = f_AIRS, # fórmula
list(AIRS_05_2003),
basis = BF_AIRS_plane,
# funciones básicas espaciales
BAUs = BAUs_AIRS_plane, # BAUs
est_error = FALSE,
# no se estima el error porque viene inducido por el modelo
average_in_BAU = TRUE)
```

```
## Normalising basis function evaluations at BAU level...
```

```
## Binning data...
## Binned data in 0.27 seconds
```

- **Quinto paso**
  Se estiman los parámetros del modelo con el comando SRE.fit, usando el algoritmo EM como vimos en el marco teórico del modelo.

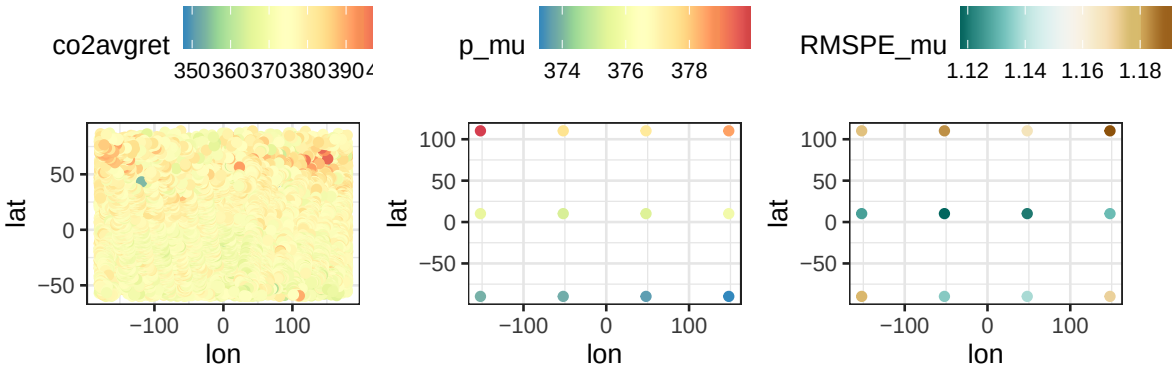
Se especifica con el argumento n_EM que el número máximo de iteraciones debe ser 1.

```
S_AIRS_plane = SRE.fit(S_AIRS_plane, # SRE model
n_EM = 1, # máximo número de iteraciones para el algoritmo EM
tol = 0.01,
print_lik=FALSE) # no se imprime la probabilidad en cada iteración
```

```
##   |
## Maximum EM iterations reached
```

- **Sexto paso**
  Por último se predice.

```
Pred_AIRS_plane = predict(S_AIRS_plane, type = c("link", "mean"))

plotlist_AIRS = plot(S_AIRS_plane, Pred_AIRS_plane)
ggpubr::ggarrange(plotlist = plotlist_AIRS,
                  nrow = 1, align = "hv", legend = "top")
```



4.2.2.2. Sobre superifcie esférica

- **Primer paso**
  Se utilizan los datos del ejemplo anterior, cargados de nuevo, sobre la superficie plana pero tomando el sistema de referencia de coordenadas sobre la esfera.

```
data("AIRS_05_2003")

AIRS_05_2003 =
dplyr::filter(AIRS_05_2003,day %in% 1:3) %>%
# se toman 3 días
dplyr::mutate(std=co2std) %>%
# se renombra la variable co2std
dplyr::select(lon,lat,co2avgret,std)
# se seleccionan las columnas que se necesitan
coordinates(AIRS_05_2003) = ~lon+lat
# se convierte en un objeto de tipo espacial

proj4string(AIRS_05_2003) = CRS("+proj=longlat +ellps=sphere")
# coordenadas sobre la esfera
```

- **Segundo paso**
  Se va a construir las BAUs sobre la esfera utilizando la función auto_BAUs. En dicha función existe un argumento isea3h_res que indica el número de resolución de las celdas del Sistema de Mallas Globales y Discretas, que oscila en este paquete entre 0 y 6.
  Para resoluciones mayores habría que utilizar el paquete ddgrids, que sale del alcance de este documento.

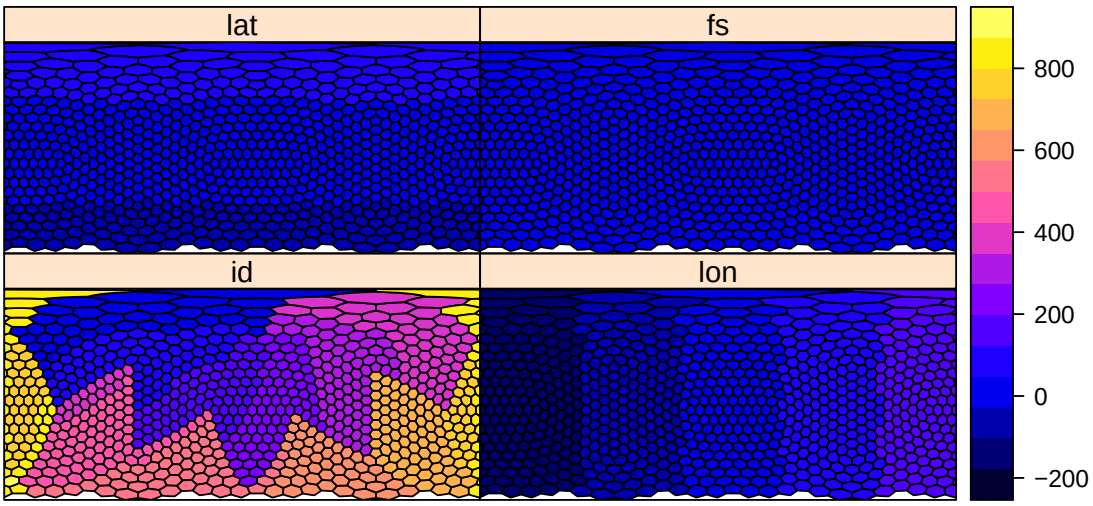
El DGGRID o Sistema de Mallas Globales y Discretas fue desarrollado por el OGC (Open Geospatial Consortium) con el objetivo de mejorar la forma de georreferenciar datos geográficos sin tener que toparse con sistemas de coordenadas proyectadas.

Existen varias opciones para elegir la malla, aunque en la función auto_BAUs utiliza ISEA3H (Icosahedral Snyder Equal Area Aperture 3 Hexagonal Grid), cuadrícula o malla de tipo hexagonal sobre la esfera.

```
BAUs_AIRS_sphere = auto_BAUs(manifold =sphere(), # espacio
isea3h_res = 4, # número de resolución de las BAUs máximo
type = "hex", # malla hexagonal
data = AIRS_05_2003)

BAUs_AIRS_sphere$fs = 1 # fine-scale component
class(BAUs_AIRS_sphere)
```

```
## [1] "SpatialPolygonsDataFrame"
## attr(,"package")
## [1] "sp"
```

```
spplot(BAUs_AIRS_sphere)
```

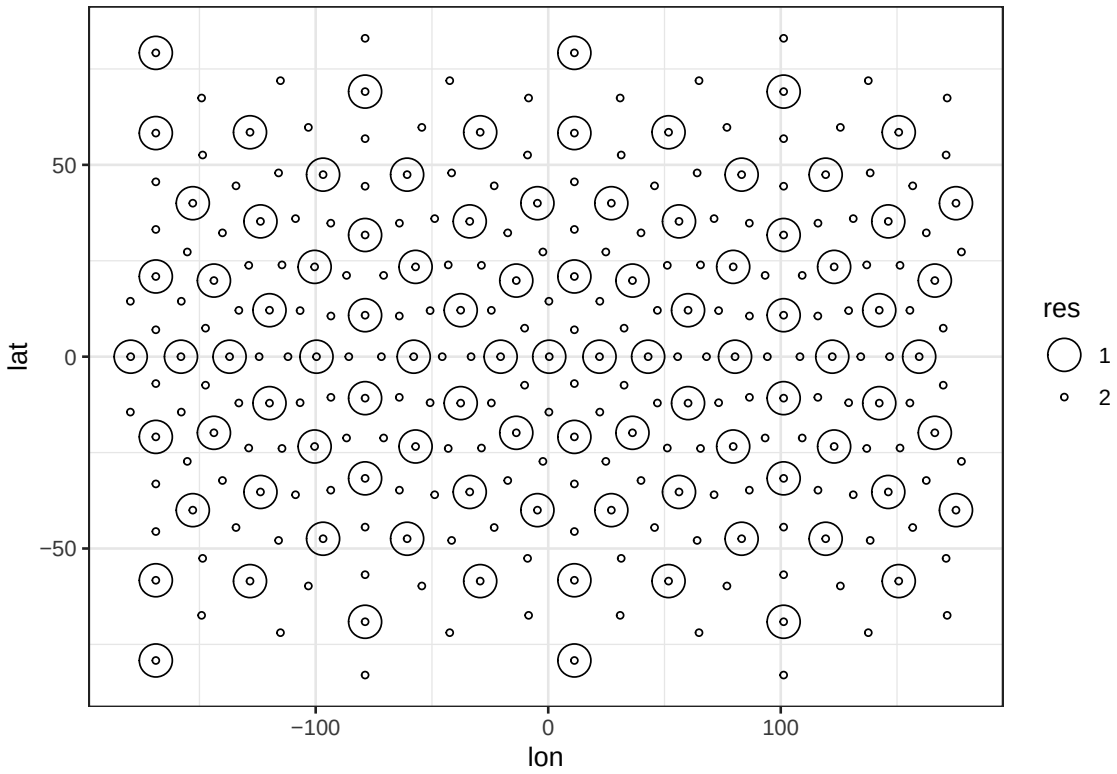


- **Tercer paso**
  De nuevo como en el caso anterior se generan las funciones básicas espaciales gaussianas.

```
BF_AIRS_sphere = auto_basis(manifold = sphere(),
data=AIRS_05_2003, # conjunto de datos
nres = 2, # número de resoluciones
type = "Gaussian") # función básica gaussiana
```

De nuevo se usa show_basis. En el caso de la esfera, los centros de las funciones básicas se muestran como círculos con tamaños correspondientes a su resolución.

```
basis_s_plot = show_basis(BF_AIRS_sphere) + xlab("lon") + ylab("lat")
basis_s_plot
```



- **Cuarto paso**
  Se utiliza la fórmula del apartado anterior y se genera de nuevo el modelo.

```
S_AIRS_sphere = SRE(f = f_AIRS, # fórmula
list(AIRS_05_2003),
basis = BF_AIRS_sphere, # funciones básicas espaciales
BAUs = BAUs_AIRS_sphere, # BAUs
est_error = FALSE,
# no se estima el error porque viene inducido por el modelo
average_in_BAU = TRUE)
```

```
## Normalising basis function evaluations at BAU level...
## Binning data...
## Binned data in 2.21 seconds
```

```
S_AIRS_sphere
```

```
## Formula: co2avgret ~ lat + 1
## Assumed response distribution: gaussian
## Specified link function: identity
## Method of model fitting:
## Number of datasets: 1
## Number of basis functions: 364
## Class of basis functions: Basis
## Number of BAUs [extract using object@BAUs]:  793
```

```
## Number of observations [extract using object@Z]:  788
## Mean obs. variance at BAU level [extract using object@Ve]: 1.529667
## Fine-scale variance proportionality constant [extract using object@sigma2fshat]: 0
## Dimensions of C in Z = C*Y + e [extract using object@Cmat]:  c(788L, 793L)
## Dimensions of S in Y = X*alpha + S*eta + delta [extract using object@S]:  c(788L,
## Number of covariates: 2
```

```
# se observa como ha generado 364 funciones básicas espaciales y 2348 BAUs
```

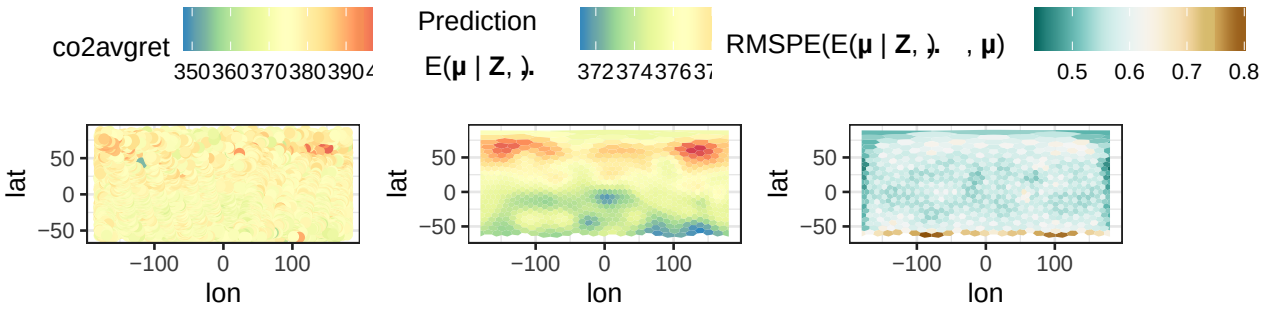
- **Quinto paso**
  Se estiman los parámetros del modelo con el comando SRE.fit y con número de iteraciones del algoritmo EM de nuevo 1.

```
S_AIRS_sphere = SRE.fit(S_AIRS_sphere, # SRE model
n_EM = 1, # máximo número de iteraciones para el algoritmo EM
tol = 0.01,
print_lik=FALSE) # no se imprime la probabilidad en cada iteración
```

```
##   |                                                                      |
## Maximum EM iterations reached
```

- **Sexto paso**
  Por último se predice.

```
Pred_AIRS_sphere = predict(S_AIRS_sphere, type = c("link", "mean"))

Pred_AIRS_sphere_df = as(Pred_AIRS_sphere,"data.frame")

plotlist_AIRS_sphere = plot(S_AIRS_sphere, Pred_AIRS_sphere)
ggpubr::ggarrange(plotlist = plotlist_AIRS_sphere,
                  nrow = 1, align = "hv", legend = "top")
```



4.3. Implementación datos espacio temporales

Aunque FRK fue diseñado principalmente para modelar y predecir datos espaciales, también se puede implementar en datos espacio temporales, como se comentó anteriormente. Aún así, este paquete se encuentra limitado con respecto a la implementación de los datos espacio temporales. En particular, no es posible un cambio temporal del soporte y la medida del error estándar no está implementado.

En el paquete FRK la dimensión temporal se refleja en las funciones básicas, ya que se construyen tanto temporales como espaciales y se modelan conjuntamente mediante la función TensorP, como se ve en el ejemplo a continuación.

- **Primer paso**
  Se renombra la columna std a co2std, se toma una muestra aleatoria de 2000 puntos y se crea el campo temporal.

```
data(AIRS_05_2003)

set.seed(1)
AIRS_05_2003 = mutate(AIRS_05_2003, std=co2std) %>%
# se renombra la columna std
sample_n(20000)
# muestra aleatoria de 20000 puntos

AIRS_05_2003 = within(AIRS_05_2003,
```

```
{time = as.Date(paste(year,month,day,sep="-"))})
# se genera el campo temporal
```

Con todo esto, se genera el objeto espacio temporal.

```
STObj = stConstruct(x = AIRS_05_2003,
space = c("lon","lat"), # datos espaciales
time ="time", # datos temporales
crs = CRS("+proj=longlat +ellps=sphere"),
# sistema de referencia de coordenadas
interval=TRUE)
# los datos temporales se agrupan en intervalos

class(STObj) # STIDF
```

```
## [1] "STIDF"
## attr(,"package")
## [1] "spacetime"
```

```
STObj$std = 2
```

- **Segundo paso**
  Se crean las BAUs.

```
grid_BAUs = auto_BAUs(manifold=STsphere(),
# datos espacio temporales en la esfera
data=time(STObj), # datos temporales
cellsize = c(5,5,1),
type="grid",
tunit = "days") # los datos temporales se espacian en días

grid_BAUs$fs = 1
```

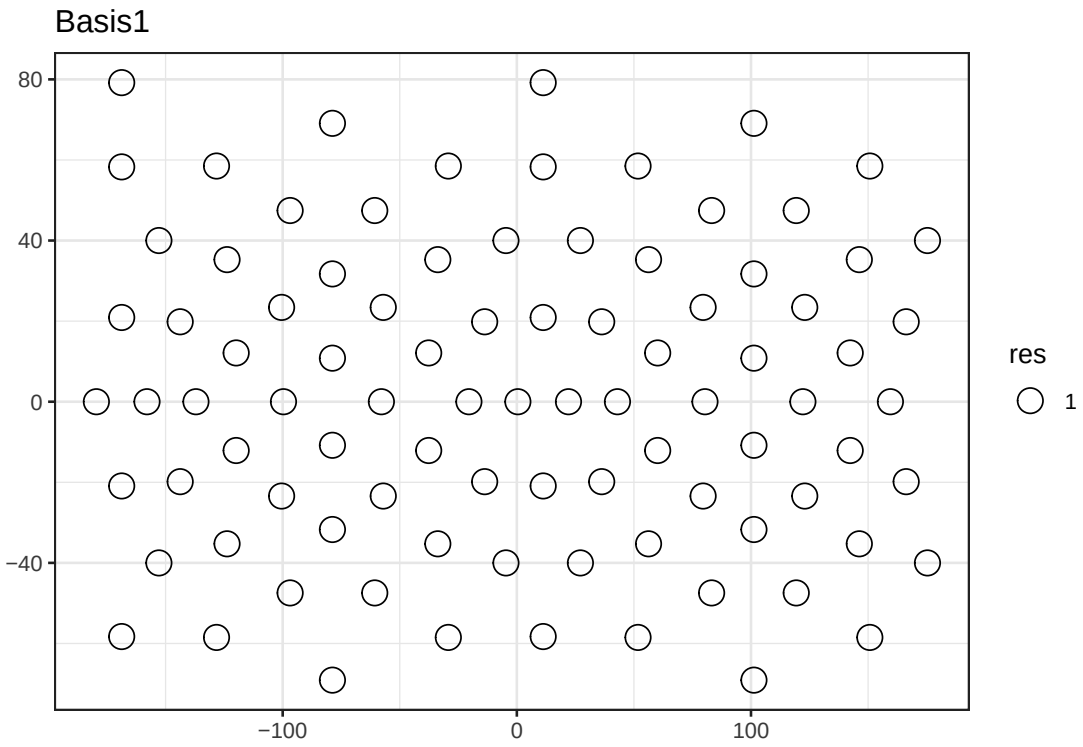
- **Tercer paso**
  Se generan las funciones espaciales básicas y las temporales, y para generar las espacio temporales utilizamos la función TensorP.

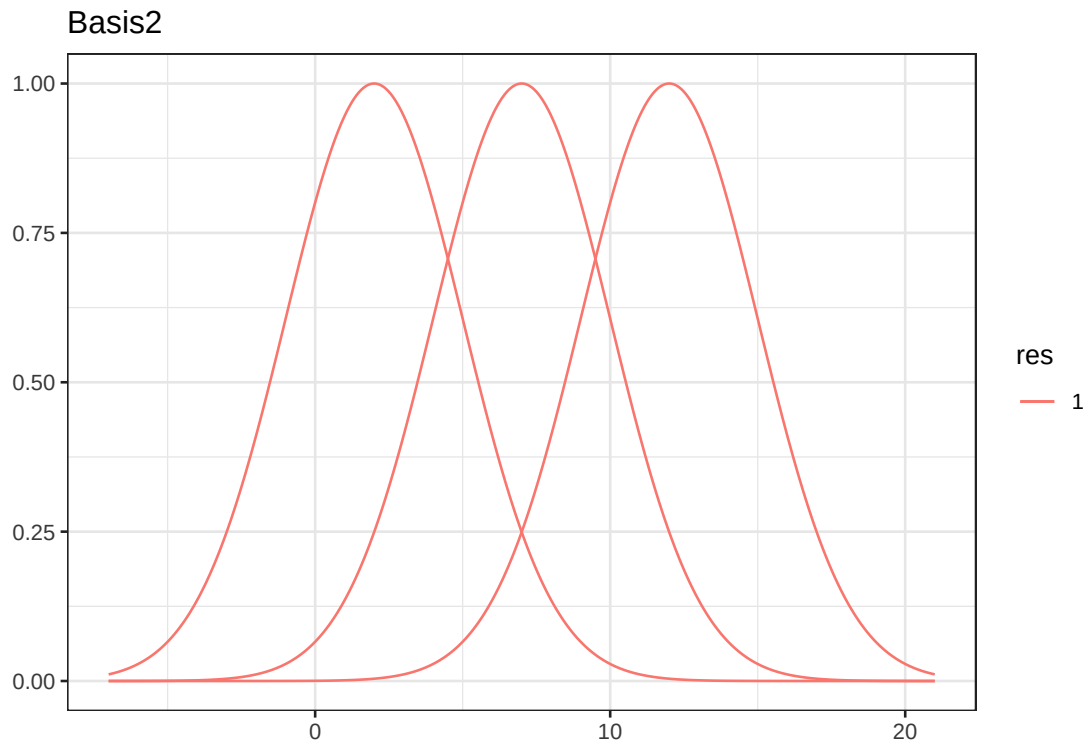
```
G_spatial = auto_basis(manifold = sphere(),
data=as(STObj,"Spatial"),
nres = 1, # número de resoluciones
prune= 15,
type = "Gaussian", #funciones básicas gaussianas
subsamp = 2000,
isea3h_lo = 2)
```

```
G_temporal = local_basis(manifold=real_line(), # funciones en los reales
loc = matrix(c(2,7,12)),
scale = rep(3,3),
type = "Gaussian")

G_spacetime = TensorP(G_spatial,G_temporal)
```

En este caso, al ser las funciones básicas espacio temproales de la clase 'TensorP_Basis', se visualizan en dos gráficos separados: uno para las funciones básicas espaciales, como se vio antes, y otro para las funciones básicas temporales.

```
show_basis(G_spacetime)
```





- **Cuarto paso**
  Se utiliza la misma función f que en el caso espacial.

```
f_st = co2avgret ~ lat +1

S_AIRS_st = SRE(f = f_st, # fórmula
data = list(STObj), # objeto espacio temporal
basis = G_spacetime, # funciones básicas espacio temporales
BAUs = grid_BAUs, # BAUs espacio temporales
est_error = FALSE, # no se estima el error estándar
average_in_BAU = TRUE)
```

```
## Normalising basis function evaluations at BAU level...
## Binning data...
## Binned data in 1.5 seconds
## Binned data in 1.75 seconds
## Binned data in 1.71 seconds
## Binned data in 1.76 seconds
## Binned data in 1.95 seconds
## Binned data in 1.53 seconds
## Binned data in 1.64 seconds
## Binned data in 1.61 seconds
## Binned data in 1.87 seconds
## Binned data in 1.77 seconds
## Binned data in 1.36 seconds
## Binned data in 1.4 seconds
```

```
## Binned data in 1.39 seconds
## Binned data in 1.58 seconds
## Binned data in 1.38 seconds
```

```
# Media de datos en el interior de las BAUs
```

- **Quinto paso**
  Se predicen los parámetros mediante el algoritmo EM.

```
S_AIRS_st = SRE.fit(S_AIRS_st, #
n_EM = 1, # máximo número de iteraciones para el algoritmo EM
tol = 0.01) # criterio de convergencia
```

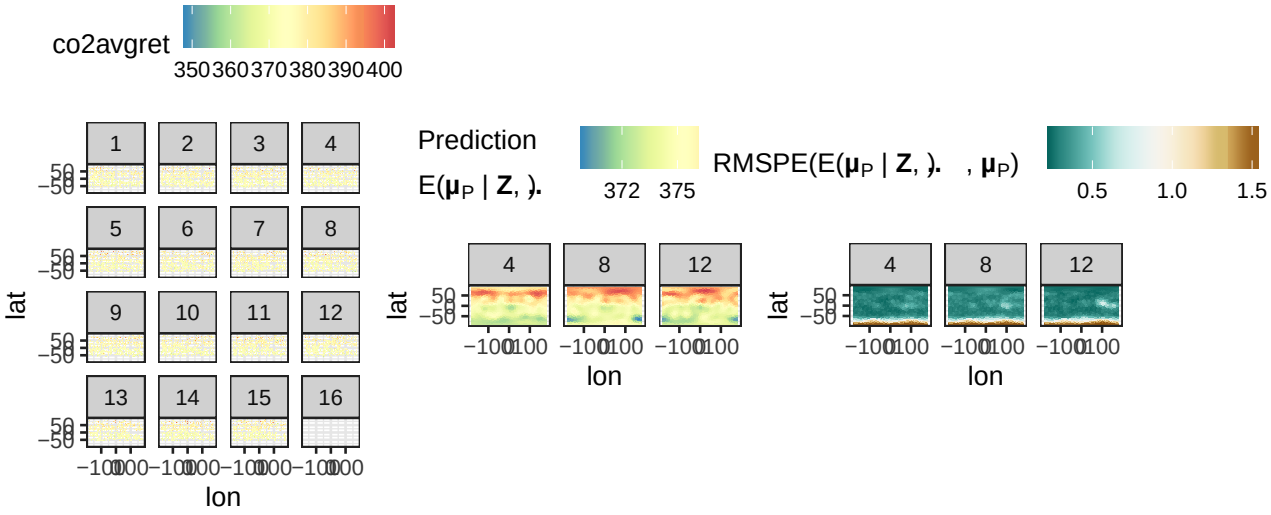
```
##   |                                                                      |
## Maximum EM iterations reached
```

- **Sexto paso**
  Predicción de los días concretos.

```
Pred_AIRS_st = predict(S_AIRS_st, obs_fs = TRUE,
pred_time = c(4L,8L,12L), type = c("link", "mean")) # días donde se predice

Pred_AIRS_st_df = as(Pred_AIRS_st,"data.frame")

plotlist_AIRS_st = plot(S_AIRS_st, Pred_AIRS_st)
```

```
## To plot the STIDF data provided in the SRE object, we use the binned data in objec
## NA values detected in the data, which will be transparent in the final plot. If yo
##
```

```
ggpubr::ggarrange(plotlist = plotlist_AIRS_st,
                  nrow = 1, align = "hv", legend = "top")
```


co2avgret
350 360 370 380 390 400
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
lat
lon
Prediction
E(μP | Z,).
372 375
RMSPE(E(μP | Z,). , μP)
0.5 1.0 1.5
4 8 12
lat
lon

Bibliografía

JJ Allaire, Yihui Xie, Jonathan McPherson, Javier Luraschi, Kevin Ushey, Aron Atkins, Hadley Wickham, Joe Cheng, Winston Chang, and Richard Iannone. *rmarkdown: Dynamic Documents for R*, 2022. URL https://CRAN.R-project.org/package=rmarkdown. R package version 2.14.

Hajer Braham, Sana Ben Jemaa, Gersende Fort, Eric Moulines, and Berna Sayrac. Fixed rank kriging for cellular coverage analysis. *IEEE Transactions on Vehicular Technology*, 66, 05 2015. doi: 10.1109/TVT.2016.2599842.

Chris Brunsdon and Lex Comber. *An Introduction to R for Spatial Analysis and Mapping*. SAGE Publications, first edition edition, 2015.

Patricia Carracedo and A. Debón. Selección de modelos espacio-temporales con datos de panel en matlab y r. *Revista Electrónica de Comunicaciones y Trabajos de ASEPUMA*, 18:93–118, 12 2017. doi: 10.24309/recta.2017.18.2.01.

Noel Cressie and Gardar JohannessonSourcer. Fixed rank kriging for very large spatial data sets. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 70(1):209–226, 2008.

Noel Cressie, Tao Shi, and Emily Kang. Fixed rank filtering for spatio-temporal data. *Journal of Computational and Graphical Statistics*, 19:724–745, 01 2010. doi: 10.2307/25765367.

Paula García. "£qué es un sig, gis o sistema de información geográfica?". Disponible en https://geoinnova.org/blog-territorio/que-es-un-sig-gis-o-sistema-de-informacion-geografica/, 2021.

María Dolores Jiménez Gamero. *Apuntes de la asignatura Series Temporales*. Documento inédito, sevilla: facultad de matemáticas, universidad de sevilla edition, 2020.

M. Katzfuss and N. Cressie. Tutorial on fixed rank kriging (frk) of co2 data. *Proceedings of technical report, report no*, 01 2011.

Pedro L. Luque-Calvo. *Escribir un Trabajo Fin de Estudios con R Markdown*, 2017.

Pedro L. Luque-Calvo. *Cómo crear Tablas de información en R Markdown*, 2019.

Antonio López-Quílez and Facundo Muñoz. Review of spatio-temporal models for disease mapping. 01 2009.

Jean Mas. *Análisis espacial con R: Usa R como un Sistema de Información Geográfica*. 08 2018. ISBN 978-608-4642-66-4.

José Montero and Gema Fernández-Avilés. Functional kriging prediction of pollution series: the geostatistical alternative for spatially-fixed. *Revista de Estudios de Economía Aplicada*, 31:145–179, 01 2015.

Songthip Ounpraseuth. Gaussian processes for machine learning. carl edward rasmussen and christopher k. i. williams. *Journal of the American Statistical Association*, 103, 03 2008. doi: 10.2307/27640057.

Edzer Pebesma. Spacetime: Spatio-temporal data in r. *Journal of Statistical Software*, 51:1–30, 11 2012. doi: 10.18637/jss.v051.i07.

Edzer Pebesma. *sf: Simple Features for R*, 2022a. URL https://CRAN.R-project.org/package=sf. R package version 1.0-7.

Edzer Pebesma. *spacetime: Classes and Methods for Spatio- Temporal Data*, 2022b. URL https://github.com/edzer/spacetime. R package version 1.2-6.

Edzer Pebesma and Roger Bivand. *sp: Classes and Methods for Spatial Data*, 2022. URL https://CRAN.R-project.org/package=sp. R package version 1.4-7.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2016. URL https://www.R-project.org/.

RStudio Team. *RStudio: Integrated Development Environment for R*. RStudio, Inc., Boston, MA, 2015. URL http://www.rstudio.com/.

M. Ugarte. A brief introduction to spatio-temporal modelling. *BEIO, Boletín de Estadística e Investigación Operativa, ISSN 1699-8871, Vol. 24, Nž. 2, 2008, pags. 5-10*, 01 2008.

Hadley Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2016. ISBN 978-3-319-24277-4. URL https://ggplot2.tidyverse.org.

Hadley Wickham and Garrett Grolemund. *R for Data Science: Import, Tidy, Transform, Visualize, and Model Data*. O'Reilly, first edition edition, 2017.

Hadley Wickham, Winston Chang, Lionel Henry, Thomas Lin Pedersen, Kohske Takahashi, Claus Wilke, Kara Woo, Hiroaki Yutani, and Dewey Dunnington. *ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics*, 2022a. URL https://CRAN.R-project.org/package=ggplot2. R package version 3.3.6.

Hadley Wickham, Romain François, Lionel Henry, and Kirill Müller. *dplyr: A Grammar of Data Manipulation*, 2022b. URL https://CRAN.R-project.org/package=dplyr. R package version 1.0.9.

Yihui Xie. knitr: A comprehensive tool for reproducible research in R. In Victoria Stodden, Friedrich Leisch, and Roger D. Peng, editors, *Implementing Reproducible Computational Research*. Chapman and Hall/CRC, 2014. URL http://www.crcpress.com/product/isbn/9781466561595. ISBN 978-1466561595.

Yihui Xie. *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition, 2015. URL https://yihui.org/knitr/. ISBN 978-1498716963.

Yihui Xie. *knitr: A General-Purpose Package for Dynamic Report Generation in R*, 2022. URL https://yihui.org/knitr/. R package version 1.39.

Yihui Xie, J.J. Allaire, and Garrett Grolemund. *R Markdown: The Definitive Guide*. Chapman and Hall/CRC, Boca Raton, Florida, 2018. URL https://bookdown.org/yihui/rmarkdown. ISBN 9781138359338.

Yihui Xie, Christophe Dervieux, and Emily Riederer. *R Markdown Cookbook*. Chapman and Hall/CRC, Boca Raton, Florida, 2020. URL https://bookdown.org/yihui/rmarkdown-cookbook. ISBN 9780367563837.

Andrew Zammit-Mangion and Noel Cressie. Frk: An r package for spatial and spatio-temporal prediction with large datasets. *Journal of Statistical Software*, 98, 05 2017. doi: 10.18637/jss.v098.i04.

Andrew Zammit-Mangion and Matthew Sainsbury-Dale. *FRK: Fixed Rank Kriging*, 2022. URL https://CRAN.R-project.org/package=FRK. R package version 2.0.5.