Modelización Bayesiana de datos de patrones puntuales espacio-temporales

Álvaro Briz Redón

Departamento de Estadística e Investigación Operativa
Universitat de València

alvaro.briz@uv.es

6 y 7 de mayo de 2025, Bogotá, D.C.
IV Congreso Colombiano de Estadística

VNIVERSITAT DE VALÈNCIA

Tabla de contenidos

Objetivos del cursillo

Los principales objetivos del cursillo son los siguientes:

- Proporcionar una introducción a la modelización de patrones puntuales espacio-temporales.
- Proporcionar ejemplos de uso de un paquete de R, `nimble`, como herramienta para realizar inferencia Bayesiana sobre este tipo de modelos.
- Proporcionar diversos ejemplos de modelos para patrones puntuales espacio-temporales.

NO son objetivos del cursillo:

- Profundizar sobre procesos puntuales.
- Profundizar sobre inferencia Bayesiana.

Repositorio asociado al cursillo

El material del cursillo, incluyendo estas diapositivas y múltiples códigos en R, se encuentra en el siguiente **repositorio de GitHub**:

`https://github.com/albrizre/Course-IV-CCE`

En próximos días continuaré actualizando y depurando el repositorio. Si hace falta cualquier aclaración, o detectas algún error, no dudes en contactarme.

¿Qué es un proceso puntual?

- Es un proceso estocástico que describe la ocurrencia de puntos (*eventos*) sobre un espacio $W \subset \mathbb{R}^d$.
- Una realización de un proceso puntual es un conjunto de puntos $\boldsymbol{x} = \{x_i\}_{i=1}^n$, donde $x_i \in W$, el espacio de estudio.
- Cada realización del proceso puntual se denomina *patrón puntual* sobre W.

Algunos ejemplos

- Tiempos de llegada de clientes a una tienda ($W = [0, T]$).
- Ubicación de árboles en un bosque ($W \subset \mathbb{R}^2$).
- Localización espacio-temporal de eventos sísmicos ($W \subset \mathbb{R}^2 \times [0, T]$).

Algunos ejemplos

En general, se distinguen tres tipos de patrones puntuales según su configuración:

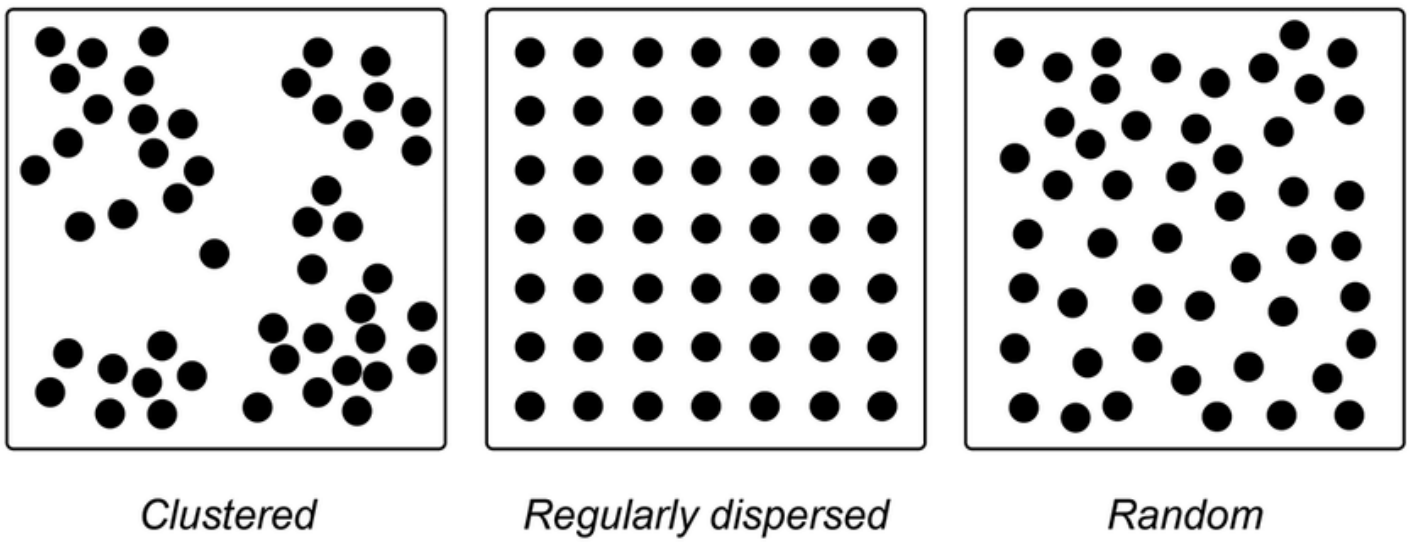


Figura: Los tres principales patrones puntuales. Extraído de Kempf (2020)

Métodos para procesos/patrones puntuales

Se disponen de numerosos métodos para analizar patrones puntuales:

- Técnicas no paramétricas para describir la función de intensidad del proceso, la cual tiene que ver con la *medida de primer orden* del proceso.
- Herramientas para el estudio de las *medidas de segundo orden* del proceso, las cuales permiten distinguir el tipo de patrón (clusterizado, regular o aleatorio). Existen múltiples alternativas, la mayoría de ellas de tipo funcional (Diggle, 2013): K-función, L-función, etc.
- Modelos paramétricos para describir la intensidad del proceso. Esto nos permite tener en cuenta tanto propiedades de primer orden como de segundo orden. **Nos centramos en este aspecto en este cursillo**.
- ...

Proceso puntual de Poisson homogéneo

Empezamos por el proceso puntual más simple: el proceso Poisson homogéneo. Presenta las siguientes características:

- Definido por un único parámetro, que representa que la *función de intensidad* es constante: $\lambda > 0$.
- Propiedades:
  - El número de puntos en una región $A \subset \mathbb{R}^d$ se distribuye según la siguiente distribución de Poisson:

    $$N(A) \sim \text{Poisson}(\lambda|A|),$$

    donde $|A|$ representa el volumen de A.
  - Independencia: Si A_1 y A_2 son regiones disjuntas ($A_1 \cap A_2 = \varnothing$), entonces $N(A_1)$ y $N(A_2)$ son independientes.

Función de intensidad

Hemos dicho que la función de intensidad en el proceso Poisson homogéneo es constante e igual a λ. **¿Qué es exactamente la función de intensidad?**

Sobre un espacio $W \subset \mathbb{R}^d$, la función de intensidad se define como:

$$\lambda(x) = \lim_{|\mathrm{d}x| \to 0} \frac{\mathbb{E}[N(\mathrm{d}x)]}{|\mathrm{d}x|}, \forall x \in W$$

En palabras, esto significa que el número de puntos esperados en una región infinitesimal alrededor de $x \in W$. Por tanto, si $\lambda(x) = \lambda$:

$$E(N(A)) = \lambda |A|, \forall A \subset W$$

Función de verosimilitud del proceso Poisson homogéneo

Dada una realización $\boldsymbol{x} = \{x_1, \ldots, x_n\}$ en una región W de volumen $|W|$, puede deducirse que la función de verosimilitud del proceso Poisson homogéneo es (Daley and Vere-Jones, 2002):

$$L(\lambda; \boldsymbol{x}) = \exp(-\lambda|W|) \cdot \lambda^n$$

Por tanto, la función de log-verosimilitud es:

$$\log L(\lambda; \boldsymbol{x}) = n \log \lambda - \lambda|W|$$

Puede comprobarse que el estimador de máxima verosimilitud para λ es:

$$\hat{\lambda}_{MLE} = \frac{n}{|W|}$$

Proceso puntual de Poisson inhomogéneo

En general, es poco realista pensar que la intensidad de un proceso es constante sobre todo el espacio W. El proceso Poisson inhomogéneo mantiene las dos propiedades básicas del Poisson homogéneo, pero ahora:

$$E(N(A)) = \int_A \lambda(x)dx$$

Si asumimos que la intensidad $\lambda(x)$ es paramétrica y dependiente de un vector de parámetros, θ, la función de verosimilitud es (Daley and Vere-Jones, 2002):

$$L(\theta; \boldsymbol{x}) = \exp\left(-\int_W \lambda(u)\, du\right) \prod_{i=1}^{n} \lambda(x_i)$$

Y la de log-verosimilitud:

$$\log L(\theta; \boldsymbol{x}) = \sum_{i=1}^{n} \log \lambda(x_i) - \int_W \lambda(u)\, du$$

Interpretación de la log-verosimilitud

Sobre la fórmula de la log-verosimilitud:

$$\log L(\theta; \boldsymbol{x}) = \sum_{i=1}^{n} \log \lambda(x_i) - \int_W \lambda(u)\, du$$

- La log-verosimilitud crece si $\lambda(x)$ es elevada en las localizaciones donde se ha observado un evento.
- La log-verosimilitud decrece si $\int_W \lambda(u)\, du$ es alta, es decir, se penaliza que la intensidad sea alta en toda la región.
- La integral $\int_W \lambda(u)\, du$ solo tendrá solución analítica si la forma en que depende $\lambda(x)$ de θ es sencilla...

Aproximación de la integral

En la práctica, la integral $\int_W \lambda(u)\, du$ se aproxima empleando una malla o cuadrícula sobre el espacio $W \subset \mathbb{R}^d$:

$$\int_W \lambda(u)\, du \approx \sum_{j=1}^{m} \lambda(u_j) \cdot w_j$$

donde u_j son puntos sobre la malla y w_j el peso de cada celda de la malla (volumen de la celda en $\mathbb{R}^d$).

Introducción a la inferencia Bayesiana

Veamos ahora una introducción minimalista a la inferencia Bayesiana. Existen múltiples libros que profundizan sobre esta materia, como el de Gelman et al. (2013).

Esencialmente, en inferencia Bayesiana debemos tener presente que **cada uno de los parámetros de un modelo sigue una cierta distribución de probabilidad**. En inferencia frecuentista, en cambio, se asume que los parámetros son valores constantes y desconocidos.

Para poder trabajar en inferencia Bayesiana, **debemos asignar**, *a priori*, **una distribución a cada uno de los parámetros implicados en el modelo**. Esta distribución se denomina *distribución prior* o *distribución previa*. El **objetivo** de la inferencia es **actualizar esta distribución en base a los datos**, dando lugar a la *distribución a posteriori*.

Potenciales ventajas de la inferencia Bayesiana

- Permite incorporar información previa: conocimiento experto, datos históricos, etc.
- Produce distribuciones completas, no solo estimadores puntuales.
- Flexible para modelos jerárquicos o estructuras complejas.

Desventaja: Computacionalmente costosa.

Fórmula de Bayes

La fórmula de Bayes es el resultado clave para llevar a cabo la inferencia Bayesiana. Si asumimos un parámetro θ y un conjunto de datos observado, D, la fórmula de Bayes nos permite escribir (sin entrar en detalles):

$$p(\theta|D) = \frac{p(D|\theta)p(\theta)}{p(D)},$$

donde cada elemento de la fórmula representa lo siguiente:

- $p(\theta|D)$: Distribución a posteriori de θ
- $p(D|\theta)$: Función de verosimilitud del modelo
- $p(\theta)$: Distribución previa de θ
- $p(D)$: Evidencia (constante de normalización)

Fórmula de Bayes

Obviando el término $p(D)$, que no depende de θ, se tiene:

$$p(\theta|D) \propto p(D|\theta)p(\theta),$$

o, de forma equivalente:

$$\log p(\theta|D) = \log p(D|\theta) + \log p(\theta) + \textit{constante}$$

En la práctica, actualizar $p(\theta|D)$ en base a $p(D|\theta)$ y $p(\theta)$ resulta complicado. Existen métodos basados en la simulación, como los de tipo Markov chain Monte Carlo (MCMC) que luego emplearemos.

Modelos jerárquicos Bayesianos

En el contexto de la inferencia Bayesiana, es habitual modelizar datos siguiendo una estructura que se conoce como *modelo jerárquico Bayesiano*, el cual está compuesto por los siguientes niveles:

- **Nivel 1:** Modelo para las observaciones $\rightarrow X_i \sim p(x|\theta_1, ..., \theta_m)$

  $p(x|\theta_1, ..., \theta_m)$ es la distribución que asumimos como generadora de la muestra, la cual define la **función de verosimilitud** del modelo.

- **Nivel 2:** Distribuciones sobre los parámetros *principales* $\rightarrow$ $\theta_j \sim p(\cdot|\tau_{j1}, ..., \tau_{jq_j})$

  $p(\cdot|\tau_{j1}, ..., \tau_{jq_j})$ es la distribución previa sobre θ_j

- **Nivel 3:** Distribuciones sobre los *hiperparámetros* (si es necesario) $\rightarrow$ $\tau_{j\ell} \sim p(\cdot)$

  $p(\cdot)$ es la distribución previa sobre $\tau_{j\ell}$, la cual es una distribución completamente determinada, que ya no depende de ningún parámetro desconocido (por ejemplo, $\tau_{j\ell} \sim Ga(2, 2)$)

Ejemplo: Regresión de Poisson jerárquica

Si $\boldsymbol{y} = (y_1, ..., y_n)$ es una realización de una muestra aleatoria Poisson y $\boldsymbol{x} = (x_1, ..., x_n)$ es una variable que informa sobre los valores de $\boldsymbol{y}$ (covariable), podemos plantear el siguiente modelo lineal generalizado (GLM):

$$Y_i \sim Po(\mu_i)$$

$$\log \mu_i = \alpha + \beta x_i$$

Esta estructura define el **Nivel 1** del que hablábamos antes.

Para completar la estructura de este modelo como modelo jerárquico Bayesiano, faltaría establecer distribuciones previas sobre α y β (**Nivel 2**).

Por ejemplo:

$$\alpha, \beta \sim N(\mu = 0, \sigma = 10)$$

Así pues, en este caso no habría hiperparámetros (no habría **Nivel 3**).

Cadena de Markov Monte Carlo (MCMC)

En algunos casos, puede obtenerse la distribución a posteriori de un parámetro de interés de forma analítica y relativamente sencilla. Sin embargo, en general, la inferencia Bayesiana depende de métodos de simulación, los cuales nos permiten aproximarnos a la distribución de interés $p(\theta|D)$. En concreto, se dispone de los métodos de tipo Markov chain Monte Carlo (MCMC), que esencialmente hacen lo siguiente:

- Genera una cadena $\{\theta^{(t)}\}$ cuya distribución límite es la posterior de interés, $p(\theta|D)$.
- Se basa en "explorar" el espacio de parámetros según reglas probabilísticas.
- Cada nuevo valor depende solo del anterior (propiedad de tipo Markov).

¿Qué métodos MCMC existen? Metropolis-Hastings, Gibbs sampling, etc.

Algoritmo de Metropolis-Hastings

1. Dado $\theta^{(t)}$, proponer θ^* con densidad $q(\theta^*|\theta^{(t)})$.
2. Calcular razón de aceptación:

$$\alpha = \text{mín}\left(1, \frac{p(D|\theta^*)p(\theta^*)q(\theta^{(t)}|\theta^*)}{p(D|\theta^{(t)})p(\theta^{(t)})q(\theta^*|\theta^{(t)})}\right)$$

3. Aceptar $\theta^{(t+1)} = \theta^*$ con probabilidad α, si no, $\theta^{(t+1)} = \theta^{(t)}$.

Muestreador de Gibbs

- Tipo de procedimiento MCMC que se emplea cuando se pueden deducir las distribuciones condicionales completas.
- En cada paso se actualiza un parámetro a la vez, condicionado a los demás.

$$\theta_i^{(t+1)} \sim p(\theta_i | \theta_{-i}^{(t)}, D)$$

- Muy eficiente en modelos jerárquicos.

Software para inferencia Bayesiana: NIMBLE

En general, implementar algoritmos de tipo MCMC es complicado. Por suerte, existen numerosas alternativas de software para realizar inferencia Bayesiana. En este cursillo, nos centramos en NIMBLE, el cual está asociado al paquete `nimble` de R. Para instrucciones de instalación, ver `https://r-nimble.org/html_manual/cha-installing-nimble.html`.

Sobre NIMBLE:

- Entorno flexible para ajustar modelos **complejos** en R mediante inferencia Bayesiana.
- Usa sintaxis similar a BUGS/JAGS.
- Rápido y eficiente, compilado en C++.
- Permite personalizar algoritmos MCMC y otros métodos numéricos (avanzado).

Ejemplo básico en NIMBLE

El modelo jerárquico Poisson antes descrito se implementaría con el siguiente código:

```
code <- nimbleCode({
  for(i in 1:n){
    # Etapa 1, verosimilitud del modelo
    y[i] ~ dpois(mu[i])
    log(mu[i]) <- alpha + beta*X[i]
  }
  # Etapa 2, previas sobre parámetros principales
  alpha ~ dnorm(0, sd = 10)
  beta ~ dnorm(0, sd = 10)
})
```

- La sintaxis resulta natural siguiendo la estructura del modelo.
- Además, hay que incluir valores iniciales para los parámetros, definir parámetros del procedimiento MCMC, etc. Luego lo veremos con más detalle.

Ejemplo básico en NIMBLE

- Podemos usar NIMBLE para ajustar modelos lineales, lineales generalizados, con efectos fijos y/o aleatorios (espaciales, temporales, etc.), etc.
- Sin embargo, por defecto solo podemos usar modelos con función de verosimilitud "estándar", en el sentido de que la verosimilitud sea la que se obtiene de asumir una distribución "clásica" para las observaciones (Nivel 1 del modelo jerárquico). Fundamentalmente, modelos cuya respuesta sigue una distribución de la familia exponencial.
- Para solventar esta limitación y ser capaces de ajustar otros modelos ("no estándar"), como son los modelos de tipo proceso puntual, podemos recurrir a algunas estrategias basadas en **reescribir la verosimilitud del modelo, de modo que se corresponda con la de un modelo estándar**.

El zeros-ones trick

Vamos a describir una estrategia, llamada "zeros-ones trick", la cual permite implementar modelos sujetos a cualquier función de verosimilitud (no estándar). Para más detalles sobre esta técnica, consultar Ntzoufras (2011).

El "zeros-ones trick" puede ejecutarse a través de la distribución de Poisson o a través de la Bernoulli. Nos centramos en el caso Poisson.

Vamos a reescribir la verosimilitud asociada al proceso puntual como la verosimilitud asociada a un modelo con respuesta Poisson...

El zeros-ones trick

Nos centramos, para facilitar su comprensión, en el término $\sum_{i=1}^{n} \log \lambda(\boldsymbol{x}_i)$ que aparece en la log-verosimilitud de interés. Podemos reescribirlo del siguiente modo:

$$\sum_{i=1}^{n} \log \lambda(\boldsymbol{x}_i) = \log \prod_{i=1}^{n} \lambda(\boldsymbol{x}_i) = \log \prod_{i=1}^{n} e^{\log \lambda(\boldsymbol{x}_i)} =$$

$$= \log \prod_{i=1}^{n} \frac{e^{-(-\log \lambda(\boldsymbol{x}_i))}(-\log \lambda(\boldsymbol{x}_i))^0}{0!} = \log \prod_{i=1}^{n} P(0, -\log \lambda(\boldsymbol{x}_i)),$$

donde $P(\cdot, -\log \lambda(\boldsymbol{x}_i))$ representa la función de probabilidad de una Poisson de parámetro $-\log \lambda(\boldsymbol{x}_i)$.

El zeros-ones trick

Análogamente, para la aproximación de la integral que aparece en la log-verosimilitud, $-\int_W \lambda(u)\,du \approx -\sum_{j=1}^{m} \lambda(u_j)\cdot w_j$, podemos reescribir:

$$-\sum_{j=1}^{m} \lambda(u_j)\cdot w_j = \sum_{j=1}^{m} \log e^{-\lambda(u_j)\cdot w_j} = \log \prod_{j=1}^{m} e^{-\lambda(u_j)\cdot w_j} =$$

$$= \log \prod_{j=1}^{m} \frac{e^{-\lambda(u_j)\cdot w_j}(\lambda(u_j)\cdot w_j)^0}{0!} = \log \prod_{j=1}^{m} P(0, \lambda(u_j)\cdot w_j),$$

donde $P(\cdot, \lambda(u_j)\cdot w_j)$ representa la función de probabilidad de una Poisson de parámetro $\lambda(u_j)\cdot w_j$.

El zeros-ones trick

Por tanto, la función de verosimilitud del proceso puntual es equivalente a la del siguiente modelo de respuesta Poisson:

$$Y_i \sim Po(\mu_i)$$

$$\mu_i = -\log \lambda(\boldsymbol{x}_i), i = 1, ..., n$$

$$\mu_i = \lambda(u_{i-n}) \cdot w_{i-n}, i = n+1, ..., n+m,$$

para una muestra observada $\boldsymbol{y} = (y_1, ..., y_n, y_{n+1}, ..., y_{n+m}) = (0, ..., 0)$.

Además, hay que tener en cuenta que el término μ_i debe ser positivo para que la distribución de Poisson esté bien definida. La opción habitual es reemplazar μ_i por $\mu_i + C$, donde C es una constante positiva "grande" (Ntzoufras, 2011). En nuestro caso, trabajaremos con $C = 10^8$.

Datos de patrón puntual en criminología

En los ejemplos que se mostrarán en el cursillo se hará uso de un conjunto de datos que contiene información sobre robos ocurridos en la ciudad de Valencia (España) en el año 2017. En concreto:

- Nos centramos en el centro de la ciudad y sus distritos colindantes.
- Se observaron 641 eventos de tipo robo (llamadas al número 112 codificadas como este tipo de delito).
- Para cada evento se dispone de su localización espacial y temporal, por lo que podemos escribir (x_i, y_i, t_i) para el evento i. Además:
  - Las coordenadas (x_i, y_i) están en el sistema UTM.
  - La coordenada t_i está en días, dentro del intervalo $[0, 365]$
- Se dispone de algunas covariables sociodemográficas definidas espacialmente sobre el área de estudio.

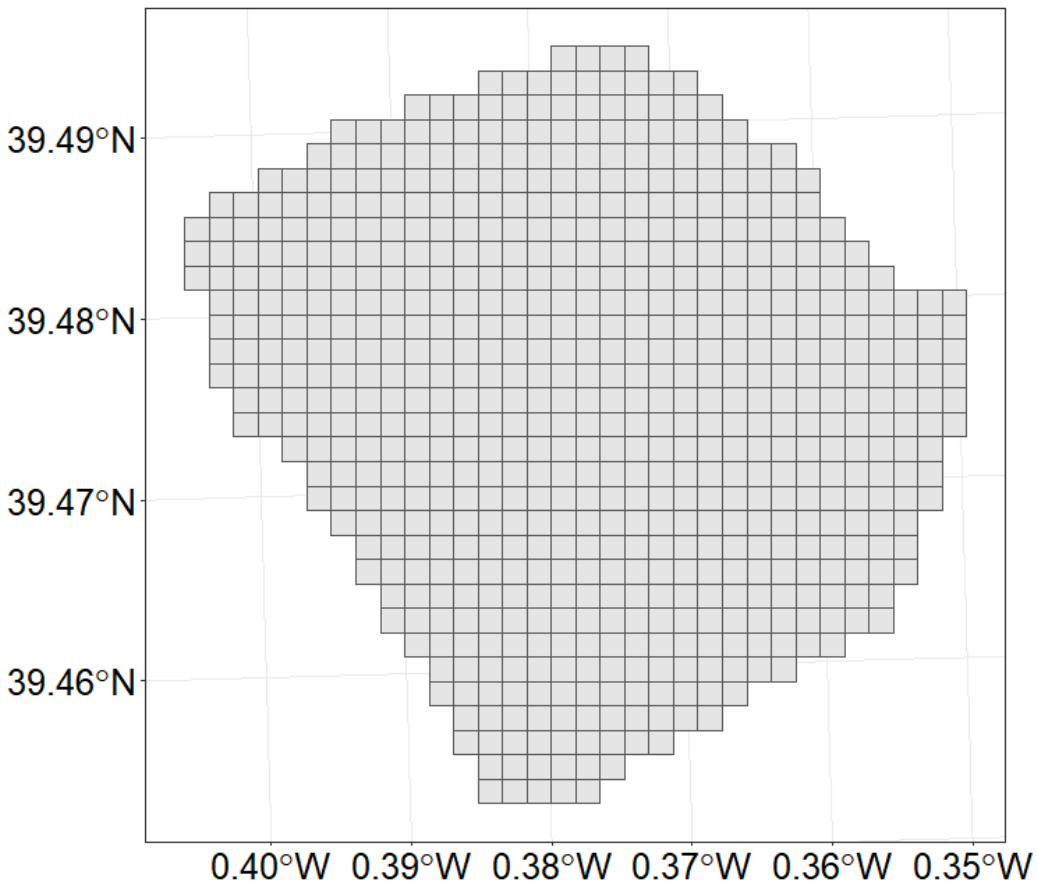
Área de estudio: Ventana espacial

Esta imagen contiene la zona de Valencia que consideraremos:

Figura: Imagen extraída de *OpenStreetMap*

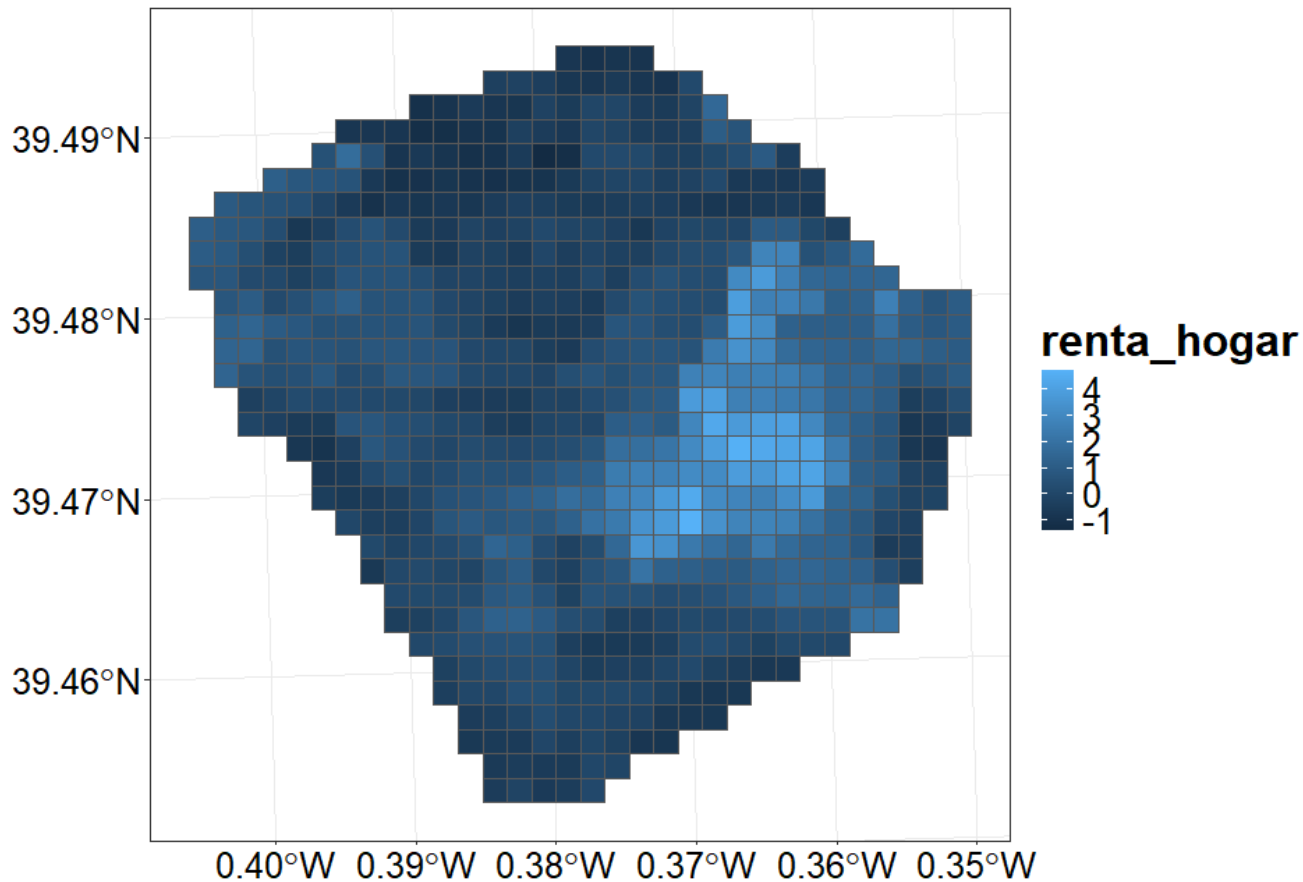
Área de estudio: Ventana espacial

Trabajaremos sobre la siguiente malla, que denotamos G, la cual contiene 647 celdas de 150×150 m^2 (de área 22500 m^2):



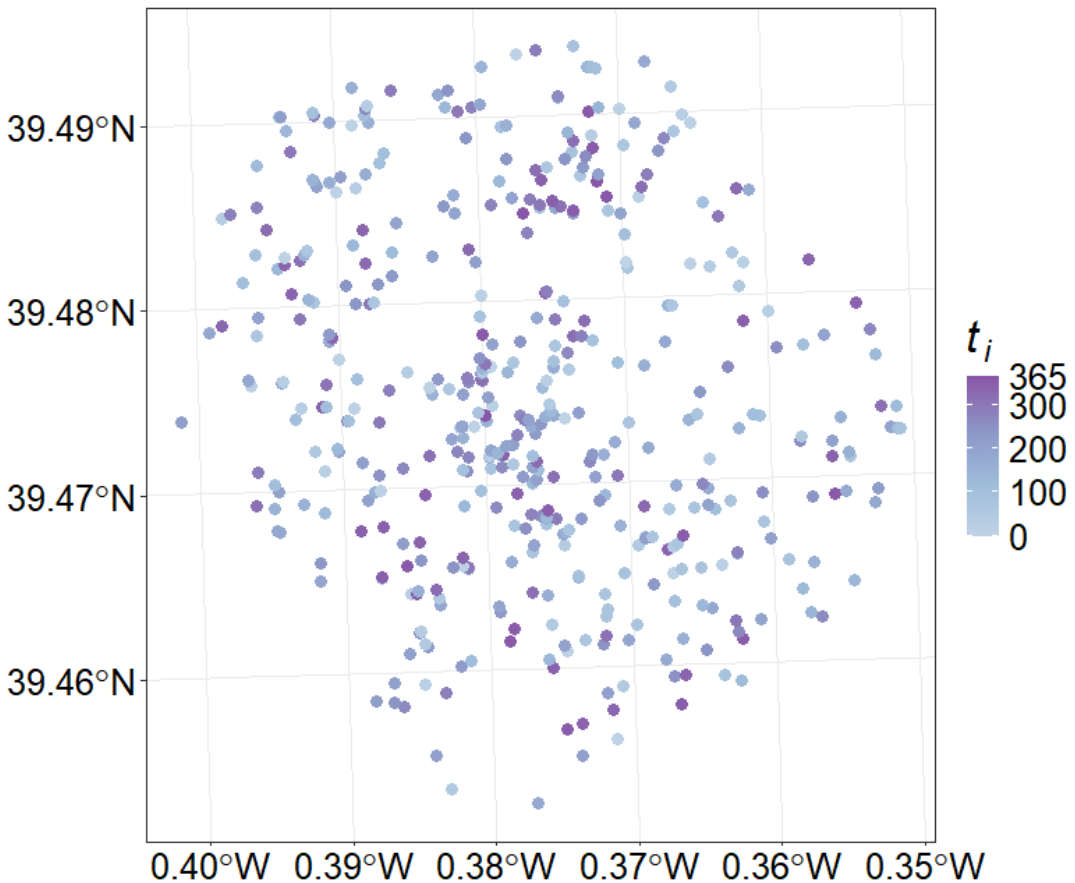
Covariables espaciales

Sobre esta malla se dispone de información de las siguientes cuatro covariables con valores estandarizados respecto de los valores del conjunto de la ciudad: `poblacion_65_mas`, `poblacion_15_29`, `poblacion_extranjera`, `renta_hogar`.



Patrón espacio-temporal

Este es el patrón puntual de eventos sobre la región espacio-temporal considerada:



Aproximación de la integral

Recordemos que la integral $\int_W \lambda(u)\,du$ se aproxima con una cuadrícula:

$$\int_W \lambda(u)\,du \approx \sum_{j=1}^{m} \lambda(u_j) \cdot w_j$$

donde u_j son puntos en una malla y w_j el peso (área/peso del píxel).

En nuestro caso concreto, para $W = G \times T$:

$$\int_G \int_T \lambda(x, y, t)\,dt\; d(x, y) \approx \sum_{k=1}^{365} \sum_{\ell=1}^{647} \lambda(x_\ell, y_\ell, t_k) \cdot 22500 \cdot 1,$$

donde 22500 representa el área de cada celda espacial y 1 la longitud del intervalo temporal usado para aproximar la integral (se divide el intervalo temporal $T = [0, 365]$ en subintervalos de longitud 1).

Ejemplo: Proceso Poisson homogéneo

Función de intensidad: $\lambda(x, y, t) = \lambda_0$

Parámetros a estimar: λ_0

Distribuciones previas:

$$\lambda_0 \sim Ga(a, b)$$

$$a = 0.1\frac{n}{|W|}, b = 0.1$$

Por tanto, se elige una previa moderadamente informativa, con media igual a la estimación de máxima verosimilitud, $\frac{n}{|W|}$, y varianza con valor $10\frac{n}{|W|}$.

Ejemplo: Proceso Poisson homogéneo

Modelo en `nimble`:

```
Poisson_hom <- nimbleCode({
  lambda0 ~ dgamma(0.1*lambda0_base,0.1)

  for (i in 1:N_events) {

    lambda_event[i] <- lambda0
    log_L[i] <- log(lambda_event[i])
    z[i] <- -log_L[i]+C
    zeros[i] ~ dpois(z[i])

  }

  for (i in (N_events+1):N) {

    lambda_int[1:S,i-N_events] <- rep(lambda0,S)
    # Integral approximation
    log_L[i] <- -sum(lambda_int[1:S,i-N_events])*area_cell*1
    z[i] <- -log_L[i]+C
    zeros[i] ~ dpois(z[i])

  }
})
```

Ejemplo: Proceso Poisson homogéneo

Carga de datos y creación de objetos:

```
load("Data/covariates.rda")
load("Data/grid_crimen.rda")
load("Data/events_crime_valencia.rda")
grid_val_sf=as(grid_val,"sf")
time_events=events_study_area$Time
area_cell=max(st_area(grid_val_sf))
Tmax=365
time_points=seq(0,Tmax,1)
```

Constantes del modelo:

```
constants <- list(
  N = length(time_events)+length(time_points),
  N_events = length(time_events),
  lambda0_base = length(time_events)/(sum(st_area(grid_val_sf))*Tmax),
  C = 100000000,
  S = length(grid_val),
  area_cell = area_cell
)
```

Ejemplo: Proceso Poisson homogéneo

Valores iniciales para los parámetros a estimar:

```
inits <- function() list(lambda0 = as.numeric(constants$lambda0_base))
```

Respuesta del modelo (data):

```
data <- list(zeros = rep(0,constants$N))
```

Parámetros a monitorizar:

```
monitors_pars <- c("lambda0","lambda_int")
```

Ajuste del modelo (parámetros del procedimiento MCMC):

```
mcmc.output <- nimbleMCMC(Poisson_hom,
                          data = data, inits = inits, constants = constants,
                          monitors = monitors_pars,
                          % Parámetros del MCMC
                          niter = 20000, nburnin = 10000, nchains = 2, thin = 10,
                          summary = TRUE, WAIC = TRUE)
```

Ejemplo: Proceso Poisson homogéneo

Sobre los parámetros del procedimiento MCMC. Recordemos que para cada parámetro del modelo, se genera una cadena $\{\theta^{(t)}\}$. Debemos escoger los siguientes valores:

- `niter`: Número total de valores a simular para cada parámetro implicado en el modelo.
- `nburnin`: Número de valores de la parte inicial de la cadena que se descartan. Esto es conveniente ya que el procedimiento MCMC tarda en converger a la distribución a posteriori objetivo.
- `thin`: Número que indica cada cuántos valores simulados del procedimiento MCMC se guardan para su posterior análisis. Se utiliza sobre todo por ahorrar memoria.
- `nchains`: Número de cadenas MCMC que se generan para cada uno de los parámetros implicados. Es conveniente generar múltiples cadenas para confirmar que todas ellas son coherentes. En otro caso, puede indicar de la existencia de problemas de convergencia, identificación, etc.

Ejemplo: Proceso Poisson homogéneo

Análisis de la convergencia:

```
library(coda)
combinedchains <- mcmc.list(mcmc(mcmc.output$samples$chain1[,"lambda0"]),
                            mcmc(mcmc.output$samples$chain2[,"lambda0"]))
plot(combinedchains)
```

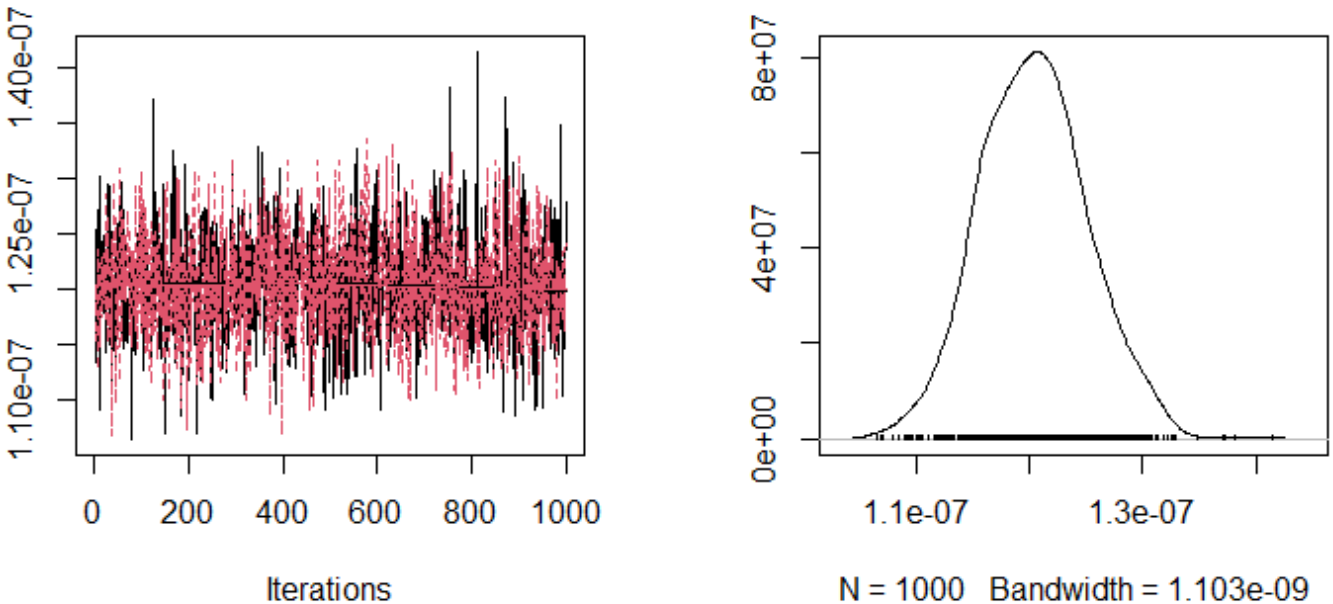


Figura: Análisis de las cadenas MCMC con el paquete coda. En la gráfica de la derecha se muestra la aproximación obtenida de la distribución $p(\lambda_0|D)$

Ejemplo: Proceso Poisson homogéneo

Análisis de la convergencia:

```
gelman.diag(combinedchains)
gelman.plot(combinedchains)
effectiveSize(mcmc.output$samples$chain1[,"lambda0"])
```

El resultado de `gelman.diag(combinedchains)` nos da una estimación del valor conocido como $\hat{R}$ (Gelman and Rubin, 1992), el cual nos indica si existe coherencia entre las cadenas. Valores cercanos a 1 (habitualmente, inferiores a 1.1) nos indican coherencia y, por tanto, ausencia de problemas de convergencia o identificación.

Ejemplo: Proceso Poisson homogéneo

Análisis de la convergencia:

```
gelman.diag(combinedchains)
gelman.plot(combinedchains)
effectiveSize(mcmc.output$samples$chain1[,"lambda0"])
```

Por otra parte, la función `effectiveSize` devuelve el número de simulaciones efectivas, entendidas estas como el número de simulaciones obtenidas, ajustadas por el nivel de autocorrelación existente entre ellas. En nuestro caso, el número efectivo máximo, teniendo en cuenta los valores que hemos escogido para el procedimiento MCMC, sería $(20000 - 10000)/10 = 1000$. Este aspecto de la autocorrelación también puede valorarse con la función `acf`.

Ejemplo: Proceso Poisson homogéneo

Resumen de resultados:

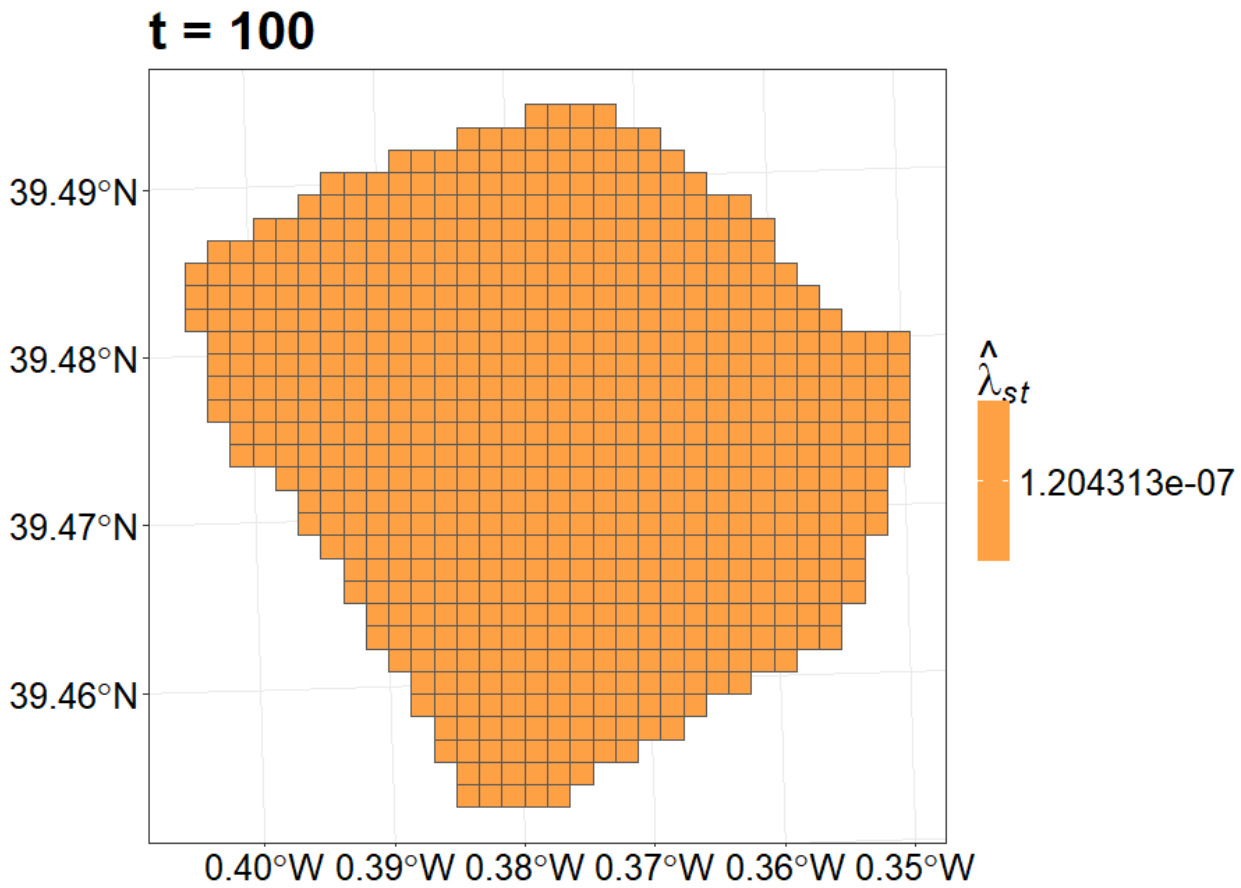
```
head(mcmc.output$summary$all.chains)
                          Mean      Median      St.Dev.    95%CI_low    95%CI_upp
lambda0           1.204093e-07 1.20331e-07 4.756428e-09 1.116049e-07 1.301098e-07
lambda_int[1, 1] 1.204093e-07 1.20331e-07 4.756428e-09 1.116049e-07 1.301098e-07
lambda_int[2, 1] 1.204093e-07 1.20331e-07 4.756428e-09 1.116049e-07 1.301098e-07
lambda_int[3, 1] 1.204093e-07 1.20331e-07 4.756428e-09 1.116049e-07 1.301098e-07
lambda_int[4, 1] 1.204093e-07 1.20331e-07 4.756428e-09 1.116049e-07 1.301098e-07
lambda_int[5, 1] 1.204093e-07 1.20331e-07 4.756428e-09 1.116049e-07 1.301098e-07
```

Se deduce una estimación $\hat{\lambda}_0 = 1.20 \cdot 10^{-7}$ (media a posteriori) y un intervalo de credibilidad al 95 %, $\text{IC}_{95\,\%}(\lambda_0) = [1.12 \cdot 10^{-7}, 1.30 \cdot 10^{-7}]$.

Además, la variable `lambda_int` contiene los valores de $\lambda(x, y, t)$, para cada valor de la partición espacio-temporal (para cada celda del grid y cada subintervalo de longitud 1 en $[0, 365]$). Podemos mapear los valores de esta variable.

Ejemplo: Proceso Poisson homogéneo

Este es la intensidad estimada para el día $t = 100$, pero es la misma en cualquier momento del periodo:



Ejemplo: Proceso Poisson inhomogéneo

Pasamos ahora a un modelo inhomogéneo que incluye las cuatro covariables sociodemográficas antes mencionadas, $X_j, j = 1, ..., 4$.

Función de intensidad:

$$\lambda(x, y, t) = \lambda_0 \exp(\beta_1 X_1(x, y) + \beta_2 X_2(x, y) + \beta_3 X_3(x, y) + \beta_4 X_4(x, y))$$

Parámetros a estimar: λ_0, β_1, β_2, β_3, β_4

Distribuciones previas:

$$\lambda_0 \sim Ga(a, b)$$

$$a = 0.1\frac{n}{|W|}, b = 0.1$$

$$\beta_j \sim N(\mu = 0, \sigma = 10), j = 1, ..., 4$$

Ejemplo: Proceso Poisson inhomogéneo

Modelo en `nimble`:

```
Poisson_inhom1 <- nimbleCode({
  lambda0 ~ dgamma(0.1*lambda0_base,0.1)

  for (j in 1:4){
    beta[j] ~ dnorm(0,sd=10)
  }

  for (i in 1:N_events) {

    lambda_event[i] <- lambda0*exp(beta[1]*X1[Id_cell[i]]+beta[2]*X2[Id_cell[i]]+
                                   beta[3]*X3[Id_cell[i]]+beta[4]*X4[Id_cell[i]])
    log_L[i] <- log(lambda_event[i])
    z[i] <- -log_L[i]+C
    zeros[i] ~ dpois(z[i])

  }

  for (i in (N_events+1):N) {

    lambda_int[1:S,i-N_events] <- lambda0*exp(beta[1]*X1[1:S]+beta[2]*X2[1:S]+beta[3]*X3[1:S]+beta[4]*X4[1:S])
    # Integral approximation
    log_L[i] <- -sum(lambda_int[1:S,i-N_events])*area_cell*1
    z[i] <- -log_L[i]+C
    zeros[i] ~ dpois(z[i])

  }
})
```

Ejemplo: Proceso Poisson homogéneo

Resumen de resultados:

```
head(mcmc.output$summary$all.chains,5)
```

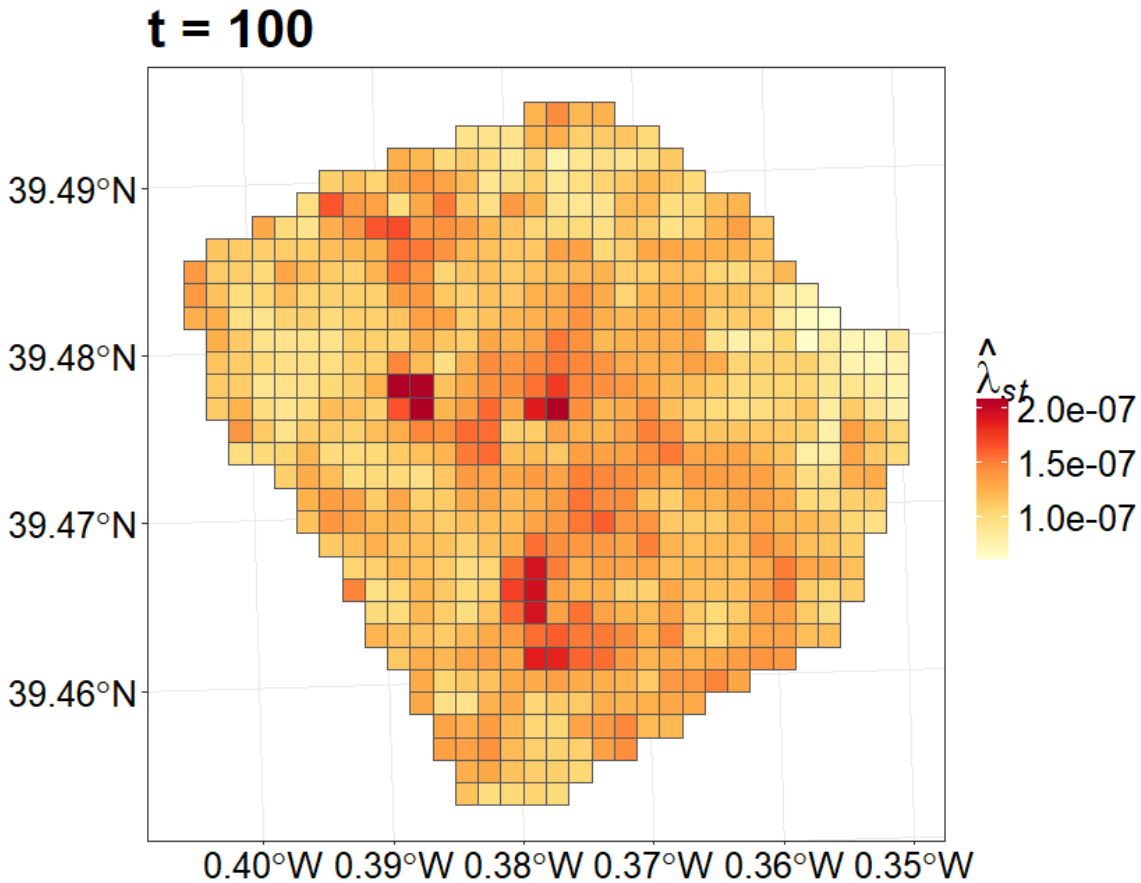
	Mean	Median	St.Dev.	95%CI_low	95%CI_upp
beta[1]	3.875520e-02	3.713176e-02	5.664475e-02	-7.402324e-02	1.550101e-01
beta[2]	-1.396682e-01	-1.406006e-01	5.107645e-02	-2.362728e-01	-3.899548e-02
beta[3]	2.707824e-01	2.710238e-01	8.452731e-02	1.052805e-01	4.329212e-01
beta[4]	1.190114e-01	1.188822e-01	4.674775e-02	2.569455e-02	2.131054e-01
lambda0	1.131005e-07	1.130285e-07	6.147917e-09	1.010141e-07	1.254342e-07

La intensidad del proceso puntual presenta una asociación positiva con el porcentaje de población extranjera y el nivel de renta, mientras que una asociación negativa con el porcentaje de población entre 15 y 29 años.

Por otra parte, la estimación de λ_0 resulta similar a la del caso anterior, aunque algo más baja.

Ejemplo: Proceso Poisson inhomogéneo

Este es la intensidad estimada para el día $t = 100$. Ahora varía espacialmente, pero vuelve a ser la misma en cualquier momento del periodo:



Ejemplo: Proceso Poisson inhomogéneo con parte temporal

Podemos incorporar un término extra que haga que la intensidad varíe temporalmente. En el caso de la ciudad de Valencia, es habitual que la intensidad de robo sea mayor en el verano. Asumimos ahora un modelo que otorga mayor intensidad al día $t = 200$ (a mitad del verano).

Función de intensidad:

$$\lambda(x, y, t) = \lambda_0 \exp(\beta_1 X_1(x, y) + \beta_2 X_2(x, y) + \beta_3 X_3(x, y) + \beta_4 X_4(x, y)) \exp(-\frac{1}{\omega}|t - 200|)$$

Parámetros a estimar: λ_0, β_1, β_2, β_3, β_4, ω

Distribuciones previas:

$$\lambda_0 \sim Ga(a, b)$$

$$a = 0.1\frac{n}{|W|}, b = 0.1$$

$$\beta_j \sim N(\mu = 0, \sigma = 10), j = 1, ..., 4$$

$$\omega \sim U(0, 1000)$$

Ejemplo: Proceso Poisson inhomogéneo con parte temporal

Modelo en `nimble`:

```
Poisson_inhom2 <- nimbleCode({
  lambda0 ~ dgamma(0.1*lambda0_base,0.1)
  for (j in 1:4){
    beta[j] ~ dnorm(0,sd=10)
  }
  omega ~ dunif(0,100)

  for (i in 1:N_events) {

    lambda_event[i] <- lambda0*exp(beta[1]*X1[Id_cell[i]]+beta[2]*X2[Id_cell[i]]+
                                     beta[3]*X3[Id_cell[i]]+beta[4]*X4[Id_cell[i]])*
                                exp(-abs(time_events[i]-200)/omega)
    log_L[i] <- log(lambda_event[i])
    z[i] <- -log_L[i]+C
    zeros[i] ~ dpois(z[i])

  }

  for (i in (N_events+1):N) {

    lambda_int[1:S,i-N_events] <- lambda0*exp(beta[1]*X1[1:S]+beta[2]*X2[1:S]+
                                                beta[3]*X3[1:S]+beta[4]*X4[1:S])*
                                              exp(-abs(time_points[i-N_events]+0.5-200)/omega)
    # Integral approximation
    log_L[i] <- -sum(lambda_int[1:S,i-N_events])*area_cell*1
    z[i] <- -log_L[i]+C
    zeros[i] ~ dpois(z[i])

  }
})
```

Ejemplo: Proceso Poisson inhomogéneo con parte temporal

Resumen de resultados:

```
head(mcmc.output$summary$all.chains,5)
                          Mean        Median      St.Dev.     95%CI_low     95%CI_upp
beta[1]           3.870810e-02  3.663111e-02 5.518833e-02 -7.092247e-02  1.483649e-01
beta[2]          -1.397210e-01 -1.389396e-01 4.984865e-02 -2.369231e-01 -4.312131e-02
beta[3]           2.732685e-01  2.744795e-01 8.253609e-02  1.100424e-01  4.362162e-01
beta[4]           1.199740e-01  1.201839e-01 4.676723e-02  2.867280e-02  2.144508e-01
lambda0           1.325805e-07  1.316306e-07 1.002717e-08  1.153853e-07  1.547891e-07
tail(mcmc.output$summary$all.chains,1)
                          Mean        Median      St.Dev.     95%CI_low     95%CI_upp
omega                624.5052      606.2203     186.5675      323.6484      977.6932
```

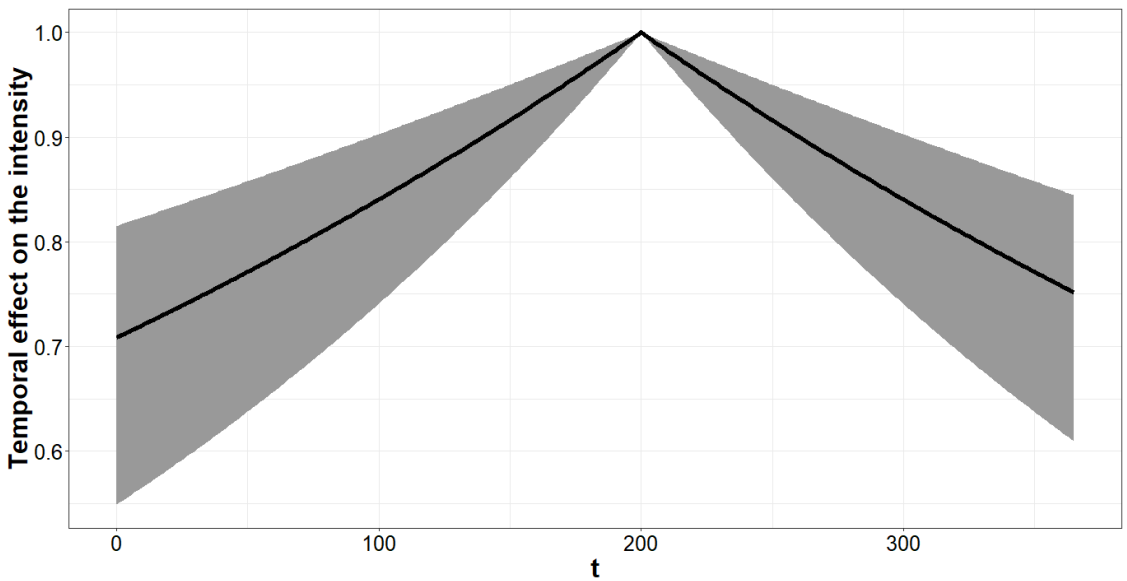
Las estimaciones de los β_j's y de λ_0 varían mínimamente (la que más varía es la de λ_0). El intervalo de credibilidad al 95 % sobre ω es bastante ancho, $\text{IC}_{95\,\%}(\omega) = [323.64, 977.69]$, lo que sugiere que existe una elevada incertidumbre sobre la estimación de este parámetro.

Ejemplo: Proceso Poisson inhomogéneo con parte temporal

Podemos medir la incertidumbre existente sobre el efecto temporal para cualquier instante $t \in [0, 365]$. En concreto, a partir de los valores MCMC simulados para ω, podemos calcular los correspondientes valores de $\exp(-\frac{1}{\omega}|t - 200|)$, para t fijado:

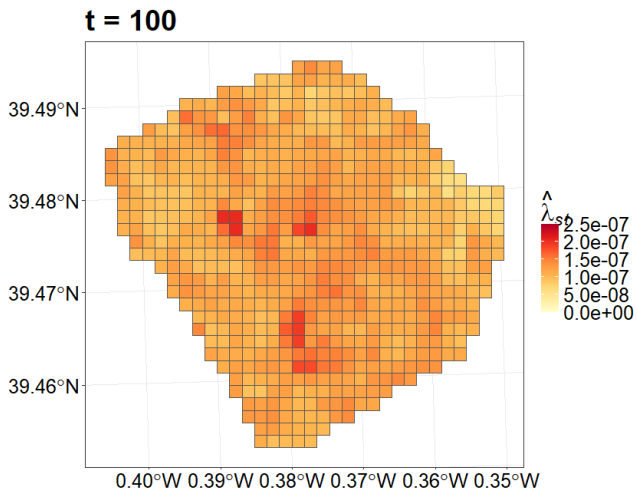
```
df_plot=c()
omegas=mcmc.output$samples$chain1[,grep("omega",colnames(mcmc.output$samples$chain1))]
for (t in seq(0,365,1)){
  temp_effect=exp(-abs(t-200)/omegas)
  df_plot=rbind(df_plot,data.frame(t,Lo=quantile(temp_effect,0.025),
                                   Mean=mean(temp_effect),Up=quantile(temp_effect,0.975)))
}
```

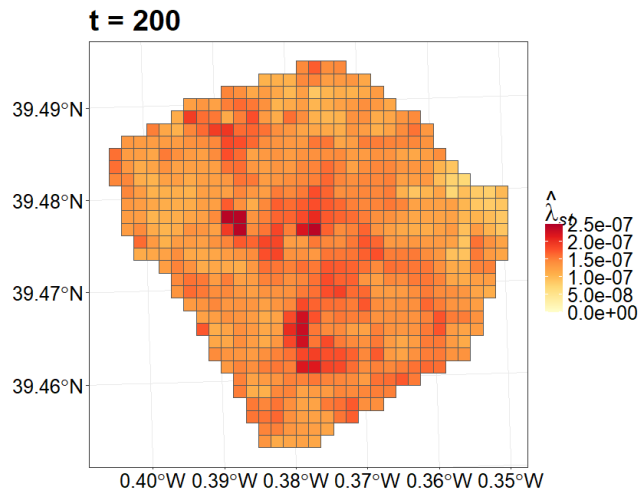
Con ggplot2 podemos obtener la siguiente gráfica:



Ejemplo: Proceso Poisson inhomogéneo con parte temporal

Por último, comprobamos ahora que la nueva estimación de la intensidad sí es variable en el tiempo.





Ejemplo: Proceso Poisson inhomogéneo con parte temporal

Otra opción es incluir la variación temporal a través de un efecto aleatorio de tipo *random walk*. En este caso, necesitamos trabajar a un cierto nivel de resolución temporal, como por ejemplo las semanas del periodo de estudio. Si denotamos por $W(t)$ a la semana del año en la que cae el instante $t \in [0, 365]$, podemos plantear el siguiente modelo.

Función de intensidad:

$$\lambda(x, y, t) = \lambda_0 \exp(\beta_1 X_1(x, y) + \beta_2 X_2(x, y) + \beta_3 X_3(x, y) + \beta_4 X_4(x, y)) \exp(\varepsilon_{W(t)}),$$

donde asumimos $\varepsilon_1 \sim N(\mu = 0, \sigma = \sigma_\varepsilon)$ y $\varepsilon_W \sim N(\mu = \varepsilon_{W-1}, \sigma = \sigma_\varepsilon)$, para $W = 2, ..., 53$.

Parámetros a estimar: λ_0, β_1, β_2, β_3, β_4, ε_1, ..., ε_{53}, σ_ε

Distribuciones previas:

$$\lambda_0 \sim Ga(a, b), a = 0.1\frac{n}{|W|}, b = 0.1$$

$$\beta_j \sim N(\mu = 0, \sigma = 10), j = 1, ..., 4$$

$$\sigma_\varepsilon \sim U(0, 10)$$

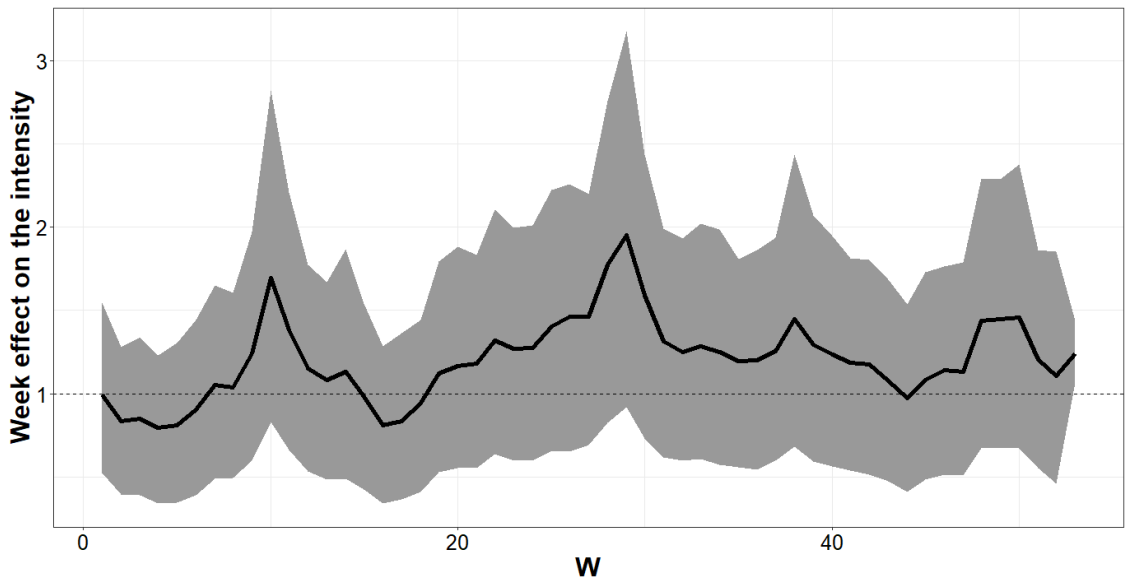
Ejemplo: Proceso Poisson inhomogéneo con parte temporal

Modelo en `nimble`:

```
Poisson_inhom3 <- nimbleCode({
  lambda0 ~ dgamma(0.1*lambda0_base,0.1)
  for (j in 1:4){
    beta[j] ~ dnorm(0,sd=10)
  }
  epsilon[1] ~ dnorm(0,sd=sigma_epsilon)
  for (W in 2:53){
    epsilon[W] ~ dnorm(epsilon[W-1],sd=sigma_epsilon)
  }
  sigma_epsilon ~ dunif(0,10)
  for (i in 1:N_events) {

    lambda_event[i] <- lambda0*exp(beta[1]*X1[Id_cell[i]]+beta[2]*X2[Id_cell[i]]+
                                   beta[3]*X3[Id_cell[i]]+beta[4]*X4[Id_cell[i]])*
                               exp(epsilon[Week_events[i]])
    log_L[i] <- log(lambda_event[i])
    z[i] <- -log_L[i]+C
    zeros[i] ~ dpois(z[i])

  }
  for (i in (N_events+1):N) {

    lambda_int[1:S,i-N_events] <- lambda0*exp(beta[1]*X1[1:S]+beta[2]*X2[1:S]+
                                              beta[3]*X3[1:S]+beta[4]*X4[1:S])*
                                          exp(epsilon[Week_time_points[i-N_events]])
    # Integral approximation
    log_L[i] <- -sum(lambda_int[1:S,i-N_events])*area_cell*1
    z[i] <- -log_L[i]+C
    zeros[i] ~ dpois(z[i])
  }
})
```

Ejemplo: Proceso Poisson inhomogéneo con parte temporal

En este caso, podemos graficar directamente la estimación sobre $\exp(\varepsilon_W)$, $W = 1, ..., 53$:



No se observa un patrón claro. Se aprecian picos en verano y en marzo (periodo de Fallas, probablemente), pero la banda de credibilidad al 95 % intersecta al 1 en todos los casos.

Algunos ejemplos más avanzados

El objetivo del resto del cursillo es describir algunos modelos de proceso puntual más sofisticados, o que permiten abordar problemáticas concretas. En particular, en algunos casos donde la inferencia Bayesiana resulta conveniente. Vamos a hablar de los siguientes ejemplos:

- Modelos con splines para estimar la intensidad espacial de forma suavizada.
- Modelos para lidiar con la infradetección de los eventos.
- Modelos para tratar con la incertidumbre en la localización temporal de los eventos.
- Modelos para tener en cuenta la interacción espacio-tiempo (modelos *self-exciting*).

Splines para la intensidad espacial

- En lugar de emplear covariables espaciales para la estimación de la intensidad espacial del proceso, podemos escoger una base de splines adecuada, $\{X_\ell(x, y)\}_{\ell=1}^{L}$. De este modo, la parte espacial de la intensidad puede modelizarse como: $\exp(\sum_{\ell=1}^{L} \beta_\ell X_\ell(x, y))$
- Brevemente, una función spline es una función suave con soporte en el área de estudio. Una base de splines permite capturar diferentes formas de variabilidad dentro de dicha área de estudio. Para consultar detalles sobre bases de splines, ver Wood (2017).
- En este caso, probamos con la familia de *thin plate splines*, la cual puede construirse mediante el paquete `npreg`, entre otros.

Splines para la intensidad espacial

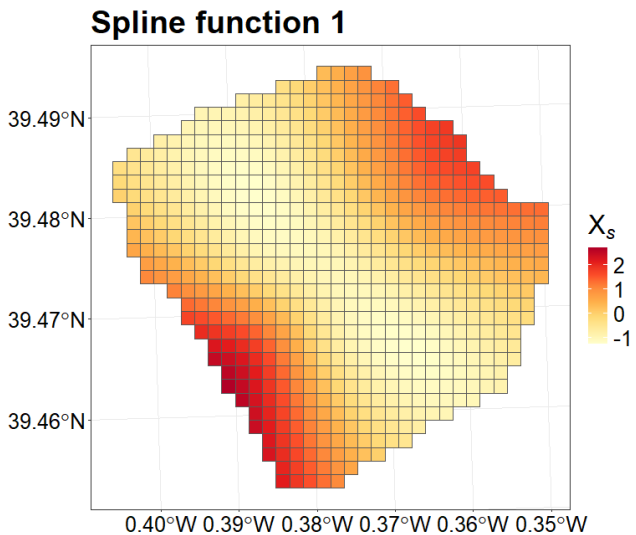
Aquí se muestra el código para su obtención. Se necesita definir un conjunto de nodos (*knots*) sobre la región espacial:

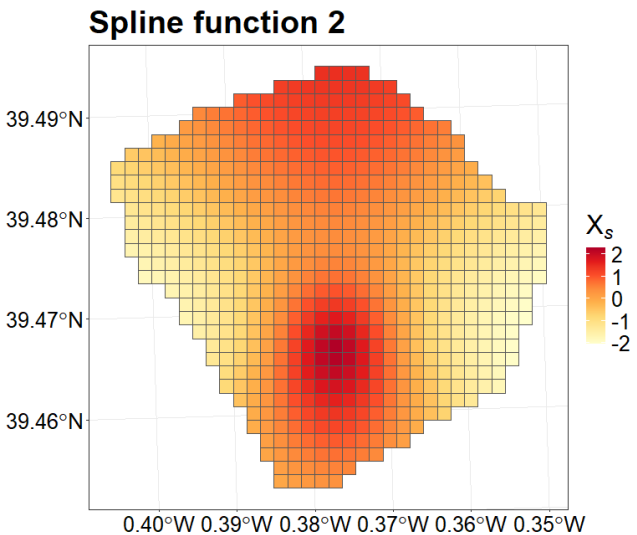
```
library(npreg)
# Spline basis over the spatial window
centroids_grid=sf::st_centroid(grid_val_sf)
aux=as.numeric(unlist(centroids_grid$geometry))
x=aux[seq(1,length(aux),2)]
y=aux[seq(2,length(aux),2)]
input=cbind(x,y)
k=4
knots=expand.grid(quantile(x,1:(k-1)/k),quantile(y,1:(k-1)/k))
X_splines=basis.tps(input, knots, m = 2, rk = TRUE, intercept = FALSE,
                    ridge = FALSE)
```

Este código da lugar a una base de tamaño 9 (con 9 funciones spline). Podríamos definir más nodos para conseguir una base que describa la variación espacial con más detalle. Sin embargo, esto aumentaría el coste computacional.

Splines para la intensidad espacial

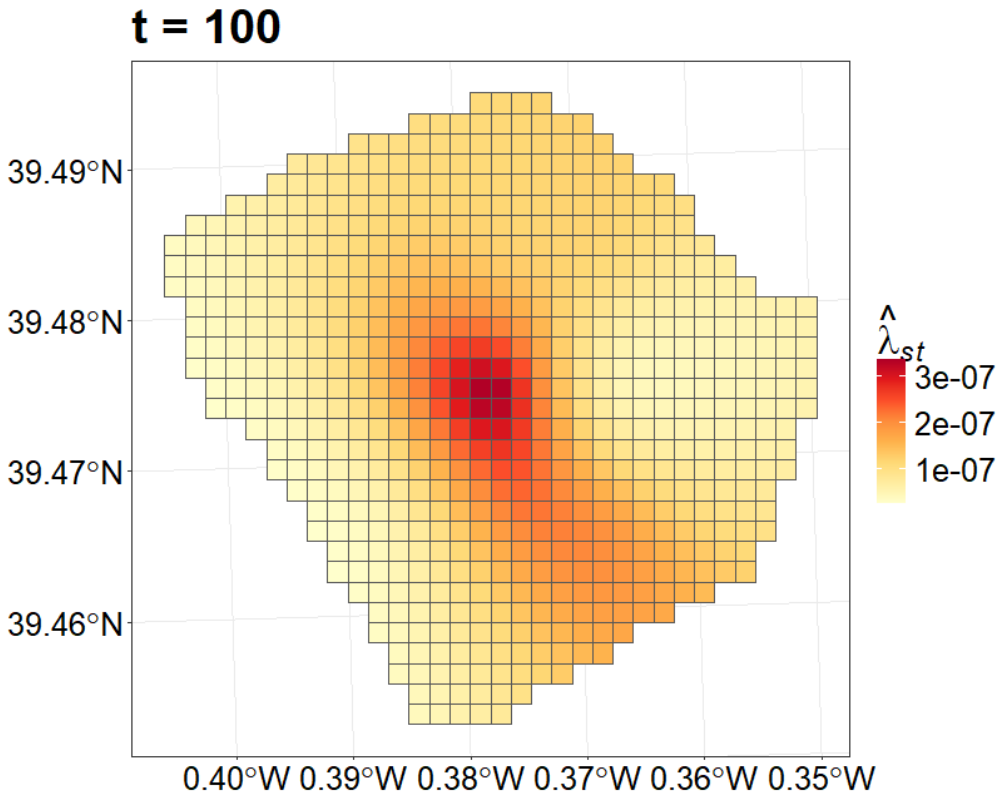
Como ejemplo, veamos los dos primeros elementos de esta base de splines:





Splines para la intensidad espacial

Esta modelización da lugar a la siguiente estimación de la intensidad espacial, por ejemplo, para $t = 100$:



Observamos que resulta mucho más suavizada que la obtenida con covariables.

Modelizar la infradetección de eventos

- Es habitual que ciertos eventos de carácter espacio-temporal se infradetecten. En concreto, en el ámbito de la criminología, algunos delitos menores no se reportan o no se identifican, lo que conlleva a la infraestimación de la intensidad del delito.
- Siguiendo la propuesta de Stoner et al. (2019) para datos de conteos, podemos plantear el siguiente modelo general para la función de intensidad:

$$\lambda(x, y, t) = \pi(x, y, t)\tilde{\lambda}(x, y, t),$$

donde $\pi(x, y, t)$ toma valores en $[0, 1]$, representando el nivel de infradetección, y $\tilde{\lambda}(x, y, t)$ representa la intensidad verdadera del proceso.
Nótese que este modelo garantiza que se cumpla que $\tilde{\lambda}(x, y, t) \geq \lambda(x, y, t), \forall(x, y, t)$.

Modelizar la infradetección de eventos

- Para simplificar, asumimos en este caso que la intensidad verdadera es constante en espacio y tiempo, $\tilde{\lambda}(x, y, t) = \lambda_0$.
- Asumimos, a su vez, que la infradetección es únicamente variable en el tiempo, $\pi(x, y, t) \equiv \pi(t)$.
- Podemos entonces modelizar $\pi(t)$ mediante una regresión logística que incluye un efecto aleatorio temporal a nivel de semana (como en ejemplos previos):

$$\text{logit}(\pi(t)) = \log\left(\frac{\pi(t)}{1-\pi(t)}\right) = \alpha + \varepsilon_{W(t)}$$

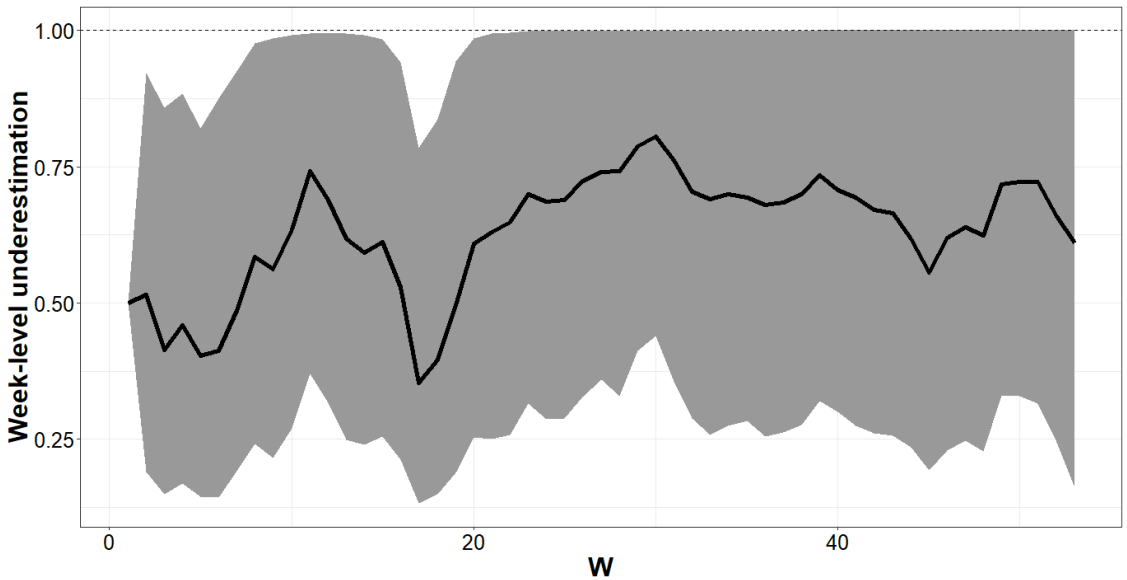
Modelizar la infradetección de eventos

El código debe modificarse, incluyendo ahora la modelización de $\pi(t)$:

```
lambda0 ~ dgamma(0.1*lambda0_base,0.1)
alpha ~ dnorm(0,sd=10)
epsilon[1] ~ dnorm(0,sd=sigma_epsilon)
for (W in 2:53){
    epsilon[W] ~ dnorm(epsilon[W-1],sd=sigma_epsilon)
    pi[W] <- exp(alpha+epsilon[W])/(1+exp(alpha+epsilon[W]))
}
sigma_epsilon ~ dunif(0,10)
```

Modelizar la infradetección de eventos

Se obtiene la siguiente estimación para $\pi(t)$ a nivel de semana:



- Se observa que, esencialmente, $\pi(t)$ está detectando la variación en la intensidad observada. Esto es porque la suposición de que $\tilde{\lambda}(x, y, t) = \lambda_0$ no es realista.
- Además, este modelo es difícilmente *identificable*, como se discute en Stoner et al. (2019). Es necesario usar previas informativas sobre los niveles de infradetección (en base al conocimiento previo), para poder identificar las diferentes partes del modelo.

Modelizar la incertidumbre en la localización temporal

- Es frecuente que nos encontremos con incertidumbre en la localización espacial y/o temporal de los eventos.
- En el ámbito de la criminología, es habitual que para ciertos delitos no se disponga de su localización temporal exacta, sino de un intervalo temporal. En estos casos, una localización espacio-temporal (x_i, y_i, t_i) pasa a ser $(x_i, y_i, t_i^{\text{start}}, t_i^{\text{end}})$, donde $[t_i^{\text{start}}, t_i^{\text{end}}]$ indica el intervalo temporal en el que ha ocurrido el evento (Briz-Redón, 2024).
- Esto nos lleva a un contexto de *datos faltantes*, que puede resolverse mediante inferencia Bayesiana asumiendo una distribución previa sobre la localización incierta, t_i:

$$t_i \sim U(t_i^{\text{start}}, t_i^{\text{end}})$$

- Si disponemos de algún conocimiento sobre el evento i, podríamos cambiar esta previa uniforme por otra más informativa.

Modelizar la incertidumbre en la localización temporal

En este caso no disponemos de eventos con incertidumbre temporal, por lo que introducimos esta incertidumbre mediante simulación:

```
# Inserting NAs in some time_events and adding temporal interval
set.seed(123)
unc_events=sort(sample(1:length(time_events),size=0.4*length(time_events)))
time_events_start=time_events
time_events_end=time_events
for (j in unc_events){
  time_events_start[j]=time_events[j]-rexp(1,rate=1/3)
  time_events_end[j]=time_events[j]+rexp(1,rate=1/3)
  if (time_events_start[j]<0){time_events_start[j]=0}
  if (time_events_end[j]>365){time_events_end[j]=364.5}
}
head(cbind(time_events,time_events_start,time_events_end))
     time_events time_events_start time_events_end
[1,]   0.1442677         0.1442677       0.1442677
[2,]   0.1542553         0.1542553       0.1542553
[3,]   0.1693016         0.1693016       0.1693016
[4,]   0.2442750         0.0000000       2.4952762(*)
[5,]   0.3117384         0.0000000       1.3964637(*)
[6,]   0.3559720         0.3559720       0.3559720
```

Modelizar la incertidumbre en la localización temporal

Hacemos la prueba de incluir esta incertidumbre con el modelo que incluye el término $\exp(-\frac{1}{\omega}|t - 200|)$ para la parte temporal.

Solamente necesitamos incorporar el siguiente código en el modelo (en el caso de usar el efecto aleatorio $\varepsilon_{W(t)}$ se necesita calcular el número de semana al que corresponde t_i en cada simulación):

```
for (i in 1:N_events) {
    time_events[i] ~ dunif(time_events_start[i],time_events_end[i])
}
```

Además, debemos incluir un valor inicial para `time_events`, ya que ahora es una variable del modelo, y los vectores `time_events_start` y `time_events_end` como constantes.

Modelizar la incertidumbre en la localización temporal

Podemos comprobar algunos resultados:

```
mcmc.output$summary[grep("time_events",rownames(mcmc.output$summary))[1:6],]
                     Mean    Median   St.Dev.  95%CI_low 95%CI_upp
time_events[1] 0.1442677 0.1442677 0.0000000 0.14426765 0.1442677
time_events[2] 0.1542553 0.1542553 0.0000000 0.15425530 0.1542553
time_events[3] 0.1693016 0.1693016 0.0000000 0.16930158 0.1693016
time_events[4] 1.2120674 1.1872063 0.7282499 0.05974255 2.4342166
time_events[5] 0.7065529 0.6950720 0.4052651 0.03236108 1.3708265
time_events[6] 0.3559720 0.3559720 0.0000000 0.35597202 0.3559720
```

Observamos que para los eventos 1, 2, 3 y 6, para los cuales no hay incertidumbre, el valor de t_i permanece inalterado. Para los eventos 4 y 5, en cambio, se estima el valor de t_i a partir de la previa.

Modelos self-exciting

- Hasta el momento, todos los modelos considerados eran **separables** en espacio y tiempo. Esto es, la función de intensidad podía expresarse como: $\lambda(x, y, t) = \lambda_S(x, y)\lambda_T(t)$.
- Esta suposición es con frecuencia poco realista, ya que impide detectar la **interacción espacio-tiempo**.
- Existen también modelos de proceso puntual que sí permiten tener en cuenta esta interacción, como por ejemplo el proceso puntual self-exciting. En este caso, la función de intensidad adopta la forma:

$$\lambda(\boldsymbol{x}, t|\mathcal{H}_t) = \lambda_0 + \sum_{j:t_j<t} \theta e^{-\frac{|t-t_j|}{\varphi_T}} e^{-\frac{\|\boldsymbol{x}-\boldsymbol{x}_j\|}{\varphi_S}},$$

donde $\mathcal{H}_t = \{(\boldsymbol{x}_i, t_i) : t_i < t\}$ representa *la historia del proceso* en tiempo t.

Modelos self-exciting

- Este tipo de modelo puede programarse como los vistos hasta ahora, aunque resulta algo más complejo.
- En concreto, requieren definir matrices 1) con las distancias espaciales/temporales entre eventos y 2) con las distancias espaciales/temporales entre los eventos y la partición de la ventana espacio-temporal.
- En el siguiente repositorio de GitHub puede encontrarse una implementación de la versión básica de este modelo y alguna extensión:

  `https://github.com/albrizre/SmoothProdSelfExc`

Evaluación del modelo: Análisis residual

- Evaluar si un modelo de tipo proceso puntual ajusta bien un patrón puntual observado resulta más difícil que en el caso de los modelos con variable respuesta numérica.
- Nos centramos ahora en mostrar cómo se realiza un **análisis residual espacial** para valorar el ajuste del modelo a nivel espacial. Seguimos la propuesta de Baddeley et al. (2005).
- Existe también un tipo de análisis residual temporal que podéis consultar en Briz-Redón and Mateu (2023).

Evaluación del modelo: Análisis residual

Propuesta de Baddeley et al. (2005) como residuo de un proceso puntual espacial:

$$SR_{\hat{\Theta}}(c) = \lambda^*(c) - \lambda^\dagger(c)$$

donde $\lambda^*(c)$ es una estimación no paramétrica de la función de intensidad, y $\lambda^\dagger(c)$ una versión suavizada del número estimado de evento en c, de acuerdo con el modelo ajustado.

- El término $\lambda^*(c)$ se calcula esencialmente como:

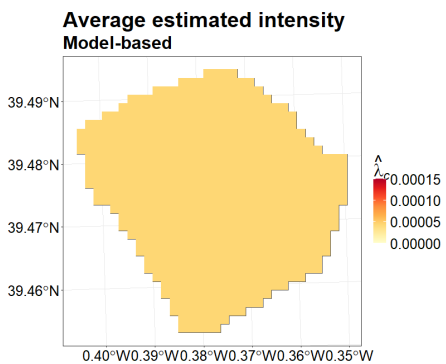
$$\lambda^*(c) = \sum_{i=1}^{n} \kappa(c - x_i),$$

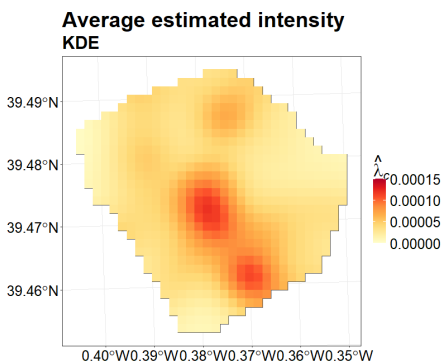
  donde $\kappa(\cdot)$ es una función *kernel* que presenta las propiedades de una función de densidad (no negativa e integra 1). La desviación típica de esta función se denomina *bandwidth* (ancho de banda).

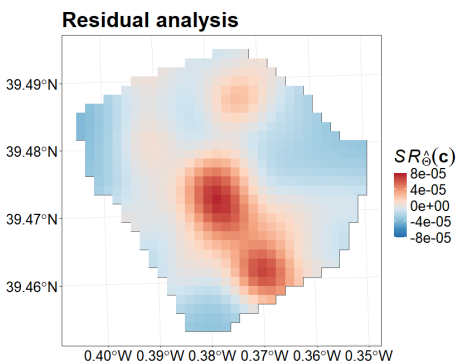
- El término $\lambda^\dagger(c)$ se calcula de forma similar pero sobre los valores estimados en base al modelo. Ahora lo vemos con código.

Evaluación del modelo: Análisis residual

Estos son los mapas correspondientes a $\lambda^*(c)$, $\lambda^\dagger(c)$ y su diferencia (el residuo, $SR_{\hat{\Theta}}(c)$) para el proceso Poisson homogéneo con el que empezamos el cursillo:







Evaluación del modelo: Análisis residual

Este código permite realizar este tipo de análisis:

```
# Residual analysis (spatial) for a given model (we load it)
load("Outputs/Poisson_hom.rda")
# First we obtain a non-parametric estimate of the intensity
pattern=ppp(x = events_study_area@coords[,1],y = events_study_area@coords[,2],
            window = as.owin(sf::st_union(grid_val_sf)))
bw_select=spatstat.explore::bw.scott(pattern,isotropic = T)
funcion_density=densityfun(pattern,sigma = bw_select)
centroids=sf::st_coordinates(sf::st_centroid(grid_val_sf,byid = T))
density_values=funcion_density(centroids[,1],centroids[,2])
# Second, we compute the smoothed values of the fitted intensity
X=ppp(x=centroids[,1],y=centroids[,2],
      window = as.owin(sf::st_union(grid_val_sf)))
lambdas=as.data.frame(mcmc.output$summary$chain1[grep("lambda_int",
                      rownames(mcmc.output$summary$chain1)),])
aux=rep(1:647,366)
lambdas$Cell=aux
lambdas_avg_cell=sqldf::sqldf("SELECT Cell, SUM(Mean) as Model from lambdas
                              GROUP BY Cell")
marks(X)=lambdas_avg_cell
smooth_X=Smooth(X, bw_select,at = "points")[,2]
```

Bibliografía I

Baddeley, A., Turner, R., Møller, J., and Hazelton, M. (2005). Residual analysis for spatial point processes (with discussion). *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 67(5):617–666.

Briz-Redón, Á. (2024). A Bayesian aoristic logistic regression to model spatio-temporal crime risk under the presence of interval-censored event times. *Journal of Quantitative Criminology*, 40(3):621–644.

Briz-Redón, Á. and Mateu, J. (2023). A mechanistic bivariate point process model for crime pattern analysis. *Stat*, 12(1):e537.

Daley, D. J. and Vere-Jones, D. (2002). *An Introduction to the Theory of Point Processes. Volume I: Elementary Theory and Methods*. Springer New York.

Diggle, P. J. (2013). *Statistical Analysis of Spatial and Spatio-Temporal Point Patterns*. Chapman and Hall/CRC.

Bibliografía II

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2013). *Bayesian Data Analysis*. CRC Press.

Gelman, A. and Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statistical Science*, 7(4):457–472.

Kempf, M. (2020). Fables of the past: landscape (re-) constructions and the bias in the data. *Documenta Praehistorica*, 47:476–492.

Ntzoufras, I. (2011). *Bayesian modeling using WinBUGS*. John Wiley & Sons.

Stoner, O., Economou, T., and da Silva, G. D. M. (2019). A hierarchical framework for correcting under-reporting in count data. *Journal of the American Statistical Association*.

Wood, S. N. (2017). *Generalized additive models: An introduction with R*. Chapman and Hall/CRC.