UNIVERSIDAD NACIONAL DE COLOMBIA

Modelado de datos en áreas

Estadística espacial

Daniela Arbeláez Montoya
Jefferson Gamboa Betancur
Jean Paul Piedrahita García

Universidad Nacional de Colombia
Ciencias, Escuela de Estadística
Medellín, Colombia
2021

Índice

1. Introducción

A lo largo del desarrollo de este documento, se mostrará la construcción de vecinos y los pesos que se pueden aplicar a los vecindarios. Una vez que este importante y a menudo exigente prerrequisito esté en su lugar, se procede a buscar formas de medir la autocorrelación espacial.

Si bien las pruebas se basan en modelos de procesos espaciales, primero se examinan las pruebas y solo posteriormente se pasa al modelado. También es interesante mostrar cómo se puede introducir la autocorrelación espacial en datos independientes, de modo que se puedan realizar simulaciones.

El conjunto de datos con el que se trabajará en esta ocasión, contiene 281 distritos censales para ocho condados centrales del estado de Nueva York complementado con límites de tramo. El área tiene una extensión de unos 160 km de norte a sur y de 120 km de este a oeste.

```
> library(rgdal)
> library(sf)
> library(spdep)
```

```
> NY8 <- readOGR("Base de datos", "NY8_utm18")
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "G:\Mi unidad\Universidad Nacional de Colombia\13. Treceavo semestre\Estadísti
## with 281 features
## It has 17 fields
```

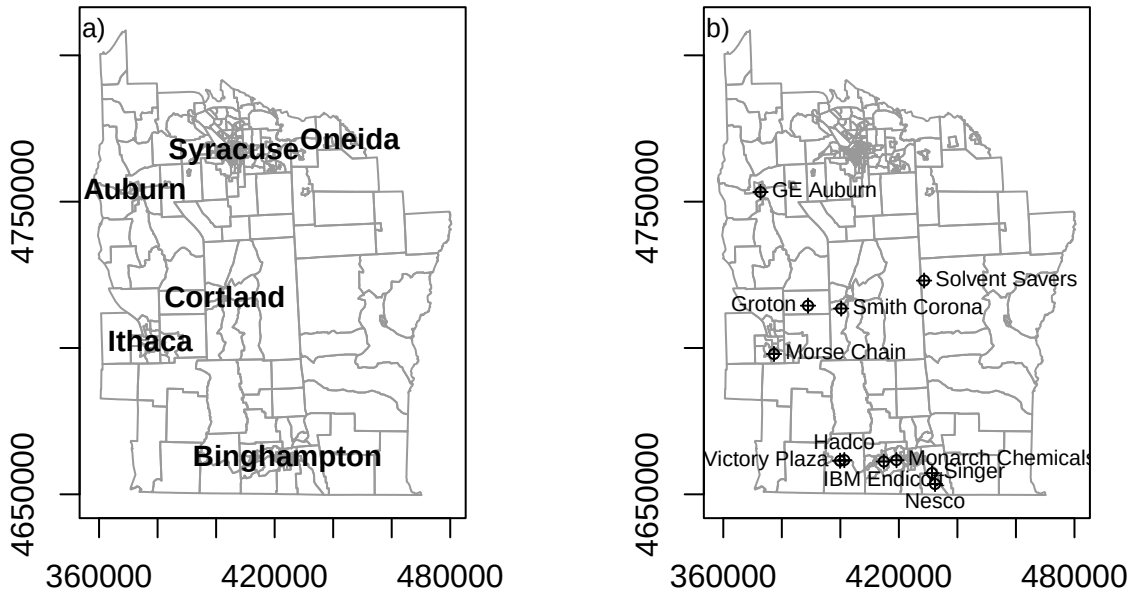
```
> TCE <- readOGR("Base de datos", "TCE")
```

```
## OGR data source with driver: ESRI Shapefile
## Source: "G:\Mi unidad\Universidad Nacional de Colombia\13. Treceavo semestre\Estadísti
## with 11 features
## It has 5 fields
```

```
> cities <- readOGR("Base de datos", "NY8cities")
```

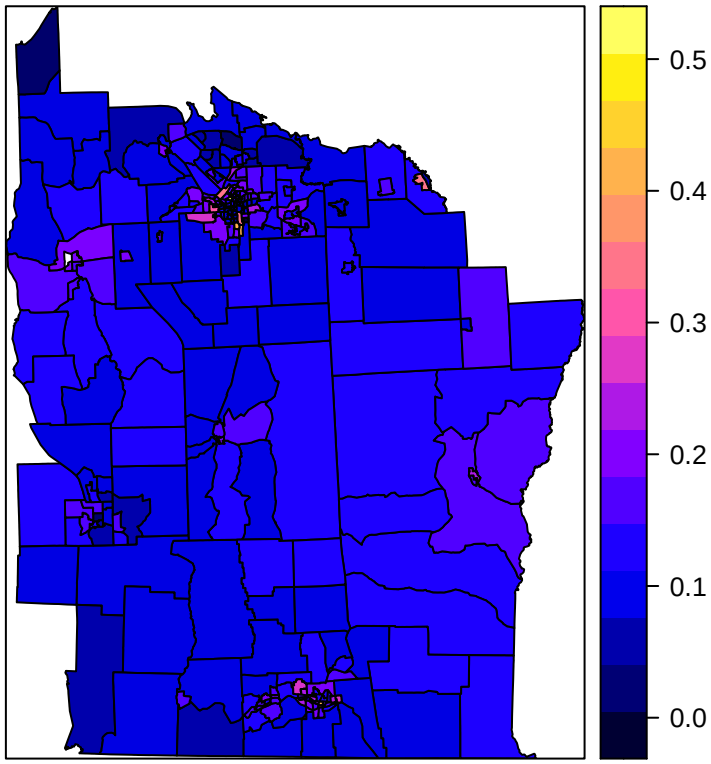
```
## OGR data source with driver: ESRI Shapefile
## Source: "G:\Mi unidad\Universidad Nacional de Colombia\13. Treceavo semestre\Estadísti
## with 6 features
## It has 1 fields
```

```
> par(mfrow=c(1,2))
> plot(NY8, border="grey60", axes=TRUE)
> text(coordinates(cities), labels=as.character(cities$names), font=2, cex=0.9)
> text(bbox(NY8)[1,1], bbox(NY8)[2,2], labels="a)", cex=0.8)
> plot(NY8, border="grey60", axes=TRUE)
> points(TCE, pch=1, cex=0.7)
> points(TCE, pch=3, cex=0.7)
> text(coordinates(TCE), labels=as.character(TCE$name), cex=0.7,
+  font=1, pos=c(4,1,4,1,4,4,4,2,3,4,2), offset=0.3)
> text(bbox(NY8)[1,1], bbox(NY8)[2,2], labels="b)", cex=0.8)
```

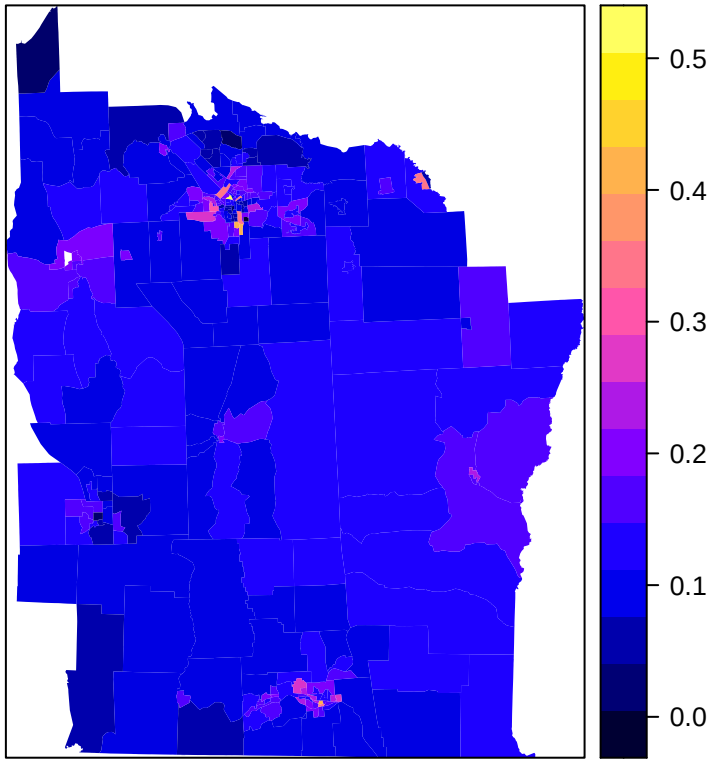


La figura *a*) muestra las principales ciudades en el área de estudio y *b*) la ubicación de 11 sitios de desechos peligrosos.

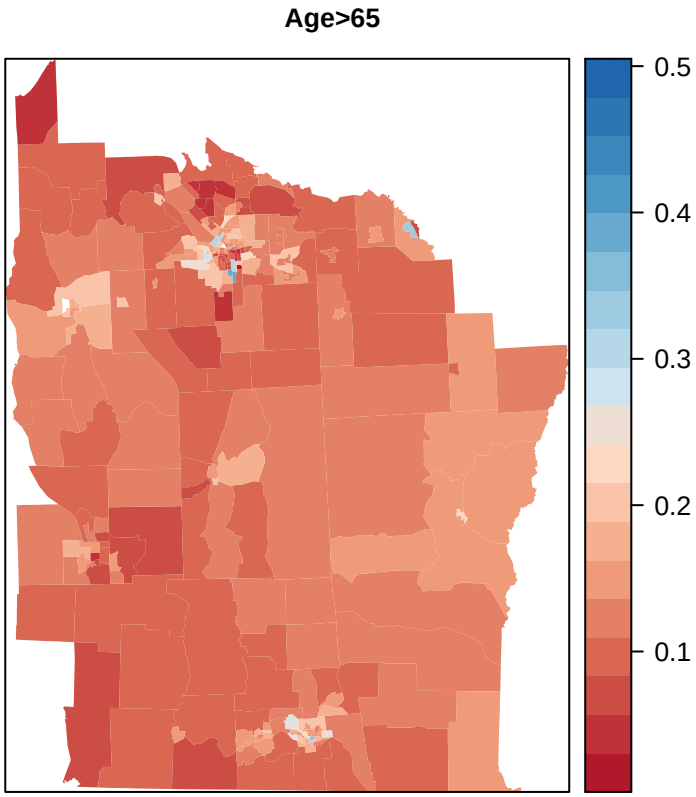
```
> spplot(NY8, c("PCTAGE65P"))
```




```
> spplot(NY8, c("PCTAGE65P"), col="transparent")
```



```
> library("RColorBrewer")
> #color palette creator function
> rds <- colorRampPalette(brewer.pal(8, "RdBu"))
> #get a range for the values
> tr_at <- seq(min(NY8$PCTAGE65P), max(NY8$PCTAGE65P), length.out=20)
> #create a color interpolating function taking the required
> #number of shades as argument
> tr_rds <- rds(20)
> #parameters
> # at - at which values colors change
> # col.regions - specify fill colors
> tr_pl <- spplot(NY8, c("PCTAGE65P"), at=tr_at, col="transparent",
+                 col.regions=tr_rds, main=list(label="Age>65", cex=0.8))
> plot(tr_pl)
```



2. Vecinos espaciales y peso espacial

La creación de ponderaciones espaciales es un paso necesario en el uso de datos de área, quizás solo para verificar que no haya patrones espaciales restantes en los residuos. El primer paso es definir a qué relaciones entre observaciones se les dará un peso distinto de cero, es decir, elegir el criterio de vecino que se usará; el segundo es asignar pesos a los enlaces vecinos identificados.

Tratar de detectar patrones en mapas de residuos visualmente no es una opción aceptable, por lo que se incluyen un montón de funciones en el paquete **spdep** para ayudar.

Las viñetas **"nb","CO69"** y **"sids"** en **spdep** incluyen discusiones sobre la creación y el uso de ponderaciones espaciales, y se pueden acceder a ellas de la siguiente manera:

```
> vignette("nb",package = "spdep")
```

2.1. Objetos vecinos

En el paquete spdep, las relaciones vecinas entre **n** observaciones están representadas por un objeto de clase **nb**. Es una lista de longitud n con los números de índice de vecinos de cada componente registrados como un vector entero. Si alguna observación no tiene vecinos, el componente contiene un número entero cero. También contiene atributos, típicamente un vector de identificadores de región de caracteres y un valor lógico que indica si las relaciones son simétricas. Los identificadores de región pueden usarse para verificar la integridad entre los datos mismos y el objeto vecino.

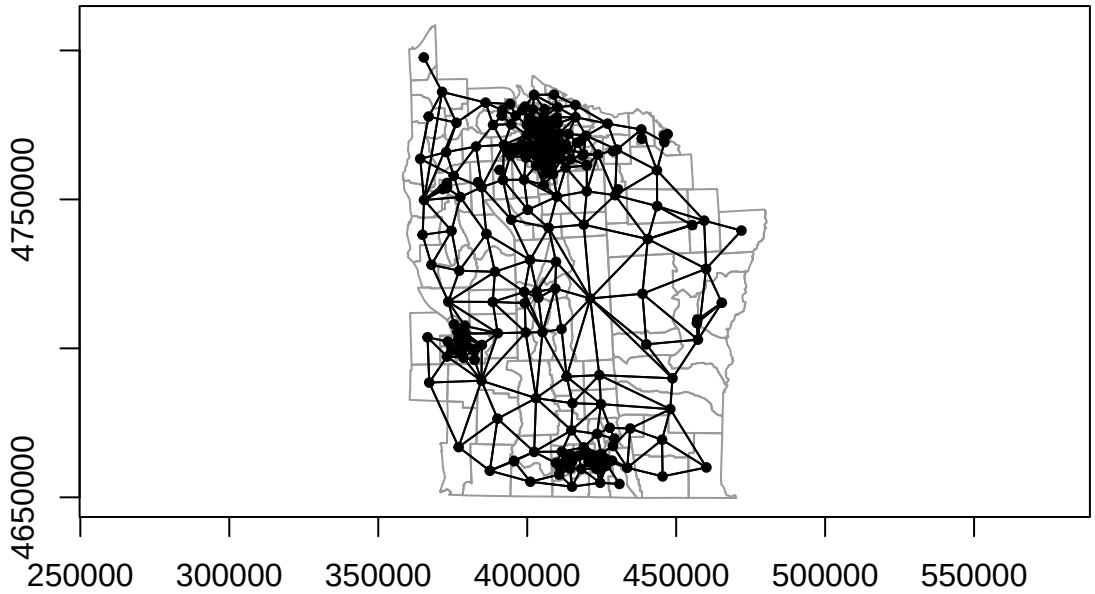
La función auxiliar **card** devuelve la cardinalidad del conjunto de vecinos para cada objeto, es decir, el número de vecinos.

```
> # reads a GAL lattice file into a neighbors list
> NY_nb <- read.gal("Base de datos/NY_nb.gal", region.id = row.names(NY8))
> summary(NY_nb)
```

```
## Neighbour list object:
## Number of regions: 281
## Number of nonzero links: 1522
## Percentage nonzero weights: 1.927534
## Average number of links: 5.41637
## Link number distribution:
##
```

```
##  1  2  3  4  5  6  7  8  9 10 11
##  6 11 28 45 59 49 45 23 10  3  2
## 6 least connected regions:
## 55 97 100 101 244 245 with 1 link
## 2 most connected regions:
## 34 82 with 11 links
```

```
> par(mfrow=c(1,1))
> plot(NY8, border="grey60", axes=TRUE)
> plot(NY_nb, coordinates(NY8), pch=19, cex=0.6, add=TRUE)
```



La figura muestra el gráfico vecino completo para el área de estudio de ocho condados.

Como ahora se tiene un objeto nb para examinar, se pueden presentar los métodos estándar para estos objetos. Hay métodos de impresión, resumen, diagrama y otros; el método de resumen presenta una tabla de la distribución del número de enlace, y tanto el método de impresión como el de resumen informan de la asimetría y la presencia de observaciones sin vecinos; la asimetría está presente cuando i es un vecino de j pero j no es un vecino de i.

Con motivos de simplicidad al mostrar como crear objetos vecinos, se trabaja con un subconjunto

del mapa que consta de los censos dentro de Syracuse, aunque los mismos principios se aplican al conjunto de datos completo.

```
> Syracuse <- NY8[NY8$AREANAME == "Syracuse city",]
> Sy0_nb <- subset(NY_nb, NY8$AREANAME == "Syracuse city")
> summary(Sy0_nb)
```

```
## Neighbour list object:
## Number of regions: 63
## Number of nonzero links: 346
## Percentage nonzero weights: 8.717561
## Average number of links: 5.492063
## Link number distribution:
##
##  1  2  3  4  5  6  7  8  9
##  1  1  5  9 14 17  9  6  1
## 1 least connected region:
## 164 with 1 link
## 1 most connected region:
## 136 with 9 links
```

Se crean varios objetos vecinos de otras maneras para usarlos mas adelante.Tres son k objetos vecinos mas cercanos, tomando k puntos mas cercanos como vecinos (k=1,2 y 4 en este caso); esto se adapta a lo largo del área de estudio, teniendo en cuenta las diferencias en las densidades de área. Naturalmente, en la mayoría de los casos,esto conduce a vecinos asimétricos, pero asegurará que todas las áreas tengan **k** vecinos.

```
> coords <- coordinates(Syracuse)
> IDs <- row.names(Syracuse)
> Sy8_nb <- knn2nb(knearneigh(coords, k = 1), row.names = IDs)
> Sy9_nb <- knn2nb(knearneigh(coords, k = 2), row.names = IDs)
> Sy10_nb <- knn2nb(knearneigh(coords, k = 4), row.names = IDs)
> dsts <- unlist(nbdists(Sy8_nb, coords))
> Sy11_nb <- dnearneigh(coords, d1 = 0, d2 = 0.75 * max(dsts),
+                       row.names = IDs)
```

El objeto k= también es útil para encontrar la distancia mínima a la que todas las áreas tienen un vecino basado en la distancia. Usando la función *nbdist*, se puede calcular una lista de vectores de distancias correspondientes al objeto vecino. El mayor valor será la distancia mínima necesaria para asegurarse de que todas las áreas estén vinculadas al menos a un vecino, por lo que se puede crear fácilmente un objeto con observaciones sin vecino estableciendo el umbral superior por debajo de este valor.

2.2. Objetos de ponderaciones espaciales

La ponderación espacial se puede ver como una lista de ponderaciones espaciales indexadas por una lista de vecinos entre i y j esel k-*ésimo* elemento del i-*ésimo* componente de la lista de ponderaciones, y k nos dice cual de los i-*ésimos* valores del componente de la lista de vecinos es igual a j. Si j no está presente en el i-*ésimo* componente de la lista de vecinos, j no es un vecino de i. Por ello algunas de las ponderaciones w_{ij} de la matriz de ponderaciones serán cero.

Una vez se establece la lista de los conjuntos de los vecinos en nuestra área de estudio, se procese a asignar los pesos espaciales, también es de tener en cuenta que se utiliza una notación bionaria cuando se sabe poco del proceso especial, dónde no hay una relación de vecino se pone 0 (cero) en caso contrario será la unidad.

La función `nb2listw` toma una lista de vecinos y lo convierte en un objeto de pesos. La conversión de los pesos se hace con un estilo W que se hace bajo una estandarización por fila para sumar la unidad. El método de impresión *(print)* para los objetos listw muestran las características de los vecinos subyacentes, el estilo de las ponderaciones espaciales y las constantes de las ponderaciones espaciales utilizadas en el cálculo de las pruebas de auto-correlación espacial. El componente de vecinos del objeto es el objeto nb subyacente, que proporciona la indexación del componente de ponderaciones.

```
> Sy0_lw_W <- nb2listw(Sy0_nb); Sy0_lw_W
```

```
## Characteristics of weights list object:
## Neighbour list object:
## Number of regions: 63
## Number of nonzero links: 346
## Percentage nonzero weights: 8.717561
## Average number of links: 5.492063
```

```
##
## Weights style: W
## Weights constants summary:
##    n   nn S0       S1      S2
## W 63 3969 63 24.78291 258.564
```

```
> names(Sy0_lw_W)
```

```
## [1] "style"      "neighbours" "weights"
```

```
> names(attributes(Sy0_lw_W))
```

```
## [1] "names"     "class"     "region.id" "call"      "GeoDa"
```

Para el `style = "W"`, los pesos varían entre la unidad divida por el mayor y el menor número de vecinos, y las sumas de los pesos para cada entidad de área son la unidad. Este estilo se puede interpretar como el **valor medio entre vecinos**. Los pesos de los enlaces que se originan en áreas con pocos vecinos son mayores que los que se originan en áreas con muchos vecinos, quizá aumentando las entidades de área en el borde del área de estudio sin querer. Esta representación ya no es simétrica, pero es similar tiende a ser simétrica.

```
> 1/rev(range(card(Sy0_lw_W$neighbours)))
```

```
## [1] 0.1111111 1.0000000
```

```
> summary(unlist(Sy0_lw_W$weights))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.1111  0.1429  0.1667  0.1821  0.2000  1.0000
```

```
> summary(sapply(Sy0_lw_W$weights, sum))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1       1       1       1       1       1
```

Las funciones:

- `card` cuenta el número de vecinos en cada región en la lista de vecinos.
- `range` toma el valor mínimo y el máximo.
- `rev` revierte el vector

- `unlist` saca la información o datos atómicos de una lista
- `sapply` aplica la función suma a cada uno de los vectores asociado a las áreas.

La configuración de `style = "B"` - *Binario* retiene un peso de unidad para cada relación de vecino, pero en este caso, las sumas de pesos de las áreas difieren según el número de áreas vecinas que tienen.

```
> Sy0_lw_B <- nb2listw(Sy0_nb, style = "B")
> summary(unlist(Sy0_lw_B$weights))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       1       1       1       1       1       1
```

```
> summary(sapply(Sy0_lw_B$weights, sum))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   4.500   6.000   5.492   6.500   9.000
```

El argumento `glist` se puede utilizar para pasar una lista de vectores de pesos generales correspondientes a las relaciones vecinas a `nb2list`. En este ejecicio se cree que la fuerza de las relaciones con los vecinos se atenúa con la distancia, por ello se estable en los pesos para que sean proporcionales ala distancia inversa entre los puntos que representan las áreas, usando `nbdists`para calcular las distancias para el objeto `nb` dado. Luego se usa `lapply` para invertir las distancias, obteniendo una estructura de pesos espaciales diferente a los anteriores. Si no se tiene ninguna razón para asumir más conocimiento sobre las relaciones con los vecinos que su existencia o ausencia, este paso es potencialmente engañoso. Si se sabe que el flujo de desplazamiento describen la estructura de las ponderaciones mejor que la alternativa binaria, puede valer la pena utilizarlas como ponderaciones generales; Sin embargo, puede haber problemas de simetría, porque tales flujos a diferencia de las distancias inversas, rara vez son simétricas.

```
> dsts <- nbdists(Sy0_nb, coordinates(Syracuse))
> idw <- lapply(dsts, function(x) 1/(x/1000))
> Sy0_lw_idwB <- nb2listw(Sy0_nb, glist = idw, style = "B")
> summary(unlist(Sy0_lw_idwB$weights))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.3886  0.7374  0.9259  0.9963  1.1910  2.5274
```

```
> summary(sapply(Sy0_lw_idwB$weights, sum))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.304   3.986   5.869   5.471   6.737   9.435
```

La función:

- `nbdists` devuelve las distancias euclidianas dadas por un vector de enlaces de vecinos (un objeto nb).
- `nb2listw` toma una lista de vecinos y lo convierte en un objeto de pesos.
- `lapply` aplica a cada distancia euclidiana el inverso.

La siguiente figura muestra tres representaciones de ponderaciones espaciales para Syracuse mostradas como matrices. La imagen `style = "W"` de la izquierda es asimétrica, con colores más oscuros ue muestran pesos más gandes para áreas con pocos vecinos. Los otros dos paneles son simétricos, pero expresan diferentes suposiciones sobre las fortalezas de las relaciones con los vecinos.

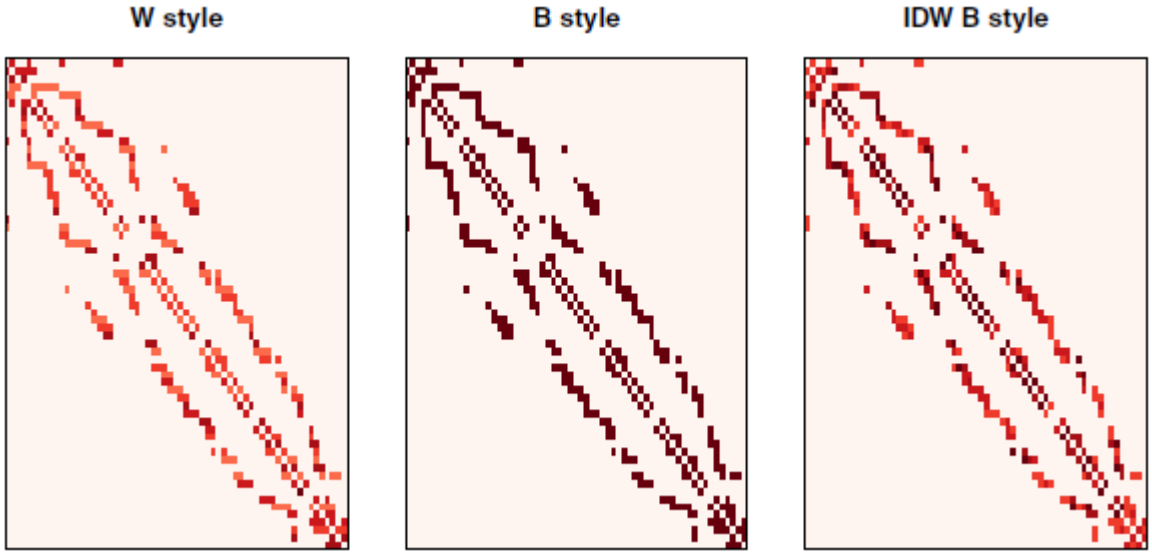


Figura 1: Tres representaciones de ponderaciones espaciales para Syracuse

La función `nb2listw` permite manejar listas de vecinos con áreas sin vecinos. Esto es debido porque la representación del conjunto vacío sea cero y debería ser representado como `NA` pero esto generaría problemas más adelante.

Es por esta razón que el argumento predeterminado es `cero.policy=FALSE`, lo que genera un error cuando se proporciona un argumento `nb` con áreas sin vecinos. Con el argumento `TRUE`permite la

creación del objeto de ponderaciones espaciales, con ponderaciones cero.

```
> Sy0_lw_D1 <- nb2listw(Sy11_nb, style = "B")
```

```
## Error in nb2listw(Sy11_nb, style = "B"): Empty neighbour sets found
```

```
> Sy0_lw_D1 <- nb2listw(Sy11_nb, style = "B", zero.policy = TRUE)
> print(Sy0_lw_D1, zero.policy = TRUE)
```

```
## Characteristics of weights list object:
## Neighbour list object:
## Number of regions: 63
## Number of nonzero links: 230
## Percentage nonzero weights: 5.794911
## Average number of links: 3.650794
## 2 regions with no links:
## 154 168
##
## Weights style: B
## Weights constants summary:
##    n   nn  S0  S1   S2
## B 61 3721 230 460 4496
```

Un problema paralelo de los conjuntos de datos con valores perdidos en las variables pero con ponderaciones espaciales especificadas se aborda mediante el método `subset.listw`, que vuelve a generar las ponderaciones para el subconjunto de áreas dado, por ejemplo, dado por `complete.cases`. Sabiendo qué observaciones están incompletas, los vecinos subyacentes y las ponderaciones se pueden subdividir en algunos casos, con el objetivo de evitar la propagación de los valores `NA` al calcular los valores con retraso espacial. Muchas pruebas y funciones de ajuste de modelos pueden llevar a cabo esto internamente si se establece el indicador de argumento apropiado, aunque el analista cuidadoso preferirá subconjuntos de los datos de entrada y los pesos antes de probar o modelar.

2.3. Manejo de objetos de ponderaciones espaciales

Hay varios paquetes contribuidos que brindan soporte para matrices dispersas, entre las cuales `Matrix` es un paquete recomendado. La envoltura `as_dgRMatrix_listw` convierte un objeto

`listw` en la matriz dispersa de formato orientado a filas comprimidas ordenadas por `Matrix`, como un objeto `dgRMatrix`, una subclase de la clase virtual `RsparseMatrix`. Es más fácil hacer una matriz dispersa orientada a filas a partir de un objeto de ponderaciones espaciales, ya que las ponderaciones están orientadas a filas. Una función que se usa mucho dentro de las funciones de prueba y ajuste de modelos es `listw2U`, que devuelve un objeto `listw` simétrico que representa la matriz de ponderaciones espaciales $\frac{1}{2}(W + W^T)$

Los objetos vecinos y de pesos pueden ser producidos en otros software e importarse a R y además se pueden exportar sin dificultad. Como ejemplo se han generado algunos archivos de GeoDa[1] a partir de distritos censales de Syracuse escritos como un shapefile, con el centroide utilizando aquí almacenado en el marco de datos. Los dos primeros son para vecinos de contigüidad. Estos archivos denomidos en formato GAL contienen solo información de vecinos y se describen en detalle en el archivo de ayuda que acompaña a la función `read.gal`

```
> Sy14_nb <- read.gal("Base de datos/Sy_GeoDa1.GAL")
> isTRUE(all.equal(Sy0_nb, Sy14_nb, check.attributes = FALSE))
```

```
## [1] TRUE
```

La función `write.nb.gal` se utiliza para escribir archivos en formato `GAL` a partir de `nb` objetos. GeoDa también crea archivos en formato `GWT`, descritos en la documentación de GeoDa y el archivo de ayuda, que también contienen información de distancia para el enlace entre las áreas, y se almacenan en una representación dispersa de tres columnas. Se pueden leer usando `read.gwt2nb`, aquí para un esquema de cuatro vecinos más cercanos, y solo usando los enlaces de vecinos.

```
> Sy16_nb <- read.gwt2nb("Base de datos/Sy_GeoDa4.GWT")
> isTRUE(all.equal(Sy10_nb, Sy16_nb, check.attributes = FALSE))
```

```
## [1] TRUE
```

Un conjunto similar de funciones está disponible para intercambiar pesos espaciales con la Biblioteca de Econometrics Espacial. La representación dispersa de pesos es similar al formato `GWT` y se puede importar usando `read.dat2listw`.

La exportación a tres formatos diferentes pasa por la función `listw2sn`, que convierte un objeto de pesos espaciales en una representación dispersa de tres columnas. El marco de datos de salida se puede escribir como un archivo en formato `GWT` con `write.sn2gwt` o como una representación

[1] https://sgsup.asu.edu/geodacenter-redirect

de texto de una matriz dispersa para Matlab™ con `write.sn2dat`. Los archivos escritos con `write.sn2gwt` se pueden leer en Stata™ con `spmat import W using`. Existe una función llamada `listw2WB` para crear una lista de pesos espaciales para `WinBUGS`, que se escribirá en un archivo usando `dput`. Para objetos vecinos, se puede usar `nb2WB`, estableciendo todos los pesos a la unidad. De manera similar, un objeto vecino se puede exportar a un archivo con `nb2INLA`, para pasar datos al argumento del **gráfico** para el modelo `"bym"` en el ajuste usando `inla`. También es posible utilizar `nb2gra` en el paquete `BayesX` para convertir `nb` objetos al formato gráfico `gra`.

El `mat2listw` se puede utilizar para revertir el proceso, cuando una matriz de ponderaciones se ha leído en `R` y necesita convertirse en un vecino y un objeto de lista de ponderaciones. Desafortunadamente, esta función no establece el estilo del objeto `listw` a un valor conocido, utilizando `M` para señalar esta falta de conocimiento. Entonces es habitual reconstruir el objeto `listw`, tratando el componente de vecinos como un objeto `nb`, el componente de pesos como una lista de pesos generales y estableciendo el estilo en la función `nb2listw` directamente. Se utilizó para la importación inicial de las contigüidades de ocho condados, como se muestra en detalle en la página de ayuda `NY_data` proporcionada con `spdep`.

Finalmente, existe una función `nb2lines` para convertir listas de vecinos en objetos `SpatialLinesDataFrame`, dadas las coordenadas de los puntos que representan las áreas. Esto permite trazar objetos vecinos de una manera alternativa y, si es necesario, exportarlos como `shapefiles`.

2.4. Uso de pesos para simular la autocorrelación espacial

En la siguiente figura se utilizó `listw2mat`para convertir un objeto de pesos espaciles en una matriz densa para su visualización. La misma función se utiliza para contruir una representación densada de l amatriz $(I - \rho\omega)$ para simular la autocorrelación espacial dentro de la función `invlrW`, donde W es una matriz de ponderaciones, ρ es un coeficiente de autocorrelación espacial e I es la matriz de identidad. Este enfoque no impone condiciones estrictas para invertir las matrices a no ser de que sea una matriz singular, y sólo se aplica a simulaciones de un proceso autoregresivo simultáneo.

Para la simulación se utilizó un vector de número aleatorios iguales al número de secciones censales en Siracusa, usando las ponderaciones de contigüidad estandarizadas por filas para introducir la autocorrelación.

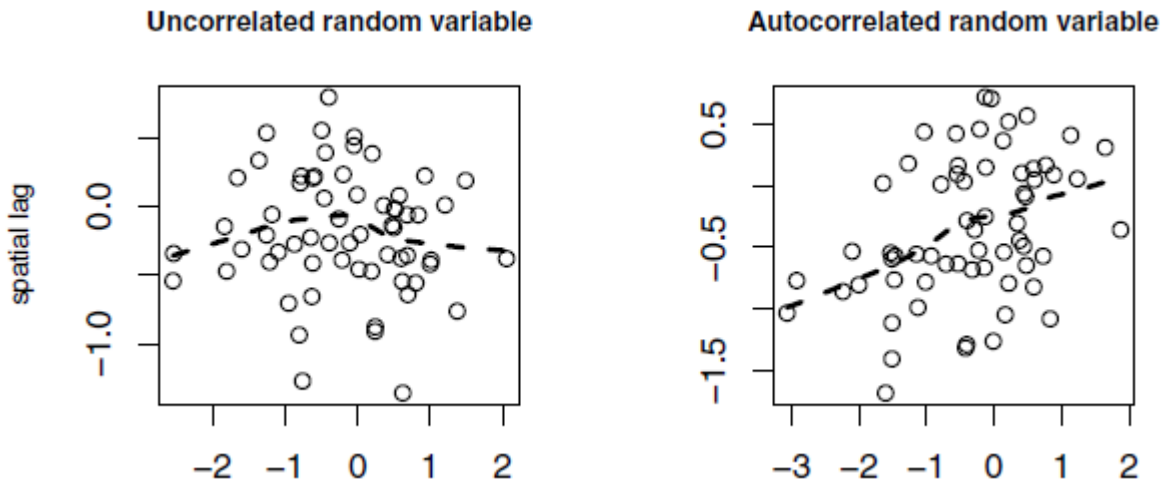


Figura 2: Simulación de la autocorrelación espacial: gráficos de retardo espacial, que muestran una línea más suave ponderada localmente

```
> set.seed(987654)
> n <- length(Sy0_nb)
> uncorr_x <- rnorm(n)
> rho <- 0.5
> autocorr_x <- invIrW(Sy0_lw_W, rho) %*% uncorr_x
```

El resultado que se muestra en la figura anterior, muestra el retardo espacial de la variable original no correlacionada versus con la variable aleatoria autocorrelacionada, donde ahora tiene una relación positiva entre los valores del tramo y el retardo espacial.

El método de retraso para objetos `listw` crea valores de "retraso espacial": $lag(y_i) = \sum_{j \in N_i} \omega_{ij} y_j$ para valores observados y_i; N_i es el conjunto de vecinos de i. Si el estilo del objeto de ponderaciones es la estandarización de filas, los valores de $lag(y_i)$ serán promedios sobre los conjuntos de vecinos para cada i, más bien como una ventana móvil definida por N_i e incluyendo valores ponderados por ω_{ij}.

Para el análisis de datos de un área depende de manera crucial la construcción de los pesos espaciales, por ello se ha tomado el tiempo para describir las opciones que enfrenta el investigador. Ahora se procederá a modelar y probar la autocorrelación espacial utilizando supuestos sobre los procesos espaciales subyacentes.

3. Prueba de autocorrelación espacial

Ahora que se tiene variedad de formas de construir pesos espaciales, se puede comenzar a usarlos para probar la presencia de autocorrelación espacial. Antes de comenzar, es muy útil revisar las suposiciones que se hacen en las pruebas; se usará la I de Moran como ejemplo, pero las consecuencias también se aplican a otras pruebas.

Las pruebas suponen que el modelo medio de los datos elimina los patrones espaciales sistemáticos de los datos.Este error de especificación del modelo medio no es en absoluto infrecuente y puede ser inevitable cuando no se dispone de observaciones sobre las variables necesarias para especificarlo correctamente.

Otro problema que puede surgir es que los pesos espaciales que usamos para las pruebas no son los que generaron la autocorrelación. Esto es un reflejo de la especificación incorrecta del modelo de la varianza de los residuos del modelo medio, que también puede incluir hacer supuestos de distribución que no son apropiados para los datos, por ejemplo, asumiendo homocedasticidad o parámetros de forma regular (asimetría y curtosis). Algunos de estos pueden abordarse transformando los datos y utilizando la estimación ponderada, pero en cualquier caso, se necesita cuidado al interpretar la autocorrelación espacial aparente que en realidad puede provenir de una especificación incorrecta.

Dado que se sabe que la variable simulada anteriormente, se extrae al azar de la distribución normal, se puede manipular para ver qué sucede con los resultados de las pruebas en diferentes condiciones. La prueba I de Moran , se calcula entonces como la razón del producto de la variable de interés y su rezago espacial, con el producto cruzado de la variable de interés y se ajusta a los pesos espaciales utilizados:

$$I = \frac{n}{\sum_{i=1}^{n}\sum_{j=1}^{n}\omega_{ij}} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n}\omega_{ij}\left(y_i-\overline{y}\right)\left(y_j-\overline{y}\right)}{\sum_{i=1}^{n}\left(y_i-\overline{y}\right)^2},$$

donde y_i es la i-ésima observación, $\overline{y}$ es la media de la variable de interés y w_{ij} es el peso espacial del vínculo entre i y j.

Los resultados del test para cinco configuraciones, se recopilan a continuación; la primera columna contiene el valor observado de I, la segunda es el valor esperado (la cual es $-1/(n-1)$ para casos de media centrada), la tercera es la varianza del estadístico bajo aleatorización, a continuación la desviación estándar y finalmente, el valor-p de la prueba para la alternativa $I > E(I)$.

	I	$E(I)$	$var(I)$	*St. deviate*	$p-value$
uncorr_x	-0.03329	-0.01613	0.00571	-0.227	0.59
autocorr_x	0.2182	-0.0161	0.0057	3.1	0.00096
autocorr_x k=1	0.1921	-0.0161	0.0125	1.86	0.031
trend_x	0.23747	-0.01613	0.00575	3.34	0.00041
lm(trend_x et)	-0.0538	-0.0309	0.0054	-0.312	0.62

```
> moran.test(uncorr_x, listw=Sy0_lw_W)

##
##  Moran I test under randomisation
##
## data:  uncorr_x
## weights: Sy0_lw_W
##
## Moran I statistic standard deviate = -0.22698, p-value = 0.5898
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##      -0.033287124      -0.016129032       0.005714128
> moran.test(autocorr_x, listw=Sy0_lw_W)

##
##  Moran I test under randomisation
##
## data:  autocorr_x
## weights: Sy0_lw_W
##
## Moran I statistic standard deviate = 3.1027, p-value = 0.0009588
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##       0.218178182      -0.016129032       0.005702793
```

```
> moran.test(autocorr_x, listw=nb2listw(Sy9_nb, style="W"))

##
##  Moran I test under randomisation
##
## data:  autocorr_x
## weights: nb2listw(Sy9_nb, style = "W")
##
## Moran I statistic standard deviate = 1.8619, p-value = 0.03131
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##        0.19207401       -0.01612903        0.01250486
```

```
> et <- coords[,1] - min(coords[,1])
> trend_x <- uncorr_x + 0.00025 * et
> moran_t <- moran.test(trend_x, listw=Sy0_lw_W)
> moran_t1 <- lm.morantest(lm(trend_x ~ et), listw=Sy0_lw_W)
```

Esto demuestra lo importante que puede ser entender que las pruebas de autocorrelación espacial también pueden reaccionar a un modelo mal especificado de la media, y que la omisión de una variable espacialmente modelada de la función media 'parecerá' como autocorrelación espacial a las pruebas.

3.1. Pruebas globales

Quizá la prueba global mas común es `moran.testy` por esta razón se continuará usandola. Otras pruebas globales implementadas en el paquete `spdep` incluyen la C de Geary (`geary.test`), la G global de GetisOrd (`gobalG.test`) y la prueba de Mantel de producto cruzado general espacial, que incluye la I de Moran, la C de Geary y la variante Sokal de la C de Geary como formas alternativas (`sp.mantel.mc`).

Todos estos test son para variables continuas, con `moran.test()` que tiene un argumento para usar un ajuste para una variable continua clasificada.

También hay pruebas de recuento de combinaciones para variables categóricas, con la variable de

interés representada como un factor: `joincount.test()` para combinaciones del mismo color, `joincount.multi()` para combinaciones del mismo y de diferente color.

El resultado de los test es una desviación estándar que se compara con la distribución Normal para encontrar el valor de probabilidad del estadístico observado bajo la hipótesis nula H_0 de no dependencia espacial para los pesos espaciales elegidos; la mayoría de veces la prueba es unilateral, con la hipótesis alternativa H_1 de que el estadístico observado es significativamente mayor que su valor esperado.

¿Cómo proceder si algunas entidades del área no tienen vecinos? De forma predeterminada, las funciones de prueba en `spdep` no aceptan ponderaciones espaciales con entidades sin vecinos a menos que el argumento `zero.policy` se establezca como **TRUE**.

```
> moran.test(NY8$Cases, listw = nb2listw(NY_nb))
```

```
##
##  Moran I test under randomisation
##
## data:  NY8$Cases
## weights: nb2listw(NY_nb)
##
## Moran I statistic standard deviate = 3.9778, p-value = 3.477e-05
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##       0.146882990      -0.003571429       0.001430595
```

El test anterior se realiza para probar autocorrelacion del número de casos de Leucemia por tramo censal, utilizando el estilo de ponderación espacial predeterminado de la estandarización de filas, y utilizando la hipótesis de aleatorización analítica para calcular la varianza de la estadística. El resultado, como se observa, es que el patrón espacial de la variable de interés es significativo.

Si añadimos el parámetro `style="B"` para hacer que todos los pesos sean iguales, se observa que el valor de la probabilidad resultante se reduce al rededor de 20 veces. Además, se sabe que por defecto `moran.test` asume aleatorización introduciendo un término de corrección basado en la curtosis de la variable de interés.

Cuando el valor de curtosis corresponde al de una variable normalmente distribuida, las dos hipótesis dan la misma varianza. En este caso hay una pequeña diferencia, pero ambas pruebas arrojan resultados similares.

```
> lw_B <- nb2listw(NY_nb, style = "B")
> moran.test(NY8$Cases, listw = lw_B)

##
##  Moran I test under randomisation
##
## data:  NY8$Cases
## weights: lw_B
##
## Moran I statistic standard deviate = 3.1862, p-value = 0.0007207
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##       0.110387402      -0.003571429       0.001279217
> moran.test(NY8$Cases, listw = lw_B, randomisation = FALSE)

##
##  Moran I test under normality
##
## data:  NY8$Cases
## weights: lw_B
##
## Moran I statistic standard deviate = 3.1825, p-value = 0.0007301
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##       0.110387402      -0.003571429       0.001282228
```

Es útil mostrar aquí que la prueba estándar en condiciones de normalidad es, de hecho la misma prueba de Moran para los residuos de regresión del modelo que incluye solo el intercepto.

```
> lm.morantest(lm(Cases ~ 1, NY8), listw = lw_B)

##
##  Global Moran I for regression residuals
##
## data:
## model: lm(formula = Cases ~ 1, data = NY8)
## weights: lw_B
##
## Moran I statistic standard deviate = 3.1825, p-value = 0.0007301
## alternative hypothesis: greater
## sample estimates:
## Observed Moran I      Expectation         Variance
##      0.110387402     -0.003571429      0.001282228
```

Usando la misma construcción, también podemos usar la aproximación de Saddlepoint en lugar de la suposición analítica normal, y una prueba exacta, ya que estos métodos son mucho más exigentes desde el punto de vista informático.

```
> lm.morantest.sad(lm(Cases ~ 1, NY8), listw = lw_B)

##
##  Saddlepoint approximation for global Moran's I (Barndorff-Nielsen
##  formula)
##
## data:
## model:lm(formula = Cases ~ 1, data = NY8)
## weights: lw_B
##
## Saddlepoint approximation = 2.9929, p-value = 0.001382
## alternative hypothesis: greater
## sample estimates:
## Observed Moran I
##        0.1103874
```

```
> lm.morantest.exact(lm(Cases ~ 1, NY8), listw = lw_B)

##
##  Global Moran I statistic with exact p-value
##
## data:
## model:lm(formula = Cases ~ 1, data = NY8)
## weights: lw_B
##
## Exact standard deviate = 2.9923, p-value = 0.001384
## alternative hypothesis: greater
## sample estimates:
## [1] 0.1103874
```

También podemos utilizar una prueba de Monte Carlo, una prueba Bootstrap de permutación, en la que los valores observados se asignan aleatoriamente a tractos y la estadística de interés calcula los tiempos `nsim`. Dado que, en el caso global en contraste con el local, tenemos suficientes observaciones y podemos repetir esta permutación potencialmente muchas veces sin repetición.

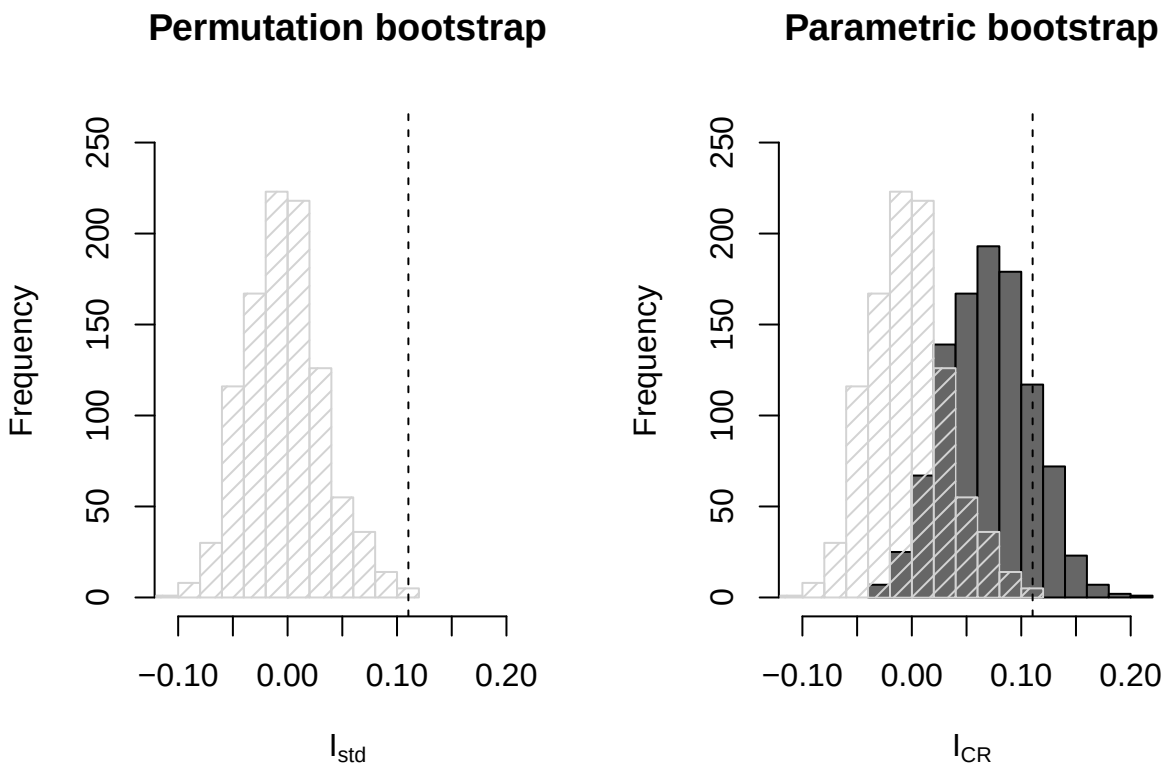
```
> # Monte Carlo test
> set.seed(1234)
> bperm <- moran.mc(NY8$Cases, listw = lw_B, nsim = 999)
> bperm

##
##  Monte-Carlo simulation of Moran I
##
## data:  NY8$Cases
## weights: lw_B
## number of simulations + 1: 1000
##
## statistic = 0.11039, observed rank = 1000, p-value = 0.001
## alternative hypothesis: greater
```

También se incluye una evaluación de la importancia de la autocorrelación en los recuentos de casos. La tasa global constante r se calcula primero y se utiliza para crear recuentos esperados

para cada tramo censal multiplicando por la población.

```
> r <- sum(NY8$Cases)/sum(NY8$POP8)
> rni <- r * NY8$POP8
> CR <- function(var, mle) rpois(length(var), lambda = mle)
> MoranI.pboot <- function(var, i, listw, n, S0, ...) {
+     return(moran(x = var, listw = listw, n = n, S0 = S0)$I)
+ }
>
> set.seed(1234)
>
> library(boot)
>
> boot2 <- boot(NY8$Cases, statistic = MoranI.pboot, R = 999, sim = "parametric", ran.ger
+     listw = lw_B, n = length(NY8$Cases), S0 = Szero(lw_B), mle = rni)
>
> pnorm((boot2$t0 - mean(boot2$t))/sd(boot2$t[, 1]), lower.tail = FALSE)
```

```
## [1] 0.1472112
```



Histogramas de valores simulados de la I de Moran con permutaciones aleatorias de los datos y simulaciones paramétricas desplazando la distribución de la I de Moran hacia la derecha, porque está tomando en cuenta el impacto de las poblaciones heterogéneas de la zona; el valor observado de la I de Moran está marcado por una línea vertical.

```
> rni <- fitted(glm(Cases ~ 1 + offset(log(POP8)), data = NY8, family = "poisson"))
```

Estos recuentos esperados *rni* se transmiten al argumento lambda a rpois para generar los conjuntos de datos sintéticos mediante el muestreo de la distribución de Poisson. El valor de la probabilidad de salida se calcula a partir del mismo I de Moran observado menos la media de los valores I simulados y se divide por su desviación estándar.

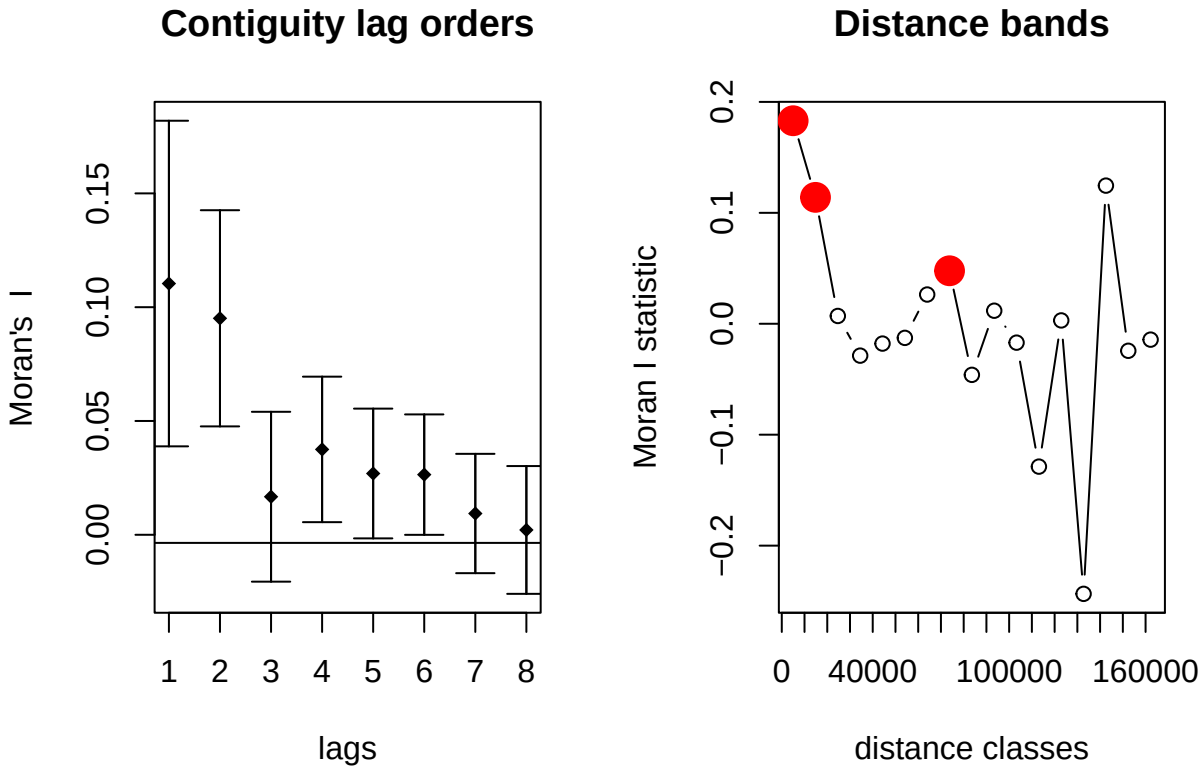
Existe una versión de la I de Moran adaptada para usar una tasa empírica de Bayes que también utiliza métodos de Monte Carlo para la inferencia.

```
> set.seed(1234)
> EBImoran.mc(n = NY8$Cases, x = NY8$POP8, listw = nb2listw(NY_nb, style = "B"), nsim = 9

##
##  Monte-Carlo simulation of Empirical Bayes Index (mean subtracted)
##
## data:  cases: NY8$Cases, risk population: NY8$POP8
## weights: nb2listw(NY_nb, style = "B")
## number of simulations + 1: 1000
##
## statistic = 0.072497, observed rank = 983, p-value = 0.017
## alternative hypothesis: greater
```

Los resultados de las tasas empíricas de Bayes sugieren que una razón de la falta de significación del bootstrapping paramétrico del riesgo constante observado y los valores esperados podría ser que se observaron valores inusuales y extremos en áreas con poblaciones pequeñas. Una vez que se han suavizado las tasas, se encuentra cierta autocorrelación global.

```
> cor8 <- sp.correlogram(neighbours = NY_nb, var = NY8$Cases, order = 8, method = "I",
+     style = "C")
>
> library(pgirmess)
```

```
> corD <- correlog(coordinates(NY8), NY8$Cases, method = "Moran")
>
> oopar <- par(mfrow = c(1, 2))
> plot(cor8, main = "Contiguity lag orders")
> plot(corD, main = "Distance bands")
```




```
> par(oopar)
```

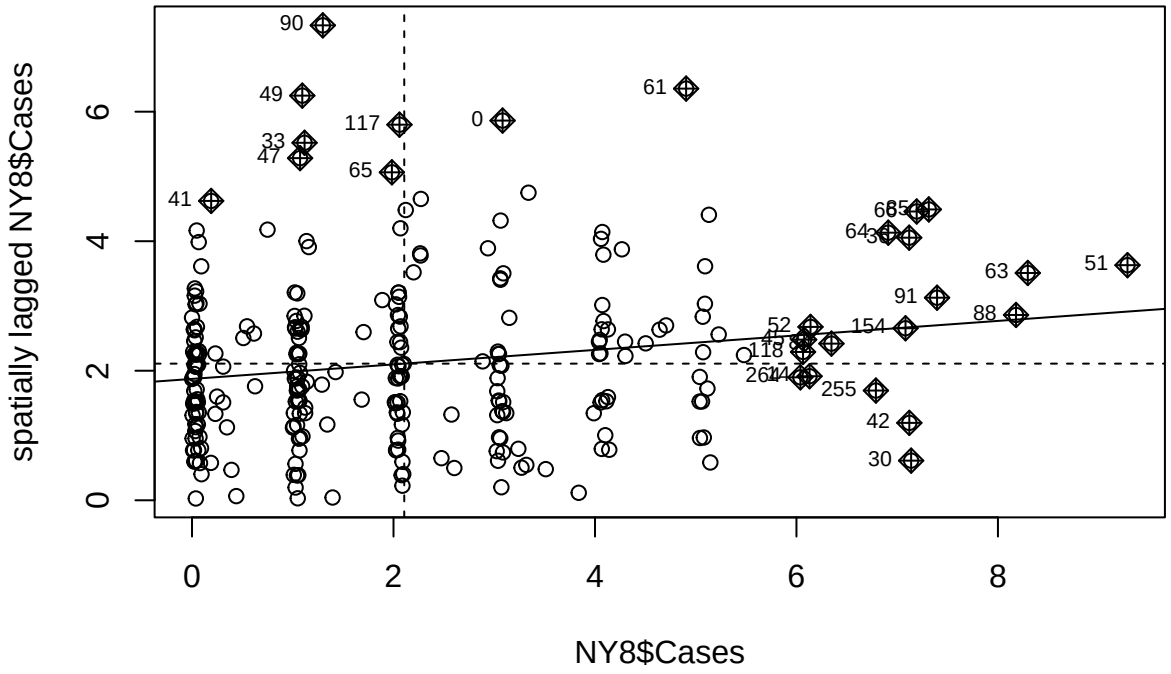
El panel derecho de la figura presenta la salida de la función `correlog` en el paquete `pgirmess` La función selecciona automáticamente bandas de distancia de casi 10 km, que abarcan toda el área de estudio. En este caso, las dos primeras bandas de 0 a 10 y 10 a 20 km tienen valores significativos.

3.2. Pruebas locales

Las pruebas globales de autocorrelación espacial se calculan en base a las relaciones locales entre los valores observados y sus vecinos. Debido a esto se pueden dividir las pruebas globales en sus componentes y construir pruebas localizadas destinadas a detectar agrupamientos "Clusters" o "hotspots" observaciones con vecinos muy diferentes.

Para entrar en detalle se explicará a continuación un diagrama de dispersión de Moran de la variable de recuento de casos de leucemia; para un análisis de los diagramas de dispersión de Moran. En el gráfico se coloca en el eje x la variable de interés y en el y la suma ponderada espacialmente de los vecinos. La I de Moran global es una relación lineal y ve representado como una pendiente; el gráfico se divide en cuatro cuadrantes que representan los valores medios de la variable y sus valores rezagados: bajo-bajo, bajo-alto, alto-bajo y alto-alto.

```
> moran.plot(NY8$Cases, listw = nb2listw(NY_nb, style = "C"))
```



Dado que el I de Moran es global al igual que los coeficientes de correlación son similares (una relación lineal), también se pueden utilizar técnicas estándar para detectar observaciones con una influencia inusualmente fuerte en la pendiente. Específicamente, `moran.plot` llama a `influence.measures` en el modelo lineal de `lm(wx ~ x)` proporcionando el coeficiente de pendiente, donde `wx` es el valor espacialmente rezagado de `x`. Esto significa que se puede ver si las relaciones locales particulares pueden influir en la pendiente más que proporcionalmente. En la siguiente figura se muestra áreas con influencia significativa (usando criterios estándar) codificadas por su cuadrante en el diagrama de dispersión de Moran.

Los valores de I_i de Moran local se construyen como las n componentes sumados para llegar a I

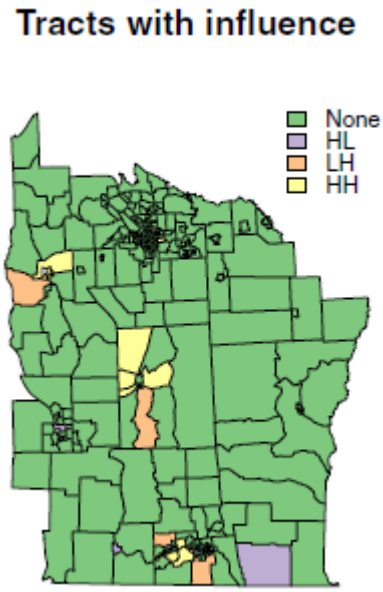


Figura 3: Tratados con influencia del cuadrante de la gráfica de dispersión de Moran

de Moran global:

$$I_i = \frac{(y_i - \overline{y}) \sum_{j=1}^{n} \omega_{ij}(y_j - \overline{y})}{\frac{\sum_{i=1}^{n}(y_i - \overline{y})^2}{n}},$$

donde se sume que la media global $\overline{y}$ es una representación adecuada de la variable de interés y. Los dos componentes del numerador, $(y_i - \overline{y})$ y $\sum_{j=1}^{n} \omega_{ij}(y_j - \overline{y})$ aparecen sin centrarse en el diagrama de dispersión de Moran.

La estadística local puede probarse en busca de divergencia de los valores esperados, bajo supuestos de normalidad y aleatorización analítica, y utilizando aproximaciones de `Sasslepoint` y métodos exactos. Los dos últimos pueden ser importantes porque el número de vecinos de cada observación es muy pequeño y esto, a su vez, puede hacer que la adopción del supuesto de normalidad sea problemático.

Es difícil detectar patrones locales residuales en presencia de autocorrelación espacial global. Por esta razón, los resultados para la dependencia local no deben verse como “absolutos", sino que estan condicionados al menos por la autocorrelación espacial global, es decir por la posible influencia de los procesos de generación de datos espaciales en un rango de escalas desde el global hasta el local. Dependencia no detectada en la escala de las observaciones.

```
> lm1 <- localmoran(NY8$Cases, listw = nb2listw(NY_nb, style = "C"))
>
> lm2 <- as.data.frame(localmoran.sad(lm(Cases ~ 1, NY8), nb = NY_nb, style = "C"))
>
> lm3 <- as.data.frame(localmoran.exact(lm(Cases ~ 1, NY8), nb = NY_nb, style = "C"))
```

Waller y Gotway (2004, p. 239) extienden su tratamiento de hipótesis de riesgo constante al I_i local de Moran, y podemos seguir su ejemplo:

```
> r <- sum(NY8$Cases)/sum(NY8$POP8)
> rni <- r * NY8$POP8
> lw <- nb2listw(NY_nb, style = "C")
> sdCR <- (NY8$Cases - rni)/sqrt(rni)
> wsdCR <- lag(lw, sdCR)
> I_CR <- sdCR * wsdCR
```

La siguiente figura muestra los dos conjuntos de valores de I_i de Moran local, calculados de la forma estándar y utilizando el supuesto de Poisson para la hipótesis de riesgo constante. Ya sabemos que el I global de Moran puede variar en valor e inferencia dependiendo de nuestras suposiciones, por ejemplo, que la inferencia debe tener en cuenta las desviaciones de nuestras suposiciones distributivas. Lo mismo se aplica aquí al supuesto para la distribución de Poisson de que su media y desviación estándar son iguales, mientras que la dispersión excesiva parece ser un problema en los datos que también muestran autocorrelación. Hay algunos cambios de signo entre los mapas, con los valores de la hipótesis de riesgo constante algo más alejados de cero.

También podemos construir una prueba de Monte Carlo simple de la hipótesis de riesgo constante de los valores locales de I_i de Moran, simulando mucho como en el caso global, pero ahora conservando todos los resultados locales. Una vez que se completa la simulación, extraemos el rango del valor I_i de Moran local de riesgo constante observado para cada tramo y calculamos su valor de probabilidad para el número de simulaciones realizadas. Usamos un enfoque paramétrico para simular los recuentos locales utilizando el recuento local esperado como parámetro de `rpois`, porque los recuentos de vecinos son muy bajos y hacen que la permutación sea imprudente. Llevar a cabo pruebas de permutación utilizando todo el conjunto de datos tampoco parece prudente, porque entonces estaríamos comparando iguales con diferentes (figura 5)

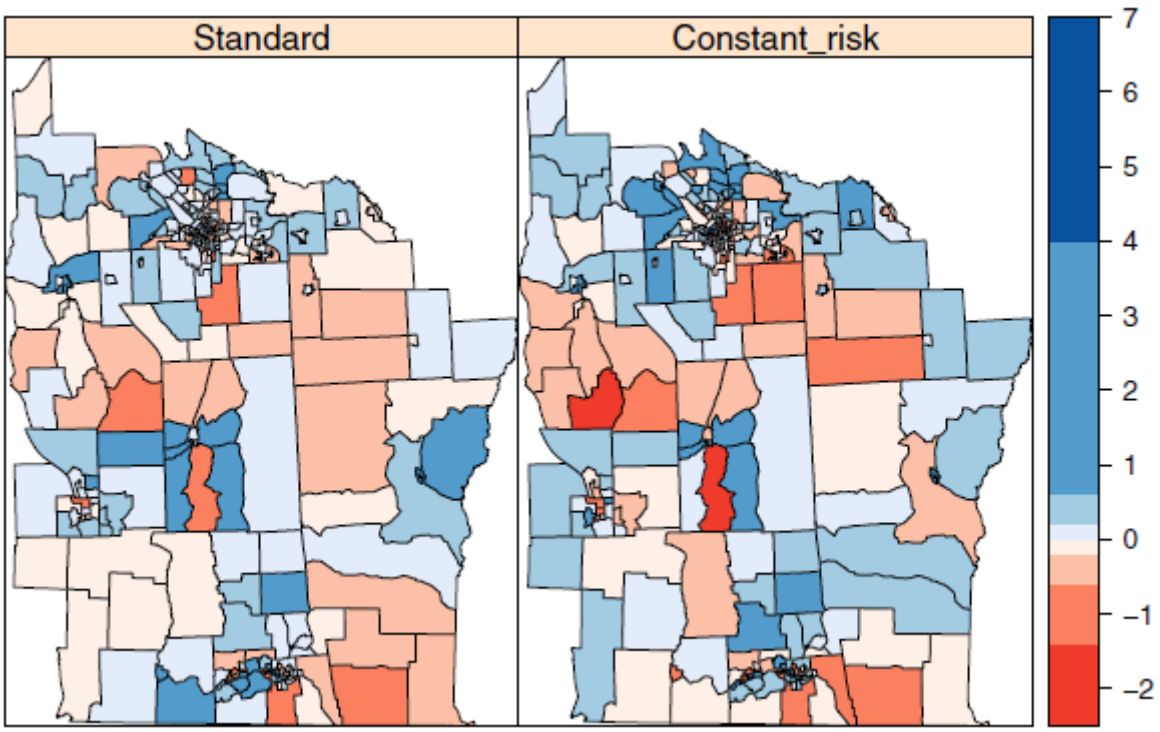
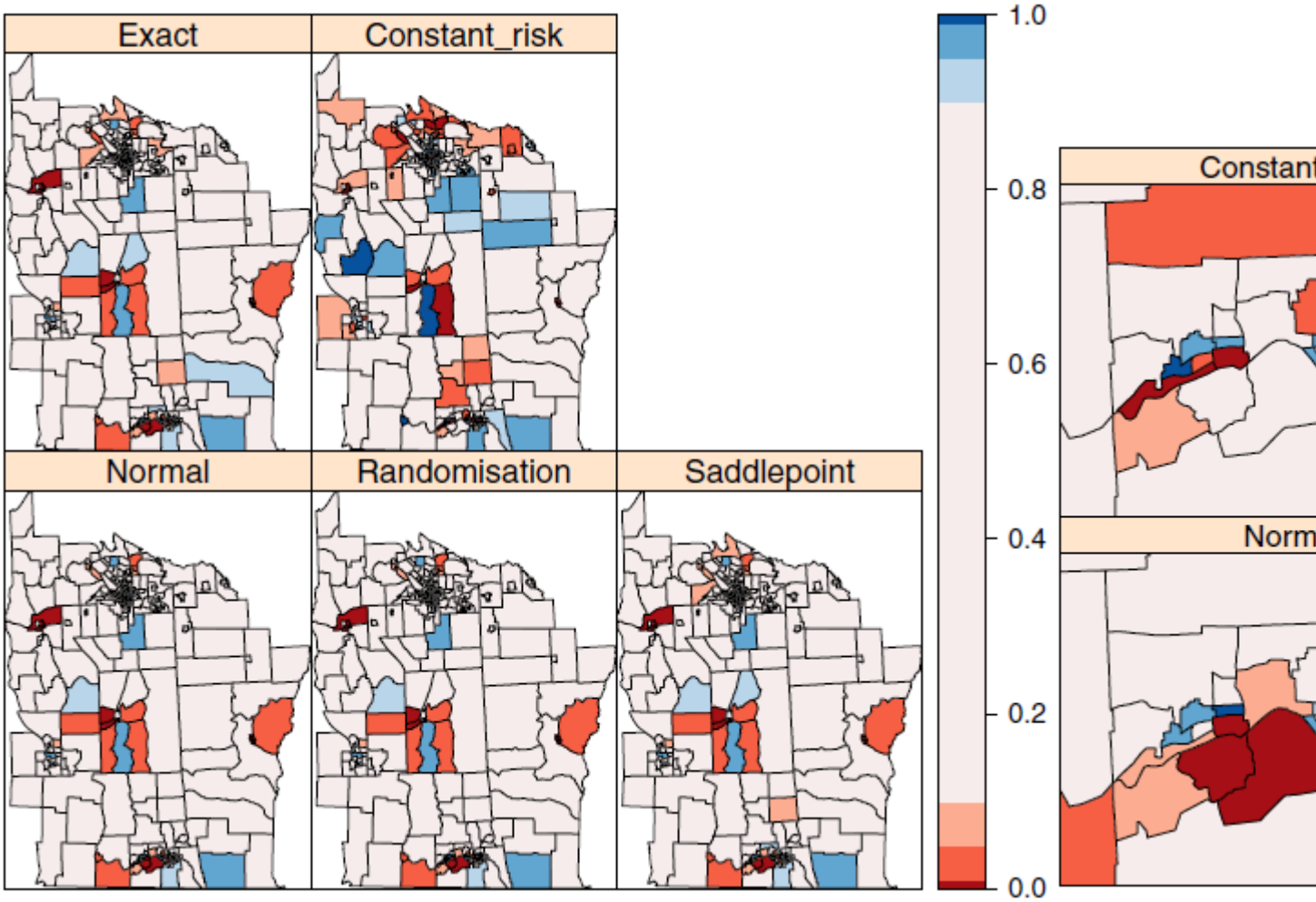


Figura 4: Valores de I_i locales de Moran calculados directamente y utilizando la hipótesis de riesgo constante

```
> set.seed(1234)
> nsim <- 999
> N <- length(rni)
> sims <- matrix(0, ncol = nsim, nrow = N)
>
> for (i in 1:nsim) {
+     y <- rpois(N, lambda = rni)
+     sdCRi <- (y - rni)/sqrt(rni)
+     wsdCRi <- lag(lw, sdCRi)
+     sims[, i] <- sdCRi * wsdCRi
+ }
>
> xrank <- apply(cbind(I_CR, sims), 1, function(x) rank(x)[1])
> diff <- nsim - xrank
> diff <- ifelse(diff > 0, diff, 0)
> pval <- punif((diff + 1)/(nsim + 1))
```

Finalmente, hacemos zoom para examinar los valores de probabilidad I_i de Moran locales para

tres métodos de cálculo para los tramos en y cerca de la ciudad de Binghampton (Fig. 6). Parece que el uso del enfoque de riesgo constante maneja la heterogeneidad en los conteos mejor que las alternativas. Estos resultados concuerdan en general con los alcanzados por Waller y Gotway (2004, p. 241), pero notamos que nuestro modelo subyacente es muy simplista. Encontrar la autocorrelación espacial no es un objetivo en sí mismo, ya sea local o global, sino más bien un paso en un proceso que conduce a un modelo adecuado. Es a esta tarea a la que nos dirigimos ahora.



4. Ajuste de modelos de datos de área

En esta sección, se muestra cómo se puede modelar la estructura espacial en dependencia entre observaciones, en particular para datos de área, pero cuando sea necesario también utilizando representaciones alternativas.

Los problemas a los que nos enfrentamos cuando intentamos ajustar modelos en presencia de autocorrelación espacial son desafiantes, sobre todo porque la autocorrelación espacial que parece

que hemos encontrado puede provenir de la especificación incorrecta del modelo. Si este es el caso, el esfuerzo invertido en modelar la estructura espacial se utilizaría mejor para mejorar el modelo en sí, tal vez manejando la heterocedasticidad, agregando una covariable faltante, revisando la forma funcional de las covariables incluidas o reconsiderando la representación distributiva de la variable de respuesta.

4.1. Enfoques de estadística espacial

En muchos casos, es común suponer que las observaciones son independientes y están distribuidas de manera idéntica, pero este puede no ser el caso cuando se trabaja con datos espaciales. Las observaciones no son independientes porque puede existir alguna correlación entre áreas vecinas. También puede ser difícil distinguir el impacto de la autocorrelación espacial y las diferencias espaciales en la distribución de la observación.

Desde un punto de vista estadístico, es posible dar cuenta de las observaciones correlacionadas considerando una estructura del siguiente tipo en el modelo. Si el vector de variables de respuesta es normal multivariante, podemos expresar el modelo de la siguiente manera:

$$Y = \mu + e$$

donde μ es el vector de medias de área, que se puede modelar de diferentes maneras y e es el vector de errores aleatorios, que asumimos normalmente se distribuye con media cero y varianza genérica V.

Para el caso de variables no normales, podríamos transformar los datos originales para lograr la Normalidad deseada. Por lo tanto, las técnicas que se describen a continuación aún se pueden aplicar a los datos transformados. Nos centramos en dos enfoques que se utilizan habitualmente en la práctica, como los modelos SAR (Autoregresivo simultáneo) y CAR (Autoregresivo condicionalmente).

Anteriormente en este capítulo, tomamos la media de los recuentos de casos de leucemia por zona como nuestro mejor entendimiento del proceso de generación de datos, complementando esto con el enfoque de riesgo constante para tratar de manejar la heterogeneidad proveniente de variaciones en las poblaciones de las zonas. Una de las alternativas examinadas por Waller y Gotway (2004, p. 348) es tomar una transformación logarítmica de la tasa:

$$z_i = log\frac{1000(Y_i + 1)}{n_i}$$

Como covariables, hemos utilizado la distancia inversa al TCE más cercano (PEXPOSURE), la proporción de personas de 65 años o más (PCTAGE65P) y la proporción de personas que poseen su propia casa (PCTOWNHOME). Para preparar el escenario, comencemos con un modelo lineal de la relación entre las proporciones de incidencia transformadas y las covariables.

```
> nylm <- lm(Z ~ PEXPOSURE + PCTAGE65P + PCTOWNHOME, data = NY8)
> summary(nylm)
```

```
##
## Call:
## lm(formula = Z ~ PEXPOSURE + PCTAGE65P + PCTOWNHOME, data = NY8)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.7417 -0.3957 -0.0326  0.3353  4.1398
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.51728    0.15856  -3.262  0.00124 **
## PEXPOSURE    0.04884    0.03506   1.393  0.16480
## PCTAGE65P    3.95089    0.60550   6.525 3.22e-10 ***
## PCTOWNHOME  -0.56004    0.17031  -3.288  0.00114 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6571 on 277 degrees of freedom
## Multiple R-squared:  0.1932, Adjusted R-squared:  0.1844
## F-statistic:  22.1 on 3 and 277 DF,  p-value: 7.306e-13
```

```
> NY8$lmresid <- residuals(nylm)
```

Mas adelante en un gráfico se muestra la distribución espacial de los valores residuales para las

secciones censales del área de estudio. Las dos variables del censo parecen contribuir a explicar la varianza en la variable de respuesta, pero no la exposición al TCE. Además, aunque hay menos autocorrelación espacial en los residuos del modelo con covariables que en el modelo nulo, está claro que hay información en los residuos que deberíamos intentar utilizar. Una prueba exacta de autocorrelación espacial en los residuos conduce a conclusiones similares

Dado que la prueba de Moran está destinada a detectar la autocorrelación espacial, podemos intentar ajustar un modelo teniendo esto en cuenta. Sin embargo, no debemos olvidar que las especificaciones erróneas detectadas por la I de Moran pueden tener una variedad de causas.

```
> NYlistw <- nb2listw(NY_nb, style = "B")
> lm.morantest(nylm, NYlistw)

##
##  Global Moran I for regression residuals
##
## data:
## model: lm(formula = Z ~ PEXPOSURE + PCTAGE65P + PCTOWNHOME, data = NY8)
## weights: NYlistw
##
## Moran I statistic standard deviate = 2.638, p-value = 0.004169
## alternative hypothesis: greater
## sample estimates:
## Observed Moran I      Expectation         Variance
##      0.083090278     -0.009891282      0.001242320
```

Antes de analizar la estimación del modelo con más detalle, vale la pena mencionar el estimador de probabilidad de perfil aproximado. Suponiendo que la variable de interés se ha desvinculado, y que las ponderaciones espaciales están estandarizadas por filas, el estimador de un paso del parámetro de regresión espacial puede producir más información comparativa que el I de Moran. Para la comparación, mostramos la estimación del parámetro de máxima verosimilitud ajustada utilizando los mismos datos:

```
> NYlistwW <- nb2listw(NY_nb, style = "W")
>
> aple(residuals(nylm), listw = NYlistwW)
```

```
## [1] 0.2051536
```

```
> spautolm(Z ~ PEXPOSURE + PCTAGE65P + PCTOWNHOME, data = NY8, listw = NYlistwW)$lambda
```

```
##    lambda
## 0.2169261
```

4.1.1. Modelos autorregresivos simultáneos

4.1.2. Modelos autorregresivos condicionales

La especificación CAR se basa en la distribución condicional de los términos de error espacial. En este caso, se da la distribución de e_i condicionada sobre e_{-i} (el vector de todos los términos de error aleatorio menos e_i mismo). En lugar de todo el vector e_{-i}, solo se utilizan los vecinos del área i, definidos de una manera elegida. Los representamos por $e_{j\sim i}$. Entonces, una forma sencilla de poner la distribución condicional de e_i es: $e_i|e_{j\sim i} \sim N(\sum_{j\sim i} \frac{c_{ij}e_j}{\sum_{j\sim i} c_{ij}}, \frac{\sigma^2_{e_i}}{\sum_{j\sim i} c_{ij}})$ donde c_{ij} son parámetros de dependencia similares a b_{ij}. Sin embargo, especificar las distribuciones condicionales de los términos de error no implica que exista la distribución conjunta. Para tener una distribución adecuada, se deben establecer algunas restricciones en los parámetros del modelo. Para nuestros propósitos de modelado, las pautas anteriores serán suficientes para obtener una especificación CAR adecuada en la mayoría de los casos.

Para ajustar un modelo CAR, podemos usar la función spautolm nuevamente. Esta vez configuramos el argumento family = "CAR" para especificar que estamos ajustando este tipo de modelos.

```
> nycar <- spautolm(Z ~ PEXPOSURE + PCTAGE65P + PCTOWNHOME, data = NY8, family = "CAR",
+     listw = NYlistw)
> summary(nycar)
```

```
##
## Call: spautolm(formula = Z ~ PEXPOSURE + PCTAGE65P + PCTOWNHOME, data = NY8,
##     listw = NYlistw, family = "CAR")
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -1.539732 -0.384311 -0.030646  0.335126  3.808848
```

```
##
## Coefficients:
##              Estimate Std. Error z value  Pr(>|z|)
## (Intercept) -0.648362   0.181129 -3.5796 0.0003442
## PEXPOSURE    0.077899   0.043692  1.7829 0.0745986
## PCTAGE65P    3.703830   0.627185  5.9055 3.516e-09
## PCTOWNHOME  -0.382789   0.195564 -1.9574 0.0503053
##
## Lambda: 0.084123 LR test value: 5.8009 p-value: 0.016018
## Numerical Hessian standard error of lambda: 0.030868
##
## Log likelihood: -275.8283
## ML residual variance (sigma squared): 0.40758, (sigma: 0.63842)
## Number of observations: 281
## Number of parameters estimated: 6
## AIC: 563.66
```

Los coeficientes estimados de las covariables en el modelo son muy similares a los obtenidos con los modelos SAR. Sin embargo, los valores p de dos covariables, la distancia al TCE más cercano y el porcentaje de personas que poseen una vivienda, están ligeramente por encima del umbral de 0,05. La prueba de razón de verosimilitud indica que existe una autocorrelación espacial significativa y el valor estimado de λ es 0.0841.

Si se considera una regresión ponderada, utilizando el tamaño de la población como ponderaciones, el mismo modelo para dar cuenta de la distribución heterogénea de la población elimina por completo la autocorrelación espacial en los datos. Los coeficientes de las covariables no cambian mucho y todas se vuelven significativas. Por tanto, modelar la autocorrelación espacial mediante especificaciones SAR o CAR no modifica los resultados obtenidos.

```
> nycarw <- spautolm(Z ~ PEXPOSURE + PCTAGE65P + PCTOWNHOME, data = NY8, family = "CAR",
+     listw = NYlistw, weights = POP8)
> summary(nycarw)
```

```
##
## Call: spautolm(formula = Z ~ PEXPOSURE + PCTAGE65P + PCTOWNHOME, data = NY8,
```

```
##     listw = NYlistw, weights = POP8, family = "CAR")
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -1.491042 -0.270906  0.081435  0.451556  4.198134
##
## Coefficients:
##              Estimate Std. Error z value  Pr(>|z|)
## (Intercept) -0.790154   0.144862 -5.4545 4.910e-08
## PEXPOSURE    0.081922   0.028593  2.8651  0.004169
## PCTAGE65P    3.825858   0.577720  6.6223 3.536e-11
## PCTOWNHOME  -0.386820   0.157436 -2.4570  0.014010
##
## Lambda: 0.022419 LR test value: 0.38785 p-value: 0.53343
## Numerical Hessian standard error of lambda: 0.038543
##
## Log likelihood: -251.5711
## ML residual variance (sigma squared): 1102.9, (sigma: 33.21)
## Number of observations: 281
## Number of parameters estimated: 6
## AIC: 515.14
```

4.1.3. Ajuste de modelos de regresión espacial