Ejemplificar_Matrices-de-Pesos_Moran_Geary_Hotspots.R

WILL

2025-10-15

```
# ============================================================================
# ESTADÍSTICA ESPACIAL EN R
# Ejemplos: Matrices de Pesos, Moran, Geary y Hotspots
# ============================================================================

# Instalar y cargar paquetes necesarios
# install.packages(c("spdep", "sf", "tmap", "RColorBrewer"))

library(spdep)      # Análisis espacial
```

```
## Cargando paquete requerido: spData
```

```
## To access larger datasets in this package, install the spDataLarge
## package with: `install.packages('spDataLarge',
## repos='https://nowosad.github.io/drat/', type='source')`
```

```
## Cargando paquete requerido: sf
```

```
## Linking to GEOS 3.13.1, GDAL 3.11.0, PROJ 9.6.0; sf_use_s2() is TRUE
```

```
library(sf)         # Manejo de datos espaciales
library(tmap)       # Visualización de mapas
library(RColorBrewer) # Paletas de colores

# ============================================================================
# [1] MATRICES DE PESOS ESPACIALES (Spatial Weight Matrix)
# ============================================================================

cat("\n=== [1] MATRICES DE PESOS ESPACIALES ===\n")
```

```
##
## === [1] MATRICES DE PESOS ESPACIALES ===
```

```
# Crear datos espaciales de ejemplo: 9 puntos en una cuadrícula 3x3
set.seed(123)
coords <- expand.grid(x = 1:3, y = 1:3)
datos <- data.frame(
  id = 1:9,
  x = coords$x,
  y = coords$y,
  valor = c(10, 15, 12, 20, 25, 18, 11, 16, 14)
)

# Convertir a objeto espacial sf
datos_sf <- st_as_sf(datos, coords = c("x", "y"))

# --- Método 1: Matriz de Contigüidad (Queen) ---
cat("\n1.1. Matriz de Contigüidad Queen (compartir vértice o lado):\n")
```

```
##
## 1.1. Matriz de Contigüidad Queen (compartir vértice o lado):
```

```
# Crear vecindarios tipo Queen
nb_queen <- dnearneigh(st_coordinates(datos_sf), 0, 1.5)
cat("Vecinos de la unidad 5 (centro):", nb_queen[[5]], "\n")
```

```
## Vecinos de la unidad 5 (centro): 1 2 3 4 6 7 8 9
```

```
# Convertir a matriz de pesos
W_queen <- nb2listw(nb_queen, style = "W")  # style="W": pesos estandarizados por fila
print("Matriz de pesos Queen (primeras 5 unidades):")
```

```
## [1] "Matriz de pesos Queen (primeras 5 unidades):"
```

```
print(listw2mat(W_queen)[1:5, 1:5])
```

```
##       1         2     3         4         5
## 1 0.000 0.3333333 0.000 0.3333333 0.3333333
## 2 0.200 0.0000000 0.200 0.2000000 0.2000000
## 3 0.000 0.3333333 0.000 0.0000000 0.3333333
## 4 0.200 0.2000000 0.000 0.0000000 0.2000000
## 5 0.125 0.1250000 0.125 0.1250000 0.0000000
```

```
# --- Método 2: Matriz de Contigüidad (Rook) ---
cat("\n1.2. Matriz de Contigüidad Rook (compartir solo lado):\n")
```

```
##
## 1.2. Matriz de Contigüidad Rook (compartir solo lado):
```

```
nb_rook <- dnearneigh(st_coordinates(datos_sf), 0, 1.1)
W_rook <- nb2listw(nb_rook, style = "W")
cat("Vecinos de la unidad 5 (centro):", nb_rook[[5]], "\n")
```

```
## Vecinos de la unidad 5 (centro): 2 4 6 8
```

```
# --- Método 3: K-vecinos más cercanos ---
cat("\n1.3. Matriz K-vecinos más cercanos (k=4):\n")
```

```
##
## 1.3. Matriz K-vecinos más cercanos (k=4):
```

```
nb_knn <- knn2nb(knearneigh(st_coordinates(datos_sf), k = 4))
```

```
## Warning in knearneigh(st_coordinates(datos_sf), k = 4): k greater than
## one-third of the number of data points
```

```
W_knn <- nb2listw(nb_knn, style = "W")
cat("Vecinos de la unidad 1:", nb_knn[[1]], "\n")
```

```
## Vecinos de la unidad 1: 2 3 4 5
```

```
# --- Método 4: Matriz de Distancia Inversa ---
cat("\n1.4. Matriz basada en distancia inversa:\n")
```

```
##
## 1.4. Matriz basada en distancia inversa:
```

```
dists <- dnearneigh(st_coordinates(datos_sf), 0, 3)
W_dist <- nb2listw(dists, glist = lapply(nbdists(dists, st_coordinates(datos_sf)),
                                         function(x) 1/x), style = "W")
cat("Pesos calculados con distancia inversa\n")
```

```
## Pesos calculados con distancia inversa
```

```
# ============================================================================
# [2] ÍNDICE DE MORAN (Autocorrelación Espacial Global)
# ============================================================================

cat("\n\n=== [2] ÍNDICE DE MORAN ===\n")
```

```
##
##
## === [2] ÍNDICE DE MORAN ===
```

```
# Calcular I de Moran
moran_test <- moran.test(datos$valor, W_queen)

cat("\nResultados del test de Moran:\n")
```

```
##
## Resultados del test de Moran:
```

```
cat("I de Moran:", round(moran_test$estimate[1], 4), "\n")
```

```
## I de Moran: -0.4232
```

```
cat("Valor esperado:", round(moran_test$estimate[2], 4), "\n")
```

```
## Valor esperado: -0.125
```

```
cat("Varianza:", round(moran_test$estimate[3], 6), "\n")
```

```
## Varianza: 0.020059
```

```
cat("Estadístico Z:", round(moran_test$statistic, 4), "\n")
```

```
## Estadístico Z: -2.1055
```

```
cat("p-valor:", round(moran_test$p.value, 4), "\n")
```

```
## p-valor: 0.9824
```

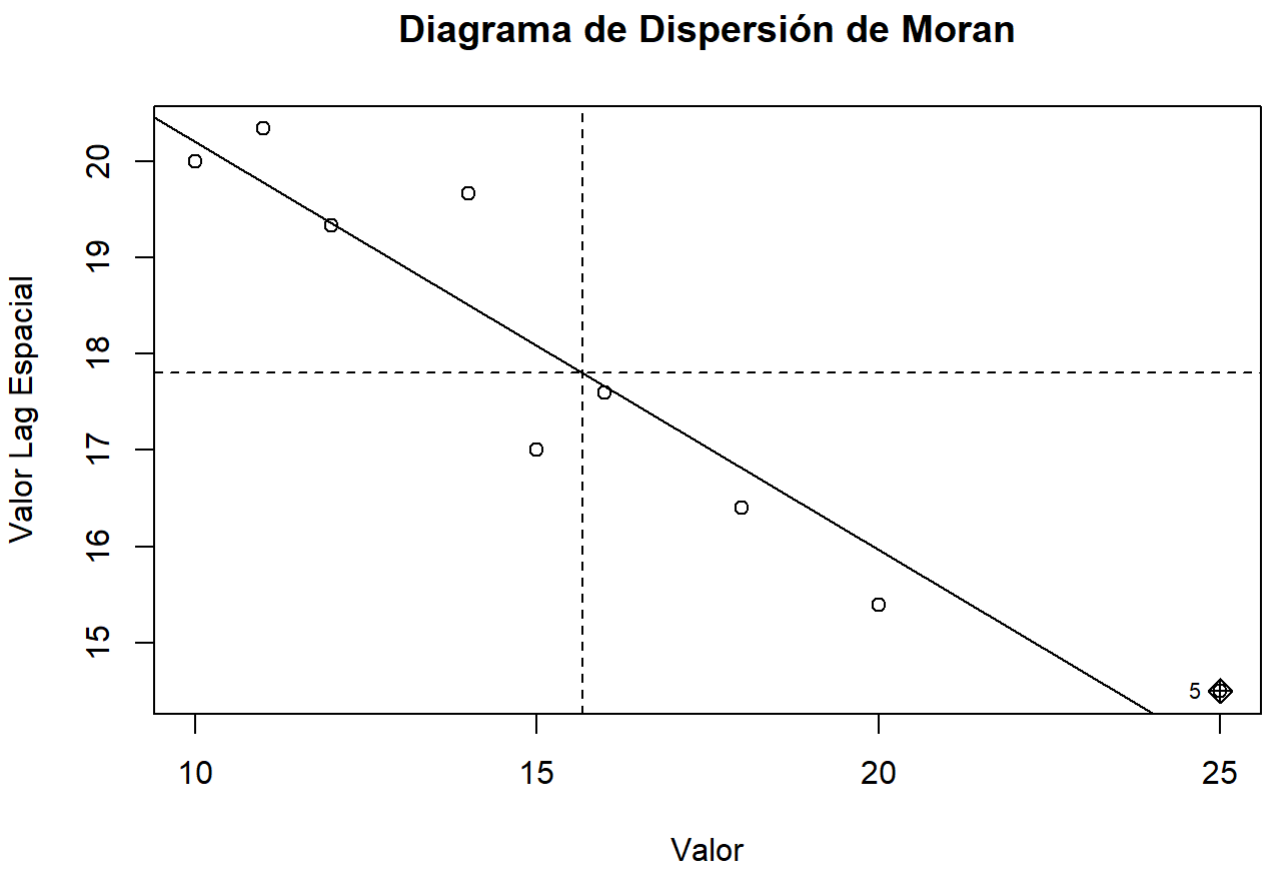
```
if(moran_test$p.value < 0.05) {
  if(moran_test$estimate[1] > 0) {
    cat("Interpretación: Existe autocorrelación espacial POSITIVA significativa\n")
    cat("(valores similares tienden a agruparse espacialmente)\n")
  } else {
    cat("Interpretación: Existe autocorrelación espacial NEGATIVA significativa\n")
    cat("(valores diferentes tienden a estar cerca)\n")
  }
} else {
  cat("Interpretación: NO hay evidencia de autocorrelación espacial\n")
}
```

```
## Interpretación: NO hay evidencia de autocorrelación espacial
```

```
# Gráfico de Moran
cat("\nCreando diagrama de dispersión de Moran...\n")
```

```
##
## Creando diagrama de dispersión de Moran...
```

```
moran.plot(datos$valor, W_queen, labels = datos$id,
           main = "Diagrama de Dispersión de Moran",
           xlab = "Valor", ylab = "Valor Lag Espacial")
```




```
# ============================================================================
# [3] ÍNDICE C DE GEARY (Autocorrelación Espacial Global)
# ============================================================================

cat("\n\n=== [3] ÍNDICE C DE GEARY ===\n")
```

```
##
##
## === [3] ÍNDICE C DE GEARY ===
```

```
# Calcular C de Geary
geary_test <- geary.test(datos$valor, W_queen)

cat("\nResultados del test de Geary:\n")
```

```
##
## Resultados del test de Geary:
```

```
cat("C de Geary:", round(geary_test$estimate[1], 4), "\n")
```

```
## C de Geary: 1.4364
```

```
cat("Valor esperado:", round(geary_test$estimate[2], 4), "\n")
```

```
## Valor esperado: 1
```

```
cat("Varianza:", round(geary_test$estimate[3], 6), "\n")
```

```
## Varianza: 0.032209
```

```
cat("Estadístico Z:", round(geary_test$statistic, 4), "\n")
```

```
## Estadístico Z: -2.4318
```

```
cat("p-valor:", round(geary_test$p.value, 4), "\n")
```

```
## p-valor: 0.9925
```

```
if(geary_test$p.value < 0.05) {
  if(geary_test$estimate[1] < 1) {
    cat("Interpretación: Existe autocorrelación espacial POSITIVA significativa\n")
    cat("(C < 1: valores similares están agrupados)\n")
  } else if(geary_test$estimate[1] > 1) {
    cat("Interpretación: Existe autocorrelación espacial NEGATIVA significativa\n")
    cat("(C > 1: valores diferentes están agrupados)\n")
  }
} else {
  cat("Interpretación: NO hay evidencia de autocorrelación espacial\n")
}
```

```
## Interpretación: NO hay evidencia de autocorrelación espacial
```

```
cat("\nComparación Moran vs Geary:\n")
```

```
##
## Comparación Moran vs Geary:
```

```
cat("- Moran I =", round(moran_test$estimate[1], 4),
    "(positivo indica agrupamiento de valores similares)\n")
```

```
## - Moran I = -0.4232 (positivo indica agrupamiento de valores similares)
```

```
cat("- Geary C =", round(geary_test$estimate[1], 4),
    "(< 1 indica agrupamiento de valores similares)\n")
```

```
## - Geary C = 1.4364 (< 1 indica agrupamiento de valores similares)
```

```
# ============================================================================
# [4] HOTSPOTS (Análisis LISA - Local Indicators of Spatial Association)
# ============================================================================

cat("\n\n=== [4] ANÁLISIS DE HOTSPOTS (LISA) ===\n")
```

```
##
##
## === [4] ANÁLISIS DE HOTSPOTS (LISA) ===
```

```
# Calcular Moran Local (LISA)
local_moran <- localmoran(datos$valor, W_queen)

# Añadir resultados al dataframe
datos$Ii <- local_moran[, "Ii"]  # I de Moran local
datos$pvalor <- local_moran[, "Pr(z != E(Ii))"]  # p-valor
datos$zscore <- local_moran[, "Z.Ii"]  # Z-score

# Clasificar en categorías LISA
datos$lag_valor <- lag.listw(W_queen, datos$valor)
datos$valor_std <- scale(datos$valor)
datos$lag_std <- scale(datos$lag_valor)

# Crear categorías de hotspots
datos$hotspot <- "No significativo"
datos$hotspot[datos$pvalor < 0.05 & datos$valor_std > 0 & datos$lag_std > 0] <- "Alto-Alto (H
otspot)"
datos$hotspot[datos$pvalor < 0.05 & datos$valor_std < 0 & datos$lag_std < 0] <- "Bajo-Bajo (C
oldspot)"
datos$hotspot[datos$pvalor < 0.05 & datos$valor_std < 0 & datos$lag_std > 0] <- "Bajo-Alto (O
utlier)"
datos$hotspot[datos$pvalor < 0.05 & datos$valor_std > 0 & datos$lag_std < 0] <- "Alto-Bajo (O
utlier)"

# Mostrar resultados
cat("\nResultados del análisis LISA:\n")
```

```
##
## Resultados del análisis LISA:
```

```
print(datos[, c("id", "valor", "Ii", "zscore", "pvalor", "hotspot")])
```

```
##   id valor          Ii     zscore     pvalor             hotspot
## 1  1    10 -1.21428571 -1.7397515 0.08190267    No significativo
## 2  2    15 -0.04395604 -0.8963757 0.37005213    No significativo
## 3  3    12 -0.66483516 -1.4396386 0.14996967    No significativo
## 4  4    20 -0.05714286  0.2094628 0.83408701    No significativo
## 5  5    25 -0.53846154        Inf 0.00000000 Alto-Bajo (Outlier)
## 6  6    18  0.08461538  0.7466898 0.45525085    No significativo
## 7  7    11 -1.07692308 -1.8860109 0.05929349    No significativo
## 8  8    16  0.03186813  1.4148128 0.15712339    No significativo
## 9  9    14 -0.32967033 -1.6433307 0.10031453    No significativo
```

```
cat("\n\nResumen de Hotspots:\n")
```

```
##
##
## Resumen de Hotspots:
```

```
table(datos$hotspot)
```

```
##
## Alto-Bajo (Outlier)    No significativo
##                   1                   8
```

```
# Interpretación
cat("\n\nInterpretación de categorías:\n")
```

```
##
##
## Interpretación de categorías:
```

```
cat("- Alto-Alto (Hotspot): Valores altos rodeados de valores altos\n")
```

```
## - Alto-Alto (Hotspot): Valores altos rodeados de valores altos
```

```
cat("- Bajo-Bajo (Coldspot): Valores bajos rodeados de valores bajos\n")
```

```
## - Bajo-Bajo (Coldspot): Valores bajos rodeados de valores bajos
```

```
cat("- Alto-Bajo (Outlier): Valor alto rodeado de valores bajos\n")
```

```
## - Alto-Bajo (Outlier): Valor alto rodeado de valores bajos
```

```
cat("- Bajo-Alto (Outlier): Valor bajo rodeado de valores altos\n")
```

```
## - Bajo-Alto (Outlier): Valor bajo rodeado de valores altos
```

```
cat("- No significativo: Sin patrón espacial significativo\n")
```

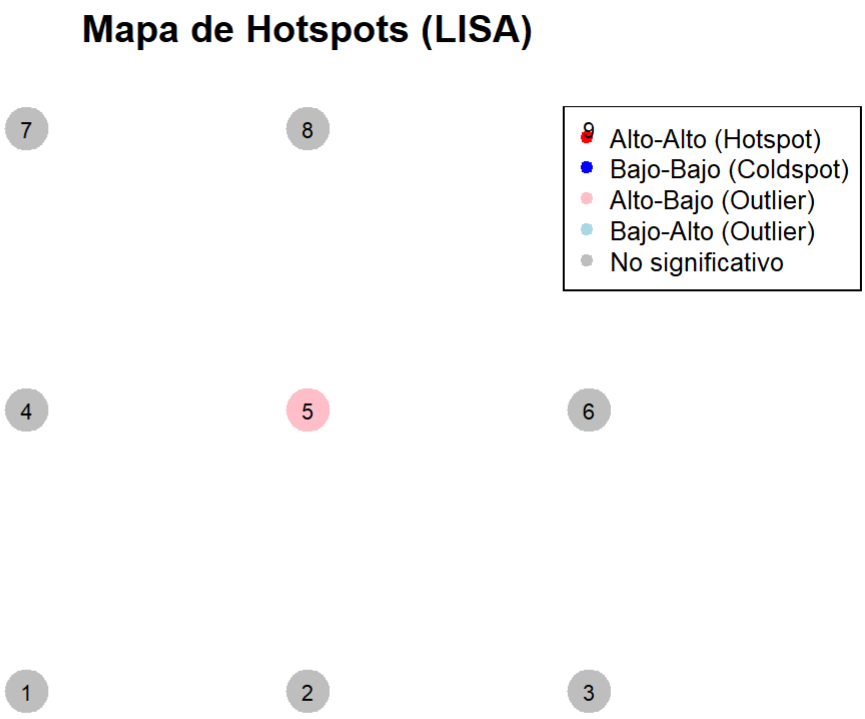
```
## - No significativo: Sin patrón espacial significativo
```

```
# Visualización de Hotspots
cat("\nCreando mapa de hotspots...\n")
```

```
##
## Creando mapa de hotspots...
```

```
# Actualizar objeto sf con categorías
datos_sf$hotspot <- factor(datos$hotspot,
                           levels = c("Alto-Alto (Hotspot)",
                                      "Bajo-Bajo (Coldspot)",
                                      "Alto-Bajo (Outlier)",
                                      "Bajo-Alto (Outlier)",
                                      "No significativo"))

# Crear mapa
colores <- c("red", "blue", "pink", "lightblue", "gray")
plot(st_geometry(datos_sf), col = colores[datos_sf$hotspot],
     pch = 19, cex = 3, main = "Mapa de Hotspots (LISA)")
legend("topright", legend = levels(datos_sf$hotspot),
       col = colores, pch = 19, cex = 0.8)
text(st_coordinates(datos_sf), labels = datos$id, cex = 0.7)
```



```
# ============================================================================
# RESUMEN FINAL
# ============================================================================

cat("\n\n=== RESUMEN DEL ANÁLISIS ===\n")
```

```
##
##
## === RESUMEN DEL ANÁLISIS ===
```

```
cat("1. Matrices de Pesos: Creadas con métodos Queen, Rook, KNN y Distancia\n")
```

```
## 1. Matrices de Pesos: Creadas con métodos Queen, Rook, KNN y Distancia
```

```
cat("2. I de Moran:", round(moran_test$estimate[1], 4), "- p-valor:",
    round(moran_test$p.value, 4), "\n")
```

```
## 2. I de Moran: -0.4232 - p-valor: 0.9824
```

```
cat("3. C de Geary:", round(geary_test$estimate[1], 4), "- p-valor:",
    round(geary_test$p.value, 4), "\n")
```

```
## 3. C de Geary: 1.4364 - p-valor: 0.9925
```

```
cat("4. Hotspots identificados:", sum(datos$hotspot != "No significativo"),
    "de", nrow(datos), "unidades\n")
```

```
## 4. Hotspots identificados: 1 de 9 unidades
```