Lab 5: Spatial Statistics

Norah Saarman

2024-10-17

Section 5.4 Exercise

https://bookdown.org/hhwagner1/LandGenCourse_book/r-exercise-week-2.html

a. Load libraries

```
library(LandGenCourse)
#library(EcoGenetics)
library(GeNetIt)
```

```
## Loading required package: nlme
```

```
library(hierfstat)
library(adegenet)
```

```
## Loading required package: ade4
```

```
##
##    /// adegenet 2.1.10 is loaded ////////////
##
##    > overview: '?adegenet'
##    > tutorials/doc/questions: 'adegenetWeb()'
##    > bug reports/feature requests: adegenetIssues()
```

```
##
## Attaching package: 'adegenet'
```

```
## The following objects are masked from 'package:hierfstat':
##
##     Hs, read.fstat
```

```
require(gstudio)
```

```
## Loading required package: gstudio
```

```
## Warning: replacing previous import 'dplyr::union' by 'raster::union' when
## loading 'gstudio'
```

```
## Warning: replacing previous import 'dplyr::intersect' by 'raster::intersect'
## when loading 'gstudio'
```

```
## Warning: replacing previous import 'dplyr::select' by 'raster::select' when
## loading 'gstudio'
```

```
##
## Attaching package: 'gstudio'
## The following objects are masked from 'package:adegenet':
##
##     alleles, ploidy

## The following object is masked from 'package:hierfstat':
##
##     Ho
```

```r
require(dplyr)
```

```
## Loading required package: dplyr

##
## Attaching package: 'dplyr'

## The following object is masked from 'package:nlme':
##
##     collapse

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
require(tibble)
```

```
## Loading required package: tibble
```

```r
require(sf)
```

```
## Loading required package: sf

## Linking to GEOS 3.10.2, GDAL 3.4.1, PROJ 8.2.1; sf_use_s2() is TRUE
```

```r
require(popgraph)
```

```
## Loading required package: popgraph
```

```r
require(RgoogleMaps)
```

```
## Loading required package: RgoogleMaps

##
## Thank you for using RgoogleMaps!

##
## To acknowledge our work, please cite the package:

##  Markus Loecher and Karl Ropkins (2015). RgoogleMaps and loa: Unleashing R
##   Graphics Power on Map Tiles. Journal of Statistical Software 63(4), 1-18.
```

```r
require(geosphere)
```

```
## Loading required package: geosphere
```

```r
require(proto)
```

```
## Loading required package: proto
```

```r
require(sampling)
```

```
## Loading required package: sampling

##
## Attaching package: 'sampling'

## The following object is masked from 'package:adegenet':
##
##     strata
```

```r
require(seqinr)
```

```
## Loading required package: seqinr

##
## Attaching package: 'seqinr'

## The following object is masked from 'package:dplyr':
##
##     count

## The following object is masked from 'package:nlme':
##
##     gls
```

```r
require(spacetime)
```

```
## Loading required package: spacetime
```

```r
require(spdep)
```

```
## Loading required package: spdep

## Loading required package: spData

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

## Registered S3 method overwritten by 'spdep':
##   method   from
##   plot.mst ape

##
## Attaching package: 'spdep'

## The following object is masked from 'package:ade4':
##
##     mstree
```

```r
require(here)
```

```
## Loading required package: here

## here() starts at /uufs/chpc.utah.edu/common/home/u6036559/git/usu-biol4750
```

b. Import to adegenet object G-Studio, Drop offspring:

```r
# 1. CSV file "./downloads/pulsatilla_genotypes.csv" --> data frame
# with "gstudio" function read_population()
Flr <- read_population("./downloads/pulsatilla_genotypes.csv",
                       type = "column",locus.columns = c(6:19))
```

```r
# 2. Select only adults with base R indexing of data frame
# rows where OffID==0, all columns
Flr <- Flr[Flr$OffID==0,]

# 3. Nothing to do here

# 4. Create genind object with "adegenet" function df2genind()
# using NA.char = ""
Flr.genind <- df2genind(X=Flr[,c(6:12)], sep=":", ncode=NULL, ind.names=Flr$ID, loc.names=NULL, pop=Flr

# 5. Check genind object
Flr.genind
```

```
## /// GENIND OBJECT /////////
##
##  // 221 individuals; 7 loci; 105 alleles; size: 129.8 Kb
##
##  // Basic content
##    @tab:  221 x 105 matrix of allele counts
##    @loc.n.all: number of alleles per locus (range: 8-25)
##    @loc.fac: locus factor for the 105 columns of @tab
##    @all.names: list of allele names for each locus
##    @ploidy: ploidy of each individual  (range: 2-2)
##    @type:  codom
##    @call: df2genind(X = Flr[, c(6:12)], sep = ":", ncode = NULL, ind.names = Flr$ID,
##     loc.names = NULL, pop = Flr$Population, NA.char = "", ploidy = 2,
##     type = "codom", strata = NULL, hierarchy = NULL)
##
##  // Optional content
##    @pop: population of each individual (group size range: 14-56)
```

```r
summary(Flr.genind)
```

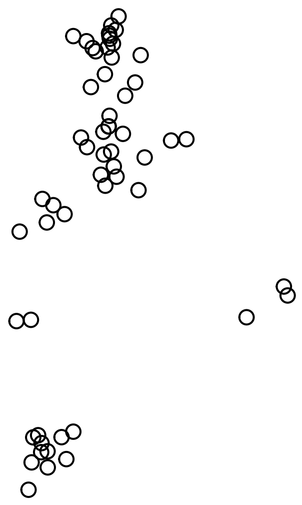
```
##
## // Number of individuals: 221
## // Group sizes: 21 56 21 22 14 42 45
## // Number of alleles per locus: 18 8 25 8 19 14 13
## // Number of alleles per group: 63 68 54 50 51 73 53
## // Percentage of missing data: 0.9 %
## // Observed heterozygosity: 0.74 0.54 0.89 0.71 0.74 0.68 0.74
## // Expected heterozygosity: 0.83 0.57 0.89 0.74 0.81 0.76 0.83
```

c. Plot locations of individuals from site A25 From data frame

```r
# Select data for Population "A25"
Sites <- Flr[Flr$Population == "A25", c("X", "Y")]

# Check if the data frame has valid coordinates
head(Sites)
```

```
##          X       Y
## 16 4422658 5425371
## 17 4422659 5425372
## 18 4422659 5425371
## 19 4422658 5425370
## 20 4422659 5425371
## 21 4422659 5425371
```

```
# Convert to an 'sf' object
Sites_sf <- st_as_sf(Sites, coords = c("X", "Y"), crs = 31468)
# Transform to WGS84 (lat/long, EPSG:4326)
Sites_latlon <- st_transform(Sites_sf, crs = 4326)

plot(Sites_latlon)
```



d. Add geographic info to genind object

Genind object @other can hold a list such as spatial coordinates

```
# Select xy coordinates from Flr
Sites <- Flr[, c("X", "Y")]

# Convert to an 'sf' object with the correct original CRS (replace 31468 if needed)
Sites_sf <- st_as_sf(Sites, coords = c("X", "Y"), crs = 31468)

# Transform to WGS84 (lat/lon, EPSG:4326)
Sites_latlon <- st_transform(Sites_sf, crs = 4326)

# Extract coordinates from the transformed sf object
latlon_matrix <- st_coordinates(Sites_latlon)

# Convert to a matrix with column names "longitude" and "latitude"
colnames(latlon_matrix) <- c("longitude", "latitude")

# Add the matrix to the @other slot with the name "xy"
Flr.genind@other <- list(xy = latlon_matrix)
```

e. Calculate genetic and geographic euclidean distance (for Site A25)

I can't tell if we are meant to proceed with just population A25? If so, I would first subset like this:

```
# Subset only site A25
Flr.A25 <- Flr.genind[Flr.genind@pop == "A25"]
```

For now I'll proceed with the full genind object.

Step 1: Genetic distance with adegenet::propShared:

```r
# Calculate genetic distance matrix
Dgen_matrix <- adegenet::propShared(Flr.A25)

# Convert to a distance vector (1 - proportion shared)
Dgen <- as.dist(1 - Dgen_matrix)
Dgen.vec <- as.vector(Dgen)
```

Step 2: Geographic with dist()

One option is to use the @other slot for lat long coordinates

```r
# Extract the 'xy' matrix from the @other slot
xy_matrix <- Flr.A25@other$xy

# Convert the matrix back into a data frame
df_xy <- as.data.frame(xy_matrix)
```

Other (simpler) option is to pull from the original data frame, this is in the grid coordinate system

```r
df_xy <- Flr[Flr$Population == "A25", c("X","Y")]
```

```r
colnames(df_xy) <- c("X", "Y")

# Calculate Euclidean geographic distances
Dgeo <- dist(df_xy)

Dgeo.vec <- as.vector(Dgeo)
```

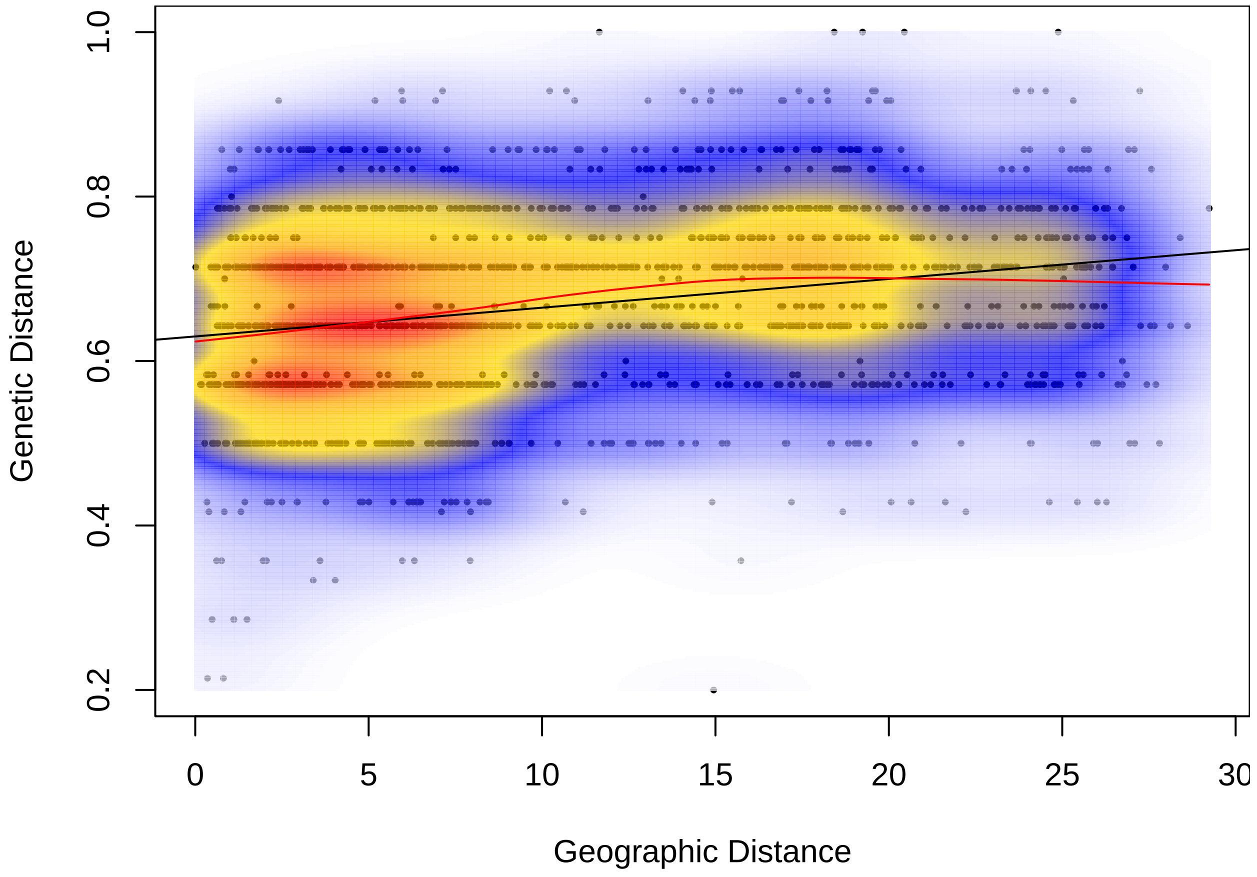
Adapt section 4a to Visualize

```r
# Load necessary packages
library(MASS)  # For kde2d()
```

```
##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select
```

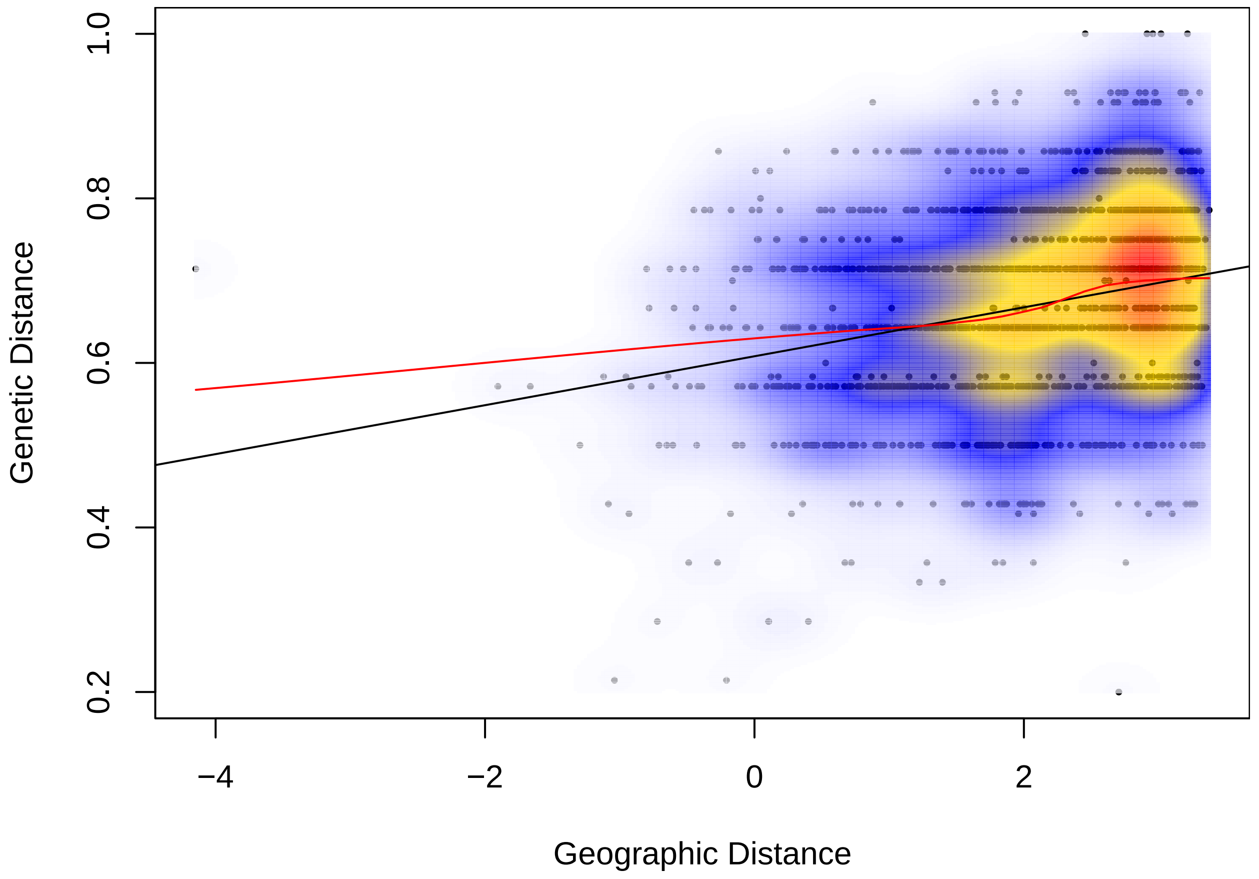
```r
library(scales)  # For transparent colors using alpha

# Step 5: Check visual linearity
par(mar = c(4, 4, 0, 0))  # Adjust plot margins

# Generate density for better visualization
dens <- kde2d(Dgeo.vec, Dgen.vec, n = 300)

# CrDgeo# Create a color palette
myPal <- colorRampPalette(c("white", "blue", "gold", "orange", "red"))

# Plot geographic vs genetic distance
```

```r
plot(Dgeo.vec, Dgen.vec, pch = 20, cex = 0.5,
     xlab = "Geographic Distance", ylab = "Genetic Distance")

# Add density image with transparency
image(dens, col = alpha(myPal(300), 0.7), add = TRUE)

# Add linear regression line
abline(lm(Dgen.vec ~ Dgeo.vec), col = "black")

# Add loess smoothing line in red
lines(loess.smooth(Dgeo.vec, Dgen.vec), col = "red")
```



With log transformation

```r
#Lets take the natural logarithm of #geographic distance:

par(mar=c(4,4,0,0))
dens <- MASS::kde2d(log(Dgeo.vec), Dgen.vec, n=300)
plot(log(Dgeo.vec), Dgen.vec, pch=20, cex=0.5,
    xlab="Geographic Distance", ylab="Genetic Distance")
image(dens, col=transp(myPal(300), 0.7), add=TRUE)
abline(lm(Dgen.vec ~ log(Dgeo.vec)))
lines(loess.smooth(log(Dgeo.vec), Dgen.vec), col="red")
```



Do you notice something unusual in the plot? Why are there so few different values of genetic distance?

We have 56 individuals, and 7 loci, 105 alleles… Maybe low heterozygosity? No. Maybe closely related or even clonal individuals because of vegetative reproduction?

Do you think there is spatial autocorrelation? If so, up to what distance? Yes, it looks like there is correlation of Dgeo and Dgen up until ~ 15 geographic units.

f. Mantel test

```r
IBD <- vegan::mantel(Dgen,Dgeo, method="pearson")
IBD
```

```
##
## Mantel statistic based on Pearson's product-moment correlation
##
## Call:
## vegan::mantel(xdis = Dgen, ydis = Dgeo, method = "pearson")
##
## Mantel statistic r: 0.2333
##       Significance: 0.001
##
## Upper quantiles of permutations (null model):
##    90%    95%  97.5%    99%
## 0.0567 0.0742 0.0847 0.1054
## Permutation: free
## Number of permutations: 999
```

g. Mantel Correlogram Without EcoGenetics

Step 3: Define Distance Classes (Binning Geographic Distances)

To create a Mantel correlogram, we need to divide the geographic distances into bins or classes.

```r
# Step 3: Create geographic distance bins (classes)
# Define the number of bins (adjust this as necessary)
num_bins <- 10

# Get the range of geographic distances
max_dist <- max(Dgeo.vec, na.rm = TRUE)

# Define the breaks for binning distances
breaks <- seq(0, max_dist, length.out = num_bins + 1)

# Assign each pairwise distance to a bin (using cut)
Dgeo_bins <- cut(as.vector(Dgeo), breaks = breaks, include.lowest = TRUE)
```

Step 4. Perform Mantel Tests for Each Distance Class

In each bin, we perform a Mantel test to assess the correlation between genetic and geographic distances.

```r
# Step 4: Perform Mantel tests for each distance bin
library(vegan)
```

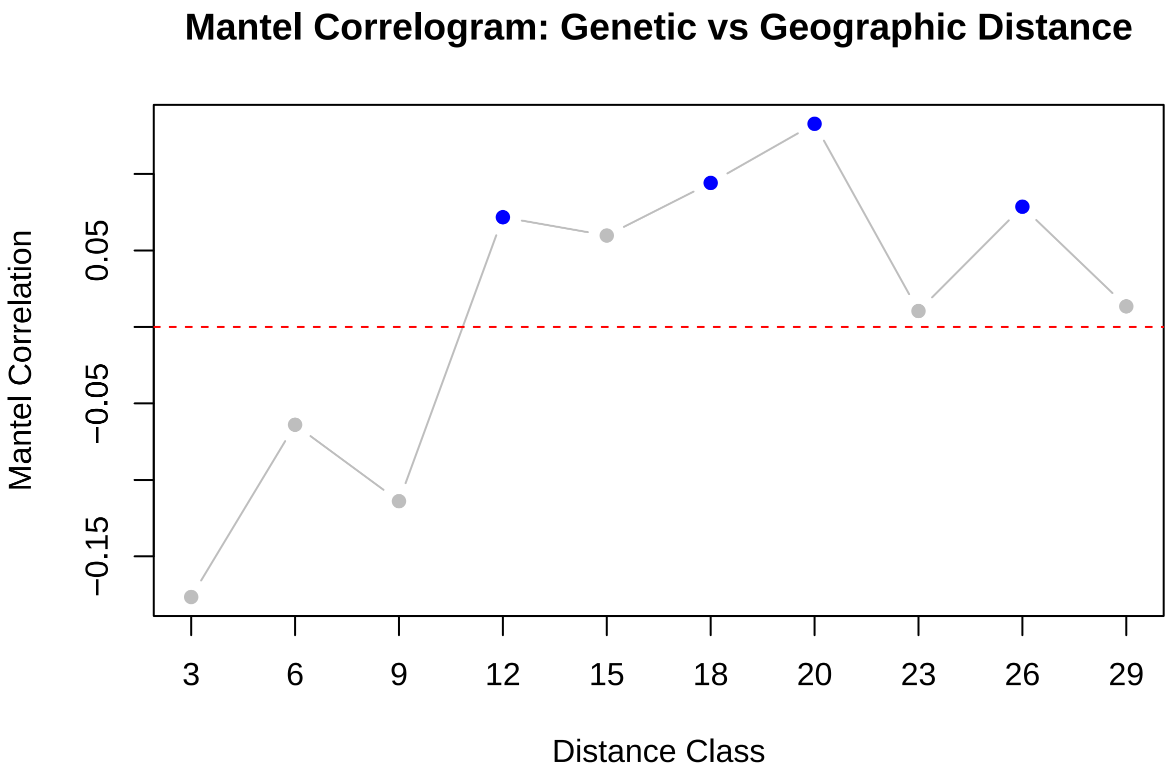
```
## Loading required package: permute

##
## Attaching package: 'permute'

## The following object is masked from 'package:seqinr':
##
##     getType

## Loading required package: lattice

## This is vegan 2.6-8
```

```r
# Initialize a vector to store the Mantel correlation coefficients
mantel_results <- numeric(num_bins)
mantel_p <- 1:num_bins

# Loop over each bin and perform a Mantel test
for (i in 1:num_bins) {
  # Create a mask for the current bin (TRUE for distances in this bin)
  bin_mask <- as.matrix(Dgeo) >= breaks[i] & as.matrix(Dgeo) < breaks[i + 1]

  # Skip bins with no data
  if (sum(bin_mask) == 0) next

  # Perform Mantel test for this bin
  mantel_test <- mantel(as.dist(bin_mask), as.dist(Dgen),
                        method = "pearson", permutations = 199)

  # Store the Mantel statistic (correlation)
  mantel_results[i] <- mantel_test$statistic
  mantel_p[i] <- mantel_test$signif
}
```

Step 5: Plot the Mantel Correlogram

We plot the Mantel correlation coefficients across the distance bins to visualize the relationship.

```r
colors =  c("gray","gray", "gray", "blue", "gray", "blue", "blue", "gray", "blue", "gray")

# Step 5: Plot the Mantel correlogram
plot(1:num_bins, xaxt='n', mantel_results, type = "b", pch = 16, col = colors,
     xlab = "Distance Class", ylab = "Mantel Correlation",
     main = "Mantel Correlogram: Genetic vs Geographic Distance")


# Replace x-axis labels with 'custom_labels'
custom_labels <- round(breaks[2:11],0)
axis(1, at=1:num_bins, labels=custom_labels)

# Add a horizontal line at 0 for reference
abline(h = 0, lty = 2, col = "red")
```



Questions: What is the range of spatial autocorrelation in P. vulgaris in site A25?

Based on a plot of genetic distance against Euclidean distance? A: We can't really tell the lower end of the range, but it only shows positive correlation up to about 15 km.

Based on where the Mantel correlogram reaches 0? A: The correlogram shows positive r values from ~10-22 km, indicating the range of spatial autocorrelation in P. vulgaris in site A25 is from ~10-22 km in pairwise euclidean distances.

Based on statistical significance tests for the Mantel correlogram (with default settings: one-sided alternative "less", Holm's adjustment)? A: The significant Mantel's r values for the bins at 12, 18, 20, 26 indicate that there is spatial autocorrelation 10 km to ~26 km, so this is a little larger of a range then we could see in the original genetic vs geographic distance plot.

Interpretation:

Range of Spatial Autocorrelation from the Mantel Correlogram Mantel Correlogram Interpretation: The Mantel correlogram shows Mantel correlation coefficients plotted across geographic distance bins. This allows you to assess at which spatial scales genetic similarity (spatial autocorrelation) is strongest.

Range of Spatial Autocorrelation:

The range is the largest geographic distance bin where the Mantel correlation coefficient is significantly greater than 0 (positive autocorrelation). When the Mantel coefficient reaches 0 (or dips below it), this marks the end of spatial autocorrelation, meaning that at larger distances, genetic similarity is no longer influenced by geographic proximity.

Using Statistical Significance to Identify the Range One-Sided Test (Alternative = "less"): The one-sided alternative hypothesis (alternative = "less") tests whether there is positive spatial autocorrelation, i.e., whether genetic distance increases with geographic distance. This is relevant for isolation-by-distance (IBD) models.

Holm's Adjustment for Multiple Tests: Since the Mantel correlogram involves multiple distance classes, Holm's adjustment ensures that statistical significance is corrected for multiple comparisons, reducing the risk of false positives.

How to Interpret Statistical Significance:

For each distance bin, check if the p-value (adjusted using Holm's method) is statistically significant (e.g., $p < 0.05$). Significant positive Mantel correlation (with alternative = "less") in early bins indicates that genetic similarity is highest at smaller geographic scales. The range of spatial autocorrelation is the largest distance bin where the Mantel test remains significant. Beyond this range, genetic similarity no longer follows a geographic pattern.

Moran's I : Lab 5 bonus

Calculate and test Moran's I for population-level genetic diversity data, allelic richness, using the genind object from above

1. Load Required Libraries and Prepare Data

```r
library(adegenet)
library(spdep)

# Ensure the genind object is loaded
Flr.genind
```

```
## /// GENIND OBJECT /////////
##
##  // 221 individuals; 7 loci; 105 alleles; size: 134 Kb
##
##  // Basic content
##    @tab:  221 x 105 matrix of allele counts
##    @loc.n.all: number of alleles per locus (range: 8-25)
##    @loc.fac: locus factor for the 105 columns of @tab
##    @all.names: list of allele names for each locus
##    @ploidy: ploidy of each individual  (range: 2-2)
##    @type:  codom
##    @call: df2genind(X = Flr[, c(6:12)], sep = ":", ncode = NULL, ind.names = Flr$ID,
##     loc.names = NULL, pop = Flr$Population, NA.char = "", ploidy = 2,
##     type = "codom", strata = NULL, hierarchy = NULL)
##
##  // Optional content
##    @pop: population of each individual (group size range: 14-56)
##    @other: a list containing: xy
```

```
# Summarize allelic richness
Richness <- PopGenReport::allel.rich(Flr.genind, min.alleles = NULL)
```

```
## Registered S3 method overwritten by 'pegas':
##   method      from
##   print.amova ade4
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
## Registered S3 methods overwritten by 'genetics':
##   method      from
##   print.locus gstudio
##   [.haplotype pegas
```

```
# Extract the mean allelic richness (Ar) for each population
allelic_richness <- Richness$mean.richness


# Use Flr to combine coordinates and population labels
coords_df <- data.frame(pop=Flr$Population, x = Flr$X, y = Flr$Y)

# Aggregate (mean lat/long) coordinates by population using aggregate()
pop_coords <- aggregate(. ~ pop, data = coords_df, FUN = mean)

str(pop_coords)
```

```
## 'data.frame':    7 obs. of  3 variables:
##  $ pop: chr  "A03" "A21" "A25" "A26" ...
##  $ x  : num  4431317 4426926 4422658 4422708 4426037 ...
##  $ y  : num  5429359 5427168 5425361 5425138 5423338 ...
```

2. Create a Spatial Weights Matrix

The spatial weights matrix defines relationships between populations based on their coordinates.

```
# convert coords into a matrix
coords_matrix <- as.matrix(pop_coords[, c("x", "y")])


# Create a distance-based spatial weights matrix
nb <- dnearneigh(coords_matrix, d1 = 0, d2 = 10000)  # Adjust d2 (distance threshold) based on study ar

# Convert neighbors list to a spatial weights matrix
listw <- nb2listw(nb, style = "W")  # W = row-standardized weights
```

3. Calculate Moran's I

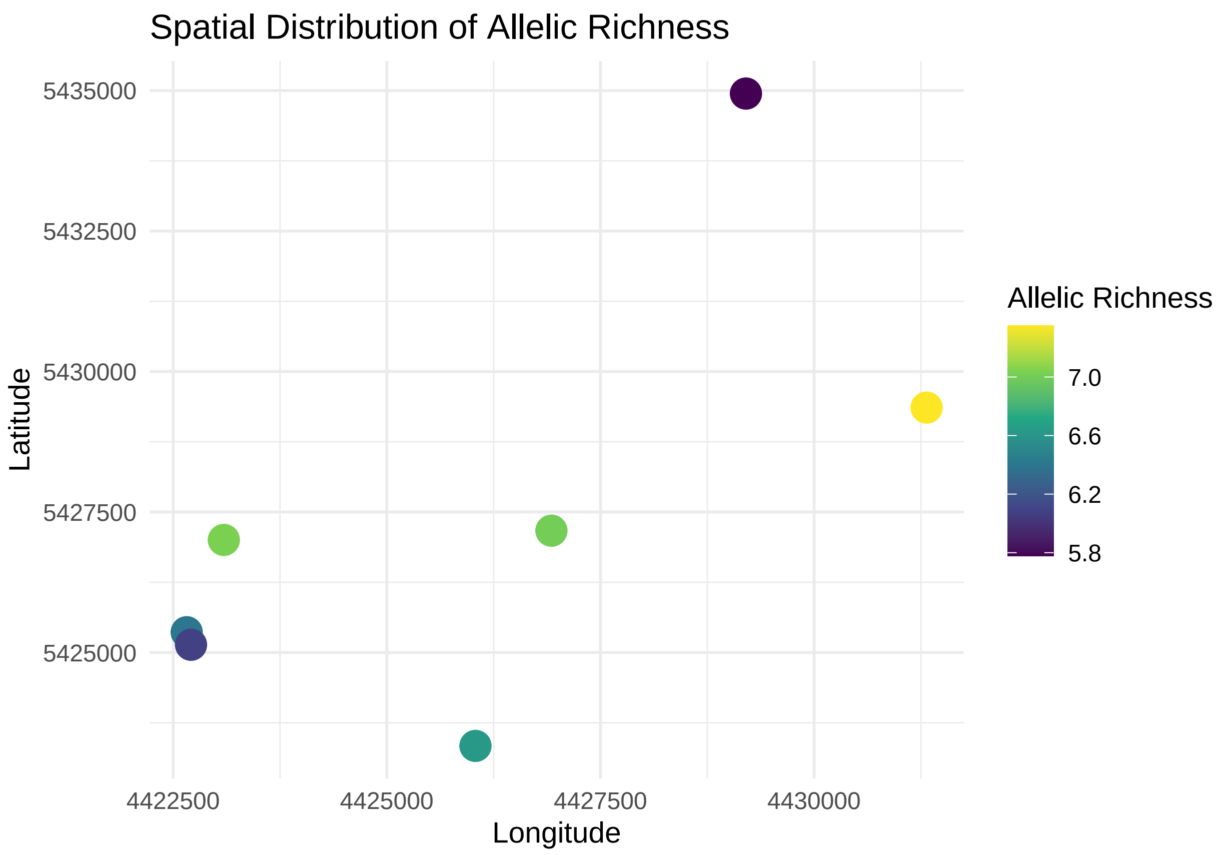
```
# Perform Moran's I test for allelic richness
moran_result <-
```

```r
spdep::moran.test(allelic_richness, listw, alternative="greater")
# Print Moran's I result
print(moran_result)
```

```
##
##  Moran I test under randomisation
##
## data:  allelic_richness
## weights: listw
##
## Moran I statistic standard deviate = -1.9265, p-value = 0.973
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic       Expectation          Variance
##       -0.38923589       -0.16666667        0.01334766
```

Optional, plot coordinates:

```r
library(ggplot2)

# Combine allelic richness with population coordinates
pop_coords$allelic_richness <- allelic_richness

# Plot
ggplot(pop_coords, aes(x = x, y = y, color = allelic_richness)) +
  geom_point(size = 5) +
  scale_color_viridis_c() +
  labs(title = "Spatial Distribution of Allelic Richness",
       x = "Longitude", y = "Latitude", color = "Allelic Richness") +
  theme_minimal()
```


Spatial Distribution of Allelic Richness
Latitude
5435000
5432500
5430000
5427500
5425000
Longitude
4422500
4425000
4427500
4430000
Allelic Richness
7.0
6.6
6.2
5.8